INTERNATIONAL STANDARD

ISO 10303-218

First edition
2004-11-15

Industrial automation systems and integration — Product data representation and exchange —

Part 218: Application protocol: Ship structures

Systèmes d'automatisation industrielle et intégration — Représentation et échange de données de produits —

Partie 218: Protocole d'application: Structures de navires

Reference number
ISO 10303-218:2004(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Figures

Tables

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/ IEC Directives, Part 2.

Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this part of ISO 10303 may be the subject of patent rights. ISO is not to be held responsible for identifying any or all such patent rights.

ISO 10303-218 was prepared by Technical Committee ISO TC 184, *Industrial automation systems and integration*, Subcommittee SC 4, *Industrial data*.

ISO 10303 is organized as a series of parts, each published separately. The structure of ISO 10303 is described in ISO 10303-1.

Each part of ISO 10303 is a member of one of the following series: description methods, implementation methods, conformance testing methodology and framework, integrated generic resources, integrated application resources, application protocols, abstract test suites, application interpreted constructs, and application modules. This part is a member of the application protocol series.

A complete list of parts of ISO 10303 is available from the Internet:

<http://www.tc184-sc4.org/titles/>

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation of product information and for the exchange of product data. The objective is to provide a neutral mechanism capable of describing products throughout their life cycle. This mechanism is suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases, and as a basis for archiving.

This part of ISO 10303 is a member of the application protocol series. This part of ISO 10303 specifies an application protocol (AP) for the definition of ship structures and its related data.

This application protocol defines the context, scope and information requirements for the communication of ship structural data and specifies the integrated resources necessary to satisfy these requirements. This part of ISO 10303 is one of a series of ship building application protocols, that together aim to provide an integrated computer interpretable product model for ships.

The series of shipbuilding industry application protocols assumes that the ship product model can be divided into separate ship systems; each covering a key element of the ship for its whole life cycle. These key elements are:

— ship arrangements;

— ship moulded forms;

— ship distribution systems.

NOTE 1 Distribution systems include piping, heating, ventilation and air conditioning, hydraulics, pneumatics, electrical, and cable way.

— ship structures;

— ship mechanical systems.

NOTE 2 Mechanical systems include cargo handling, machinery and propulsion.

— ship outfit and furnishings;

— combat systems;

— communication;

— navigation;

— operation.

Each separate system is described by one or more different application protocols. The full series of shipbuilding application protocols is depicted in Figure 1. Those aspects of the ship product model that are common to each ship application protocol are described consistently and identically in each application protocol. Within the series of shipbuilding application protocols this part of ISO 10303 details the key element of ship structures.

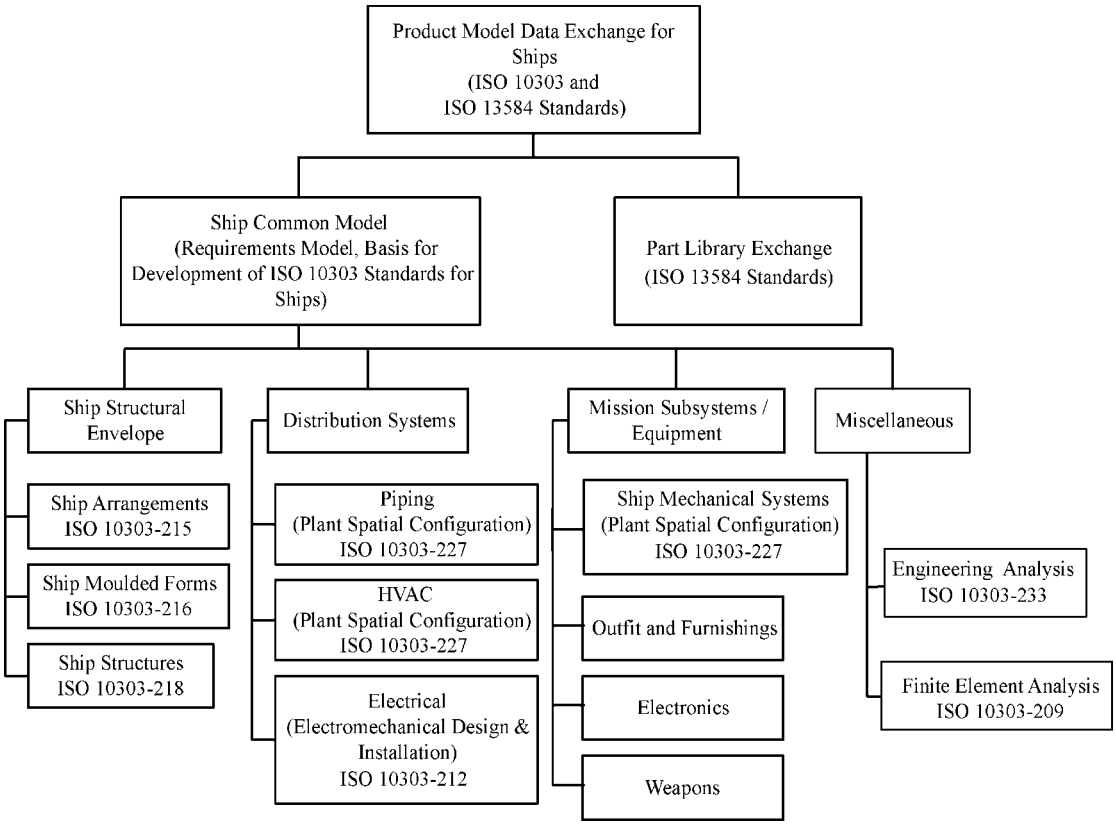


Figure 1 — Shipbuilding application protocols

This part of ISO 10303 specifies an application protocol for the exchange of product data representing a ship structures information. These data may needed to be exchanged or shared between different organizations. Such organizations include ship owners, ship classification societies, design agencies and fabricators. This AP has been developed to support the shipbuilding activities and computer applications associated with the pre-design, the main design, the manufacturing, the maintenance (repair), and the inspection and survey during operation life cycle phases for commercial or naval ships.

Figure 2 illustrates how the ship structures information may be grouped under the following:

— ship common model;

— ship common utilities;

— ship structures;

— manufacturing of ship structures;

— class approvals;

— shape representations.

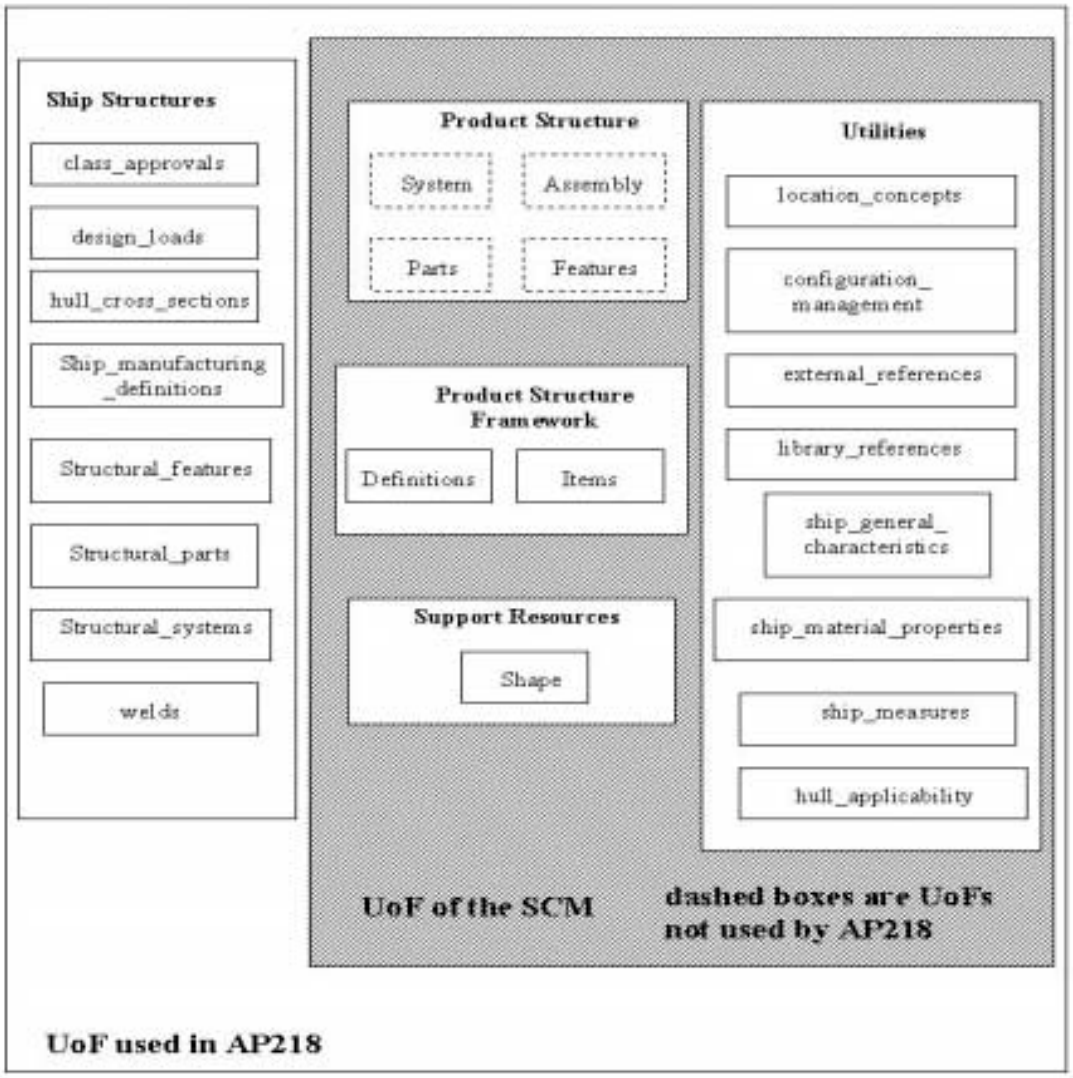


Figure 2 — Data planning model

The unit of functionality (UoF) for ship common model (SCM) specifies the shipbuilding common model which is used for all ship building APs (currently for AP215, AP216, and AP218). The SCM is considered to present the framework of Shipbuilding application protocols. The ship common model UoFs are the basis for the design and manufacturing of ship structures.

The group of UoFs labeled utilities provides the information for configuration management, location concepts, basic geometry and others, which is used by all shipbuilding application protocols.

The group of UoFs labeled ship structures describes the design definition of ship structures during the pre-design and design phase of a ship, the manufacturing definitions of ship structures during the manufacturing phase of a ship, and approvals of the structural parts and hull cross-sections of the ship.

The group of UoFs labeled product definition framework provides the key part of the Ship Common Model and provides general concepts of how to relate things, how to define their properties and how to represent them.

The group of UoFs labeled product structure provides different levels of structuring a product mainly by space, by assembly or by system in a general manner.

The group of UoFs labeled support resources provides all the STEP integrated resources (ISO 10303-41, 42, 43, 44, and 45) used in shipbuilding APs.

Application protocols provide the basis for developing implementations of ISO 10303 and abstract test suites for the conformance testing of AP implementations.

Clause 1 defines the scope of the application protocol and summarizes the functionality and data covered by the AP. Clause 3 lists the words defined in this part of ISO 10303 and gives pointers to words defined elsewhere. An application activity model that is the basis for the definition of the scope is provided in annex F. The information requirements of the application are specified in clause 4 using terminology appropriate to the application. A graphical representation of the information requirements, referred to as the application reference model, is given in annex G. The detailed technical discussion in annex L provides details that define the basis for the definition of the scope of this part of ISO 10303 and details that define the basis of harmonization with other parts of ISO 10303 for product exchange of data for ships.

Resource constructs are interpreted to meet the information requirements. This interpretation produces the application interpreted model (AIM). This interpretation, given in clause 5.1, shows the correspondence between the information requirements and the AIM. The short listing of the AIM specifies the interface to the integrated resources and is given in clause 5.2. Note that the definitions and EXPRESS provided in the integrated resources for constructs used in the AIM may include select list items and subtypes which are not imported into the AIM. The expanded listing given in annex A contains the complete EXPRESS for the AIM without annotation. A graphical representation of the AIM is given in annex H. Additional requirements for specific implementation methods are given in annex C.

Industrial automation systems and integration — Product data representation and exchange — Part 218: Application protocol: Ship structures

1 Scope

This part of ISO 10303 specifies the use of the integrated resources necessary for the scope and information requirements for the exchange of product definition data and its configuration and approval status information for ship structural systems. Configuration in this context pertains to data specific to revision tracking and change history of selected ship structural entities within the product model. Approval pertains to the company internal approval and the classification society approval. This Application Protocol supports the shipbuilding activities and applications associated with the design phase and the manufacturing phase.

NOTE 1 The application activity model in annex F provides a graphical representation of the processes and information flows that are the basis for the definition of the scope of this part of ISO 10303.

NOTE 2 The detailed technical discussion in annex L provides details that define the basis for the definition of the scope of this part of ISO 10303 and details that define the basis of harmonization with other parts of ISO 10303 for product exchange of data for ships.

The following are within the scope of this part of ISO 10303:

— product definition data pertaining to the ship structure which includes hull structure, superstructure and all other internal structures of commercial and naval ships;

— product definition data pertaining to the ship structure;

— product definition data pertaining to the pre-design phase of the ship structure;

— product definition data pertaining to the main design phase of the ship structure;

— product definition data pertaining to the manufacturing phase of the ship structure;

— product definition data pertaining to the product structuring of ships, including the structuring by system and by assemblies within the ship;

— product definition data identifying the ship general characteristics which are relevant to the design of the ship structure.

NOTE The general characteristics include ship main dimensions, designations and principle characteristics, as well as the rules, regulations and standards applicable to the ship. It also includes lightships weight distribution and free-board characteristics for the purpose of design and design approvals.

— product definition data pertaining to the ship global coordinate system, local coordinate systems and spacing grids, which are used for defining the geometry of the ship structure;

— product definition data pertaining to the geometrical representation of the ship structure parts and assemblies;

— product definition data pertaining to the hull plating and the stiffener profiles, and the definition of structural features, which comprise the ship structure parts and assemblies, including functional descriptions.

EXAMPLE Edge, corner and interior cut-outs are examples of structural features.

— product definition data pertaining to the design of the welded connections and joints of ship structure parts and assemblies, including edge preparations and weld type and size;

— product definition data pertaining to the specification of transverse cross-sections through the ship structure for the purpose of approval of strength;

— product definition data pertaining to ship design loads, including shear forces and bending moments acting on the ship structure, for the purpose of determining the longitudinal strength of the ship;

— product definition data pertaining to the weights and centres of gravity of the ship structure parts and assemblies;

— product definition data pertaining to the materials of ship structure, required to manufacture the ship or a part of it;

— product definition data pertaining to the configuration management of the ship structure, including approval, versioning and change administration;

— product definition data pertaining to external references, technical documentation and other supporting concepts which are necessary and pertinent to the design and manufacture of the ship structure parts and assemblies.

The following are outside the scope of this part of ISO 10303:

— product definition data pertaining to the ship structure at the operation and de-commissioning phases of the ship life cycle;

— business data for the management of a ship development project, such as budgets, schedules and resource requirements;

— product definition data pertaining to the direct calculation of ship structure in the design stage;

— product definition data pertaining to the coating of structural parts as well as the production tolerances;

— product definition data pertaining to the ship subdivision;

— product definition data pertaining to the ship distribution systems;

EXAMPLE The electrical, piping and HVAC systems.

— product definition data pertaining to the ship machinery and propulsion systems;

— product definition data pertaining to the ship outfit and furnishing;

EXAMPLE Ladders, some foundations, pre-assembled hull outfit items, doors, and the like.

— product definition data pertaining to ship hull structure parts which are manufactured by forging or casting.

EXAMPLE Stern frames, rudder horns and propeller shaft brackets.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 657-19 *Hot-rolled steel sections — Part 19: Bulb flats (metric series) - Dimensions, sectional properties and tolerances*

ISO/IEC 8824-1 *Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation.*

ISO 10303-1 *Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles*.

ISO 10303-11 *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual*.

ISO 10303-21 *Industrial automation systems and integration — Product data representation and exchange — Part 21: Implementation methods: Clear text encoding of the exchange structure.*

ISO 10303-22 *Industrial automation systems and integration — Product data representation and exchange — Part 22: Implementation methods: Standard data access interface.*

ISO 10303-28 *Industrial automation systems and integration — Product data representation and exchange — Part 28: Implementation methods: XML representations of EXPRESS schemas and data.*

ISO 10303-31 *Industrial automation systems and integration — Product data representation and exchange — Part 31: Conformance testing methodology and framework: General concepts.*

ISO 10303-41 *Industrial automation systems and integration — Product data representation and exchange — Part 41: Integrated generic resource: Fundamentals of product description and support.*

ISO 10303-42 *Industrial automation systems and integration — Product data representation and exchange — Part 42: Integrated generic resource: Geometric and topological representation.*

ISO 10303-43 *Industrial automation systems and integration — Product data representation and exchange — Part 43: Integrated generic resource: Representation structures.*

ISO 10303-45 *Industrial automation systems and integration — Product data representation and exchange — Part 45: Integrated generic resource: Materials.*

ISO 10303-501 *Industrial automation systems and integration — Product data representation and exchange — Part 501: Application interpreted construct: Edge-based wireframe*.

ISO 10303-508 *Industrial automation systems and integration — Product data representation and exchange — Part 508: Application interpreted construct: Non-manifold surface.*

ISO 10303-510 *Industrial automation systems and integration — Product data representation and exchange — Part 510: Application interpreted construct: Geometrically bounded wireframe.*

ISO 10303-511 *Industrial automation systems and integration — Product data representation and exchange — Part 511: Application interpreted construct: Topologically bounded surface.*

ISO 10303-514 *Industrial automation systems and integration — Product data representation and exchange — Part 514: Application interpreted construct: Advanced boundary representation.*

ISO 13584-26 *Industrial automation systems and integration — Parts library — Part 26: Logical resource: Information supplier identification*

ISO 13584-42 *Industrial automation systems and integration — Parts library — Part 42: Description methodology: Methodology for structuring part families*

3 Terms, definitions and abbreviated terms

3.1 Terms defined in ISO 10303-1

For the purposes of this document, the following terms defined in ISO 10303-1 apply.

— abstract test suite (ATS);

— application;

— application activity model (AAM);

— application context;

— application interpreted model (AIM);

— application object;

— application protocol (AP);

— application reference model (ARM);

— computer aided design (CAD);

— computer aided manufacture (CAM);

— conformance class;

— conformance requirement;

— data;

— data exchange;

— generic resource;

— information;

— integrated generic resource;

— integrated resource;

— interpretation;

— model;

— PICS proforma;

— product;

— product data;

— product information model;

— resource construct;

— test purpose;

— unit of functionality (UoF);

— validation.

3.2 Terms defined in ISO 10303-21

For the purposes of this document, the following term defined in ISO 10303-21 applies.

— implementation method.

3.3 Terms defined in ISO 10303-31

For the purposes of this document, the following terms defined in ISO 10303-31 apply.

— conformance testing;

— postprocessor;

— preprocessor.

3.4 Terms defined in ISO 10303-41

For the purposes of this document, the following terms defined in ISO 10303-41 apply.

— address;

— approval status;

— area measure;

— date and time;

— derived unit;

— identifier;

— label;

— length measure;

— mass measure;

— named unit;

— organization;

— person;

— person and organization;

— plane angle measure;

— positive length measure;

— ratio measure;

— shape representation;

— text;

— time measure;

— unit;

— volume measure.

3.5 Terms defined in ISO 10303-42

For the purposes of this document, the following terms defined in ISO 10303-42 apply.

— axis2 placement 3d;

— boundary;

— bounded curve;

— bounded surface;

— box domain;

— cartesian point;

— circle;

— closed curve;

— connected;

— curve;

— curve on surface;

— curve replica;

— edge;

— edge curve;

— fixed reference swept surface;

— geometric representation context;

— line;

— offset surface

— open curve;

— orientable;

— placement;

— point;

— surface;

— surface curve swept surface;

— vector;

— vertex.

3.6 Terms defined in ISO 10303-43

For the purposes of this document, the following terms defined in ISO 10303-43 apply.

— representation;

— representation context;

— representation item;

— representation relationship.

3.7 Terms defined in ISO 10303-216

For the purposes of this document, the following terms defined in ISO 10303-216 apply.

— after perpendicular;

— aft perpendicular;

— amidships;

— baseline;

— breadth;

— buttock line;

— camber;

— centreline;

— centreplane;

— classification;

— configuration management;

— deck;

— depth;

— forward perpendicular;

— frame;

— furnishings;

— hullform;

— hydrodynamic;

— hydrostatic;

— knuckle;

— length between perpendiculars;

— longitudinal;

— midship;

— moulded form;

— offset;

— outfit;

— sheer;

— station;

— superstructure;

— waterline.

3.8 Other terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.8.1
assembly
a hierarchical building block view of individual structural parts comprising a ship

NOTE Parts are joined together to form successively larger building blocks. These interim products are referred to as assemblies.

3.8.2
bill of material (BOM)
a list of all raw material stock required to fabricate the product

NOTE The BOM may be for an individual part, for an aggregation of parts (EXAMPLE an assembly), or for a selected volumetric region (EXAMPLE a zone) of the ship.

3.8.3
bracket; gusset
a steel plate connecting two structural parts that strengthens the connection

EXAMPLE a bracket may be a short piece of flanged plate, a tee bar, an angle bar, or a piece of flat plate

3.8.4
builder
the company that built the ship

NOTE The builder may be further categorized as either the lead yard builder (that has constructed the first ship in a series or class) or a follow yard builder.

3.8.5
builder hull number
the unique serial number assigned to the ship by the shipbuilder indicating where this ship stands with respect to all the hulls built by this yard

NOTE The first hull would be 1, the tenth would be 10, etc. This hull number also serves as a reference to the design information generated during the design and construction processes.

3.8.6
bulkhead
A term applied to any of the partition walls used for subdividing the interior of the ship into various compartments

NOTE 1 Adapted from ISO 10303-216

NOTE 2 A longitudinal bulkhead may extend over one, several, or all decks. Likewise it may extend over the complete length of a ship or only between certain transverse bulkheads.

NOTE 3 A transverse bulkhead may extend from bottom to top of the ship, or only between certain decks. Likewise it may extend over the complete width of the ship or only between certain longitudinal bulkheads. Bulkheads are often designated by their location, use, kind of material, tightness, or method of fabrication.

3.8.7
centroid
the center of an item, area or volume measured with respect to some defined location

NOTE 1 Adapted from ISO 10303-216

NOTE 2 Within shipbuilding, centroid is expressed as a vertical component measured above the baseline, a transverse component measured from the centerline, and a longitudinal component measured from either amidships or the forward perpendicular.

3.8.8
chine
the intersection of the bottom and the sides of a flat or V-bottomed ship

3.8.9
combat systems
systems on board naval ships used for Communication and Battle

NOTE Communication is carried out for detection of targets, disturbing enemy communication signals and with weapon systems on board the ship. Battle systems are the weapon systems on board the ship.

3.8.10
compartments
the spatial partitioning of the interior volume of a ship

EXAMPLE Physical subdivision of space for segregating dry and liquid cargo, rooms, passengers, crew, machinery, and equipment

NOTE A compartment is a closed bounded volume that is defined by hull form elements, such as longitudinal and transverse bulkheads, decks, hull shell, or non structural joiner bulkheads.

3.8.11
connections and joints
the connectivity between structural parts and other structural parts where the geometry and topology of the connection is static and does not change as a result of the connection

NOTE A connection represents a requirement that two parts be joined and may decompose into lower level connection requirements. Eventually, a connection requirement is fulfilled by a joint that represents a physical realization of the connection requirement.

3.8.12
coordinate system; co_ordinate_system; co ordinate system:
a reference system that associates a unique set of n parameters with each point in an n-dimensional space.

3.8.13
deadrise
the linear distance measured vertical from the ship keel to the turn of the bilge

NOTE A X deadrise would mean that the bottom of the ship would be X mm higher at the outboard location than at the centreline. It is sometimes expressed as a slope, such as X mm per Y m.

3.8.14
deadweight
the difference between the lightship weight and loaded displacement

EXAMPLE The weight of cargo plus weights of fuel, stores, water ballast, fresh water, crew and passengers, and baggage.

3.8.15
declination
a property of the stiffener indicating that it is tilted with respect to the surface, i.e. the web is not perpendicular to the surface. The angle between the web and surface normal is called the declination

3.8.16
definition geometry
a designation for geometry that defines the shape representation of an item

NOTE There is a contractual implication with respect to guaranteed accuracy associated with this type of geometry.

3.8.17
design change process
the processing of a formal request for change to one or more configuration items during the design phase

3.8.18
design load
the load that the ship is designed to carry or to withstand

NOTE It is used to present the design displacement or design strength.

3.8.19
design phase
stage within the ship life-cycle when the ship is being designed

NOTE A design phase can be subdivided into preliminary design, detailed design, and production design.

3.8.20
design zone
an abstract boundary, identifying a region, of a ship with unique requirements or characteristics that must be treated special in the design phase

EXAMPLE These zones carry designations such as Fire Zone, CPS Zone, Subsafe Zone, and Ship Work Authorization Boundary Zone.

3.8.21
diagonal
an auxiliary plane that cuts the hull in a plane perpendicular to the stations and inclined to the longitudinal centreplane of the ship

EXAMPLE a hull form element used to represent the intersection of the hull surface with longitudinally oriented plane, inclined to both the vertical and transverse global coordinate system axes

3.8.22
engineering parts
piping parts, HVAC parts, discrete parts, outfit and furnishing parts

3.8.23
explicit shape
the collection of geometry, conforming to a specific geometric modelling representation defining the shape of a structural part

EXAMPLE such as CSG or B-rep

NOTE There may be more than one explicit shape for a single part. The explicit shape may have embedded in it the geometry associated with the part's features.

3.8.24
flare
the transverse measurement indicating how much greater a waterline's half-breadth is compared to the half-breadth of the design waterline, measured at the same longitudinal reference

NOTE A X mm flare are for a waterline means that the moulded hullform is 2xX mm wider at that waterline than it is at the design waterline.

3.8.25
freeboard
the distance between the waterline and the main deck or weather deck of a ship or between the level of the water and the upper edge of the side of a small boat

3.8.26
functional structural element
a classification of ship structure associated with a particular system view of the ship

3.8.27
geometry of the ship hull
the collection of geometry elements that define the shape of the watertight envelope of the ship

NOTE This includes the underwater shell, the uppermost watertight deck or decks, and the internal watertight transverse and longitudinal bulkheads.

3.8.28
hogging
the situation of a vessel in still water as well as in a seaway, in which a vessel is bending along the longitudinal axis

NOTE The deck of the ship will be in tension, the bottom of the ship will be in compression.

3.8.29
hold
the space allocated for the stowage of cargo

3.8.30
hull number
a number assigned to the ship for its life.

NOTE When combined with a naval ship type abbreviation, it identifies the vessel.

3.8.31
hull shell
a hull form element that represents the outermost watertight envelope of a hull form section

NOTE Depending on the context of hull form section the hull shell may represent such things as the outer boundary of the ship itself, the rudder, and sonar dome.

3.8.32
hull form sections
an indicator as to whether or not the hull form section has been faired

NOTE If the section is faired , then all hull form elements contained in the section are assumed to be faired. A complete moulded hull form is fair if all its hull form sections are fair.

3.8.33
implicit shape definition
the specification of the shape of a plate or stiffener part that is independent of any specific geometric modelling representation

NOTE It is defined by a minimal set of geometry elements and attributes such that an explicit shape could be derived from it.

3.8.34
interference shape
a collection of geometry defining the shape of a structural part for performing interference analysis

NOTE This shape may include allowances for access and removal, insulation and shock clearances.

3.8.35
lightship
a ship with no cargo, stores, or personal

NOTE a ship that is complete in every respect, including operation fluids up to the required operating levels, but without any cargo, crew or their effects or any consumables.

3.8.36
main machinery
a characteristic that classifies equipment elements belonging to primary systems

EXAMPLE Such as propulsion, steering and diving.

3.8.37
maintenance phase
the operational stage of the ship when maintenance is required

3.8.38
manufacturer
a company or organization that fabricates, constructs, assembles, or delivers parts or material for the ship

3.8.39
opening
an opening in a structural part to allow penetration by another structural part, penetration of a distribution system part, passage of air or liquid, access through a structural part, lightening of a structural part to improve structural efficiency, or clearance between joined structural parts

3.8.40
predesign phase
the lifecycle phase when a first estimation of main characteristics for the ship are established

NOTE Often based on previous designs

3.8.41
production engineering
the stage within the ship life-cycle when the ship is being built, based on the production design

3.8.42
propulsion system
the system used to move the ship through the water

NOTE This can either be in a longitudinal (like the main propeller) or in a transverse (like bow thruster) direction or a combination of both.

3.8.43
ratholes
the results of removing a portions of a structural part

NOTE The ratholes are often in the shape of a semi-circle along one of its edges, in order to provide clearance

between the part and a seam weld on an abutting part.

3.8.44
raw material stock
the plate and structural shape material that is purchased from either a steel mill or distributor, from which structural parts are cut

NOTE Typically, catalogues identify the range of sizes and cross-sectional shapes that can be selected. This raw material stock may be associated with a specific certification, heat treatment and chemical composition by the mill at the time of its manufacture.

3.8.45
reference geometry
a hull form element representing information symbolically defined in the hull form model and not intended to be an exact or complete geometric definition

NOTE Reference geometry may be used when the relationship between product model components is being highlighted, not the geometric description.

EXAMPLE The outline of the propeller blade or the shaft centerline for a moulded hull form.

3.8.46
rule length
the distance on the estimated summer load line, from the fore side of the stem to the after side of the rudder post or sternpost; where there is no rudder post or sternpost, the length is the distance to the centerline of the rudder stock. The rule length is not to be less than 96% and need not be greater than 97% of the length on the summer load line. The forward end of the rule length is to coincide with the fore side of the stem on the waterline on which the rule length is measured.

3.8.47
sagging
the situation of a vessel in still water as well as in a seaway, in which a vessel is bending along the longitudinal axis: the bottom of the ship would be in tension, the deck will be in compression

NOTE This is a reverse form of hogging with excess weight amidships.

3.8.48
ship class
the grouping of multiple ships built from a single design, identified as similar by the use of a class name

3.8.49
ship offset
a point used as a coordinate for a location on a ship hull form, appendage, or deck

3.8.50
ship range
a measure of a ship ability to remain at sea in order to perform a mission

NOTE It typically refers to a combination of distance and speed.

3.8.51
sight edge
a line that lies on the hull surface and defines the longitudinal seam between adjacent plates

NOTE These shapes are controlled, to a great extent, by appearance and aesthetic appeal.

3.8.52
special ship parts
ship parts that are formed by forging or casting processes

3.8.53
system penetrations
the intersection of piping, HVAC, or outfitting part with a structural part is a non-structural system penetration or the intersection of one structural part with another structural part is a structural system penetration

NOTE Depending on the location, compensation may be required to be added to the structural part being penetrated to compensate for lost material.

3.8.54
tanks
Compartments for liquids or gases

NOTE They may be formed by the ship structure as double bottom tanks, peak tanks, deep tanks, etc., or may be independent of ship structure and installed on special supports.

3.8.55
tumblehome
the decreasing of a vessel's beam above the water line as it approaches the rail

NOTE Tumblehome is opposite of flare.

3.8.56
weight of steel
the total weight of the structural steel parts that make up the ship

NOTE During the various life cycle phases this may vary from an estimated value during the design phase to a measured value after construction is completed.

3.9 Abbreviated terms

2D	two dimensional
3D	three dimensional
AAM	application activity model
AIC	application interpreted construct

AIM	application interpreted model
ARM	application reference model
AP	application protocol
B-rep	boundary representation solid model
BSU	basic semantical unit
CAD	computer aided design
CAM	computer aided manufacture
CSG	constructive solid geometry
CPS	collective protective systems
FR	frame
GUID	globally unambiguous identifier
HVAC	heating, ventilation and air conditioning
ICLL	IMO load line conversion
ID	identification
IMO	international maritime organization
LFR	longitudinal frame
m	metre
MARPOL	marine pollution
mm	millimetre
NC	Numerical Control
PICS	protocol implementation conformance statement
SI	système international
SOLAS	safety of life at sea
SWBM	still water bending moment
SWSF	still water shear force

TFR	transversal frame
WF	webframe
X [n]	The nth logical element in the list X. This notation may be used to reference implicit values in the list
UoF	units of functionality

4 Information requirements

This clause specifies the information required for ship structures. The information requirements are specified as a set of units of functionality, application objects, and application assertions. These assertions pertain to individual application objects and to relationships between application objects. The information requirements are defined using the terminology of the subject area of this application protocol.

NOTE 1 A graphical representation of the information requirements is given in annex G.

NOTE 2 The information requirements correspond to those of the activities identified as being within the scope of this application protocol in annex F.

NOTE 3 The mapping table specified in 5.1 shows how the integrated resources and application interpreted constructs are used to meet the information requirements of this application protocol.

4.1 Units of functionality

This subclause specifies the units of functionality for the Ship Structures application protocol. This part of ISO 10303 specifies the following units of functionality:

— class_approvals;

— configuration_management;

— definitions;

— design_loads;

— external_references;

— hull_class_applicability;

— hull_cross_sections;

— items;

— library_reference;

— location_concepts;

— product_structures;

— shapes;

— ship_general_characteristics;

— ship_manufacturing_definitions;

— ship_material_properties;

— ship_measures;

— structural_features;

— structural_parts;

— structural_systems;

— welds.

The units of functionality and a description of the functions that each UoF supports are given below. The application objects included in the UoFs are defined in 4.2.

4.1.1 class_approvals

The class_approvals UoF specifies the Definition for classification approval of structural parts, systems, plates, profiles, and hull cross sections.

NOTE Definition objects may have a class approval status attached. Subtypes of a Definition object indicate the approval status.

The following application objects are used by the class_approvals UoF:

— Corrugated_part_approval;

— Corrugated_part_rejection;

— Hull_cross_section_approval;

— Hull_cross_section_rejection;

— Hull_cross_section_result;

— Plate_approval;

— Plate_rejection;

— Plate_renewal_definition;

— Profile_approval;

— Profile_rejection;

— Profile_renewal_definition;

— Reason_for_decision;

— Structural_class_approval;

— Structural_class_approval_result;

— Structural_class_conditional_approval;

— Structural_class_rejection.

4.1.2 configuration_management

The configuration_management UoF covers the characterization of a definition under these principal aspects: its approval, its controlled changes, and the identity and the relationship of different versions within a version history.

The approval information provide when, who and what has been approved and to what level of approval, as well as how approvals are related to each other. The controlled changes information describes when and who modified what definition. Also it describes the impact of the change in terms of whether or not other definitions are created, modified or deleted.

Versions describe what definition is subject to versioning and how different versions are related to each other to provide a version history.

The following application objects are used by the configuration management UoF:

— Alternative_version_relationship;

— Approval_event;

— Approval_history;

— Change;

— Change_definition;

— Change_impact;

— Change_plan;

— Change_realization;

— Change_request;

— Check;

— Envisaged_version_creation;

— Event;

— Revision;

— Revision_with_context;

— Version_creation;

— Version_deletion;

— Version_history;

— Version_modification;

— Version_relationship;

— Versionable_object_change_event.

4.1.3 definitions

The definitions UoF describes the abstract concept for the definition of items, item structures and item relationships.

The following application objects are used by the definitions UoF:

— Definition;

— Design_definition;

— Functional_definition;

— General_characteristics_definition;

— Library_definition;

— Library_design_definition;

— Library_manufacturing_definition;

— Manufacturing_definition.

4.1.4 design_loads

The design_loads UoF provides documentation on loads that are applied to ship structures to prove their strength. Both actual loads and loads required by classification rules may be specified.

EXAMPLE Bending moments and shear forces in still water or wave conditions are typical design loads.

NOTE A corresponding approval is normally based on artificial load cases that are the basis for strength calculations.

The following application objects are used by the design_loads UoF:

— Bending_moment;

— Design_load;

— Design_still_water_bending_moment;

— Design_still_water_shear_force;

— Design_swsf_values;

— Design_vertical_wave_bending_moment;

— Design_vertical_wave_shear_force;

— Maximum_permissible_still_water_bending_moment;

— Maximum_permissible_swbm_values;

— Maximum_permissible_still_water_shear_force;

— Maximum_permissible_swsf_values;

— Shear_force.

4.1.5 external_references

The external_references UoF provides the capability and mechanisms by which references can be made to information in other product model data set or refer to information outside a given data exchange or data sharing context, and defines constructs for the identification and reference of standards and documents defined in external libraries or outside of the scope of ISO 10303.

The following application objects are used by the external_references UoF:

— Document;

— Document_portion;

— Document_reference;

— Document_reference_with_address;

— External_instance_reference;

— External_reference;

— External_storage;

— Universal_resource_locator.

4.1.6 hull_class_applicability

In a class design, all elements are related to the ship class. By default, an element is applicable to all hulls in the class. Hull specific changes result in elements that are applicable to specific hulls. These changes may be applied at the item level or may be applied at the definition level.

EXAMPLE 1 Types of hull applicability at the item level would be adding a new system, compartment, or equipment.

EXAMPLE 2 Types of hull applicability at the definition level would be different revisions are used to call out hull-specific changes to properties or geometry.

The following application objects are used by the hull_class_applicability UoF:

— Hull_applicability.

4.1.7 hull_cross_sections

The hull_cross_sections UoF specifies the two-dimensional (transverse) view of the hull at any longitudinal position on the ship, which is relevant to longitudinal strength.

EXAMPLE The ship is viewed as a structural beam with a length of about the length between the perpendiculars with cross sectional properties varying over this length.

The structural steel works, making up the transverse cross section, are defined in terms of hull structural elements. A hull structural element is a plate area, possibly stiffened and possibly with cutouts, constrained by an underlying geometry, to fulfill structural or other functional roles.

The following application objects are used by the hull_cross_sections UoF:

— Flare_area;

— Hull_cross_section;

— Hull_cross_section_design_definition.

4.1.8 items

The items UoF deals with the generic product structures of a ship, reflected by the concept of items, as well as by the concept of item structures, and item relationships in which instances of items are collected and related to each other in a well defined way.

The following application objects are used by the items UoF:

— Definable_object;

— Global_id;

— Item;

— Item_relationship;

— Item_structure;

— Ship;

— Versionable_object.

4.1.9 library_reference

The library_reference UoF provides the capability and mechanisms by which references can be made to information in external libraries.

EXAMPLE A supplier library of part information is a type of library that may be referenced.

The following application objects are used by the library_reference UoF:

— Boolean_property_value;

— Bsu;

— Class_bsu;

— Integer_property_value;

— Library_element_reference;

— Logical_property_value;

— Number_property_value;

— Property_bsu;

— Property_value;

— Real_property_value;

— String_property_value;

— Supplier_bsu.

4.1.10 location_concepts

The location_concepts UoF specifies the information to locate a ship, or any part of it, in a right-handed 3D cartesian axis system. Also, it specifies the information required to subdivide any axis into intervals so that they form the reference basis for points in the axis system.

NOTE A coordinate system is either the one and only global coordinate system of the product description and root in the hierarchy, or a local coordinate system. Any number of local coordinate systems may exist. Spacing positions may be defined for any of the three global coordinate system axes. If the underlying coordinate system is the global coordinate system, the local origin may be defined with reference to spacing positions.

The following application objects are used by the location concept UoF:

— Buttock_table;

— Frame_table;

— Global_axis_placement;

— Local_co_ordinate_system;

— Local_co_ordinate_system_with_position_reference;

— Longitudinal_position;

— Longitudinal_table;

— Spacing_position;

— Spacing_position_with_offset;

— Spacing_table;

— Station_table;

— Transversal_position;

— Transversal_table;

— Vertical_position;

— Vertical_table;

— Waterline_table.

4.1.11 product_structures

The product_structures UoF describes the product structures of a ship from different points of view. It includes the assembly view of the ship, necessary for both production purposes and block oriented design.

Also, this UoF represents the functional view on a ship in order to highlight the specific aspects on each system type, used in the early design and detail design stages, and also intensively used during classification of a ship. This UoF provides a hierarchical definition of the assemblies and systems representing any number of levels.

The following application objects are used by the product_structures UoF:

— Assembly;

— Assembly_relationship;

— Composite_feature;

— Explicit_feature_design_definition;

— Feature;

— Feature_design_definition;

— Feature_relationship;

— Part;

— Part_relationship;

— System;

— System_design_definition;

— System_relationship.

4.1.12 shapes

The shapes UoF specifies the geometirc information that describes the form, contour, and configuration of the ship.

The following application objects are used by the shapes UoF:

— Advanced_brep_shape;

— Edge_based_wireframe_shape;

— Geometrically_bounded_wireframe_shape;

— Non_manifold_surface_shape.

4.1.13 ship_general_characteristics

The ship_general_characteristics UoF specifies the basic information that details the ship dimension and identification. This information is independent of any geometric context. This information includes scalar values for principal dimensions of a ship, designation information for ship related companies, ship class notation and all relevant rules and regulations. Also the information about the lightships weight distribution and free-board characteristics is included in this UoF.

The following application objects are used by the ship_general_characteristics UoF:

— Carrier;

— Class_and_statutory_designation;

— Class_notation;

— Class_parameters;

— Freeboard_characteristics;

— Lightship_definition;

— Lightship_weight_item;

— Loadline;

— Moment_3d;

— Navy_ship;

— Owner_designation;

— Principal_characteristics;

— Regulation;

— Research_ship;

— Ship_designation;

— Shiptype;

— Shipyard_designation;

— Weight_and_centre_of_gravity,

— Working_ship.

4.1.14 ship_manufacturing_definitions

The ship_manufacturing_definitions UoF specifies modifications to the detailed design geometry to account for part edge stock, part edge preparation, the addition of layout and marking information, and, in the case of curved or bent plate and structural shape parts, flat pattern development. The part geometries are associated with the raw material stock from which they will be cut. Constructs are provided for the use of manual or Numeric Control (NC) methods for laying out and cutting the parts from their raw stock and NC attributes are associated with the layout and marking information. Any production templates necessary for fabricating the parts are also identified.

Also, this UoF specifies the definitions needed by the assembly manufacturing process. The definitions are associated with the assembly orientation, assembly bounding box and footprint, or the weight and the centre of gravity of an assembly. The mechanisms specified by the weld manufacturing definition will be executed during the manufacturing life cycle.

The following application objects are used by the ship_manufacturing_definitions UoF:

— Assembly_bounding_box;

— Assembly_manufacturing_position;

— Assembly_manufacturing_definition;

— Plate_manufacturing_definition;

— Profile_manufacturing_definition;

— Structural_added_material_boundary_relationship;

— Structural_added_material_feature;

— Structural_added_material_feature_design_definition;

— Structural_manufacturing_feature;

— Structural_part_manufacturing_definition;

— Structural_weld_shrinkage_allowance_feature;

— Structural_weld_shrinkage_allowance_feature_design_definition;

— Weld_manufacturing_definition;

— Weld_testing;

— Welding_sequence.

4.1.15 ship_material_properties

The ship_material_properties UoF describes the materials used for building a ship structure or structural parts. It specifies raw materials by physical properties, not a material that is a product which may contain many raw materials. The ship materials include composite materials and homogeneous materials.

The following application objects are used by the ship_material_properties UoF:

— Homogeneous_ship_material_property;

— Ship_material_property.

4.1.16 ship_measures

The ship_measures UoF the information for representing measures for physical quantities.

The following application objects are used by the ship measures UoF:

— Centre_location;

— Derived_unit;

— Named_unit;

— Precision.

Some attributes of application objects in this part of ISO 10303 specify relationships to types of measures. The appropriate unit corresponding to each measure is defined globally in the units attribute of the Ship object (see 4.2.204), or locally in the local_units attribute of the Definition object (see 4.2.56). Units are to be instantiable in either SI units or other systems of units.

NOTE A table specifying the units to be used in the Application Interpreted Model corresponding to each measure type is provided in annex L, section 5.

4.1.17 structural_features

The structural_features UoF describes product items covering definitions of structural cutouts. The structural cutouts specify the nature of cutouts, that is the removal of material from a structural part or a composite structural object, such as panels and assemblies.

NOTE Cutouts are distinguished by the number of part boundaries that they effect into: edge cutouts, 1 boundary effected corner cutouts, at least 2 boundaries effected and interior cutouts (holes), no boundary effected.

The following application objects are used by the structural_features UoF:

— Bevel_design_definition;

— Circular_cutout_design_definition;

— Corner_cutout;

— Corner_cutout_boundary_relationship;

— Corner_cutout_design_definition;

— Drain_hole_cutout_design_definition;

— Edge_cutout;

— Edge_cutout_functional_definition;

— Edge_feature;

— Edge_feature_functional_definition;

— Elliptical_cutout_design_definition;

— Elongated_oval_cutout_design_definition;

— Free_form_interior_cutout_design_definition;

— Interior_cutout;

— Interior_cutout_design_definition;

— Interior_cutout_functional_definition;

— Inward_round_corner_design_definition;

— Outward_round_corner_design_definition;

— Part_edge_cutout_design_definition;

— Position_feature;

— Position_feature_design_definition;

— Position_feature_relationship;

— Rectangular_cutback_corner_design_definition;

— Rectangular_cutout_design_definition;

— Round_corner_design_definition;

— Round_corner_rectangular_cutout_design_definition;

— Round_edge_rectangular_cutout_design_definition;

— Seam;

— Seam_curve_relationship;

— Seam_design_definition;

— Shear_bevel_design_definition;

— Structural_cutout;

— Structural_cutout_boundary_relationship;

— Structural_feature;

— Structural_feature_relationship;

— Structural_part_penetration_relationship;

— Structural_system_penetration_relationship;

— Triangular_cutout_design_definition.

4.1.18 structural_parts

The structural_parts UoF describes ship parts used in a steel structure. A structural part is the generalization of piece parts in a steel structure.

EXAMPLE Types of structural parts are plates and profiles, stiffeners and beams.

A structural part may be a part of many different product structures such as assemblies, boundary topology relations, connections and functional decomposition. Each structural part has mass and material properties defined. Plates are specializations of structural parts and a plate is a part of the realization of the hull. Profiles are specializations of structural parts with a bounding surface defined along a path perpendicular to the surface of the cross section.

The following application objects are used by the structural_parts UoF:

— Angle_bar_cross_section;

— Angle_profile_cross_section;

— Bar_profile_cross_section;

— Bulbflat_cross_section;

— Channel_cross_section;

— Channel_profile_cross_section;

— Circular_hollow_profile_cross_section;

— Corrugated_part;

— Corrugated_part_design_definition;

— Corrugation;

— Explicit_profile_cross_section;

— Flanged_plate_cross_section;

— Flanged_profile_cross_section;

— Flat_bar_cross_section;

— Library_profile_cross_section;

— Non_circular_hollow_profile_cross_section;

— Parametric_profile_cross_section;

— Plate;

— Plate_boundary_relationship;

— Plate_design_definition;

— Plate_functional_definition;

— Plate_relationship;

— Profile;

— Profile_boundary_relationship;

— Profile_cross_section;

— Profile_curve_trace_line;

— Profile_design_definition;

— Profile_functional_definition;

— Profile_relationship;

— Profile_trace_line_relationship;

— Proprietary_profile_cross_section;

— Round_bar_cross_section;

— Section_properties;

— Square_tube_cross_section;

— Structural_part;

— Structural_part_design_definition;

— Structural_part_functional_definition;

— Structural_part_relationship;

— Structural_part_symmetry_relationship;

— T_bar_cross_section;

— Twist_location;

— W_shape_cross_section.

4.1.19 structural_systems

The structural_systems UoF describes the properties and types of structural systems.

EXAMPLE 1 A hull element can be different types of bulkheads, tanks, decks, and bottoms.

EXAMPLE 2 A panel can be defined as a stiffened plate area defined on a surface, and constrained by a closed bounding curve. The curve may be modified by features such as cutouts, and the inner structure can be affected by holes.

This UoF contains definitions not only for a predefined breakdown into commonly used abstractions, but also allows the possibility for more shipyard specific abstractions.

The following application objects are used by the structural_systems UoF:

— Built_profile;

— Corrugated_structure;

— Corrugated_structure_design_definition;

— Corrugated_structure_boundary_relationship;

— Corrugated_structure_functional_definition;

— Corrugated_structure_relationship;

— Panel_system;

— Panel_system_boundary_relationship;

— Panel_system_curve_boundary;

— Panel_system_design_definition;

— Panel_system_functional_definition;

— Panel_system_plane_boundary;

— Panel_system_relationship;

— Plate_strake;

— Plate_strake_functional_definition;

— Structural_system;

— Structural_system_adjacency_relationship;

— Structural_system_design_definition;

— Structural_system_functional_definition;

— Structural_system_relationship;

— Structural_system_symmetry_relationship.

4.1.20 welds

The welds UoF contains the information requirements necessary to present the welded connectivity between structural parts. When two or more parts are joined together, these requirements are fulfilled by structural parts connection implementation which represents a physical realization of the connection.

The definitions of this UoF consist of weld design definition and welded joint design definition. The design definitions indicate the design requirements of the welds and welded joint to ensure that they satisfy the requirements of the manufacturing process.

The following application objects are used by the welds UoF:

— Beveled_groove_weld;

— Butt_groove_weld;

— Continuous_fillet_weld;

— Electrode_chemical_composition;

— Fillet_weld;

— Groove_weld;

— Intermittent_fillet_weld;

— Spot_seam_weld;

— Structural_part_connection_implementation;

— Structural_part_joint;

— Structural_part_joint_design_definition;

— Weld;

— Weld_design_definition;

— Weld_filler_material;

— Welded_joint;

— Welded_joint_design_definition.

4.2 Application objects

This subclause specifies the application objects for the Ship Structures application protocol. Each application object is an atomic element that embodies a unique application concept and contains attributes specifying the data elements of the object. The application objects and their definitions are given below.

NOTE Those attributes inherited from supertype application objects in the Application Reference Model are not repeated in the subtype application object definitions in this subclause. Refer to the appropriate supertype application object definition for the specification of these inherited attributes.

4.2.1 Advanced_brep_shape

An Advanced_brep_shape is a shape representation that is described in ISO 10303-514.

4.2.2 Alternative_version_relationship

An Alternative_version_relationship is a relationship between two Versionable_object objects (see 4.2.263) of the same type. Each Versionable_object is an alternative to the other Versionable_object.

The data associated with a Alternative_version_relationship are the following:

— alternative_1;

— alternative_2;

— reason.

4.2.2.1 alternative_1

The alternative_1 specifies an alternative design for the Versionable_object (see 4.2.263) defined by alternative_2. See 4.3.1 for the application assertion.

4.2.2.2 alternative_2

The alternative_2 specifies an alternative design for the Versionable_object (see 4.2.263) defined by alternative_2. See 4.3.1 for the application assertion.

4.2.2.3 reason

The reason specifies the description of the relationship between the two Versionable_object (see 4.2.263) objects.

4.2.3 Angle_bar_cross_section

An Angle_bar_cross_section is a type of Angle_profile_cross_section (see 4.2.5) identified by an angular shape consisting of a web and a flange.

NOTE 1 ISO 657-1 and ISO 657-2 specify dimensions for some standard equal-leg and unequal-leg angles.

NOTE 2 Figure 3 illustrates the attributes and the shape of a Angle_bar_cross_section.

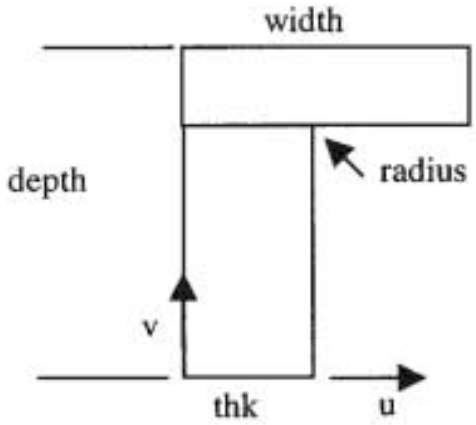


Figure 3 — Angle_bar_cross_section

The data associated with a Angle_bar_cross_section are the following:

— depth;

— radius;

— thk;

— width.

4.2.3.1 depth

The depth specifies the depth of the web including the thickness of the flange.

4.2.3.2 radius

The radius specifies the radius at the connection between the web and the flange.

4.2.3.3 thk

The thk specifies the thickness of the web and the flange.

4.2.3.4 width

The width specifies the width of the flange including the thickness of the web.

4.2.4 Angle_profile_cross_section

An Angle_profile_cross_section is a type of Parametric_profile_cross_section (see 4.2.151) identified by an angular shape consisting of a web and a flange. Each Angle_profile_cross_section is an Angle_-bar_cross_section (see 4.2.3).

NOTE ISO 657-1 and ISO 657-2 specify dimensions for some standard equal-leg and unequal-leg angles.

4.2.5 Annotation

An Annotation is the text that will be marked on the part during manufacturing to provide part identification or indications of correct part orientation within the ship.

The data associated with an Annotation are the following:

— location;

— text.

4.2.5.1 location

The location indicates the placement of the annotation.

4.2.5.2 text

The text indicates the contents of the annotation.

4.2.6 Approval_event

An Approval_event is a type of Event (see 4.2.79) that records a change in the status of the organizational review and approval of some product data.

The data associated with an Approval_event are the following:

— approval_reference;

— result;

— user_defined_result.

4.2.6.1 approval_reference

The approval_reference specifies the Approval_history effected by the event. Every Approval_event must refer to exactly one Approval_history. See 4.3.2 for the application assertion.

NOTE The Event (see 4.2.79) will be unique with respect to its associated approval together with the date and time the event occurred.

4.2.6.2 result

The result specifies that the product data for a version of the design is to be reviewed, or has been reviewed by an authorized member of the organization and has been approved, rejected, or has some other project-specific status.

The value of the result shall be one of the following:

— approved;

— rejected;

— unapproved;

— user_defined.

NOTE See 4.2.6.2.1 - 4.2.6.2.4 for the definition of each allowable value for result.

4.2.6.2.1 approved

the product data has been reviewed by the appropriate organization and is approved for use in the ship.

4.2.6.2.2 rejected

the product data has been reviewed by the appropriate organization and is not approved for use in the ship. Other product data would normally be created to replace the rejected product data.

4.2.6.2.3 unapproved

the product data has not yet been reviewed or is in the process of being reviewed for approval by the organization.

4.2.6.2.4 user_defined

project-specific approval status code to be determined by two or more exchanging organizations.

4.2.6.3 user_defined_result

The user_defined_result specifies an user-defined approval_status. User_defined_result is only to be given when Approval_event.result value is user_defined. The user_defined_result need not to be specified for a particular Approval_event.

4.2.7 Approval_history

An Approval_history is a collection of all Approval_event (see 4.2.6) objects of a specific type defined for a portion of product data. Each Approval_history may be a Structural_class_approval_result (see 4.2.220) .
The data associated with an Approval_history are the following:

— approvals;

— status;

— subject.

4.2.7.1 approvals

The approvals specifies the sequence of Approval_event objects having occurred up to this point in time. The history must consist of at least one Approval_event. The sequence of Approval_event objects is assumed to be in chronological order. There may be more than one approvals for an Approval_history. See 4.3.2 for the application assertion.

4.2.7.2 status

The status specifies the current approval status.

The value of the status shall be one of the following:

— approved;

— rejected;

— unapproved;

— user_defined.

NOTE See 4.2.7.2.1 - 4.2.7.2.4 for the definition of each allowable value for status.

4.2.7.2.1 approved

the product data has been reviewed by the appropriate organization and is approved for use in the ship.

4.2.7.2.2 rejected

the product data has been reviewed by the appropriate organization and is not approved for use in the ship. Other product data would normally be created to replace the rejected product data.

4.2.7.2.3 unapproved

the product data has not yet been reviewed or is in the process of being reviewed for approval by the organization.

4.2.7.2.4 user_defined

project-specific approval status code to be determined by two or more exchanging organizations.

4.2.7.3 subject

The subject specifies the product data this Approval_history is related to. See 4.3.3 for the application assertion.

NOTE A definition may have zero, one or many associated Approval_history objects. In case the Definition has more than one associated Approval_history, no Approval_history objects will be the same.

4.2.8 Assembly

An Assembly is a type of both Item (see 4.2.115) and Item_structure (see 4.2.117) that defines an implicit hierarchical tree structure of Assembly objects and parts from the viewpoint of manufacturing. Assembly objects are composed of other Assembly objects and Part objects. An Assembly shall be neither directly nor indirectly a subassembly of itself.

EXAMPLE 1 Examples of Assembly views are unit assembly and preconstructed section.

EXAMPLE 2 Figure 4 is an example of an Assembly.

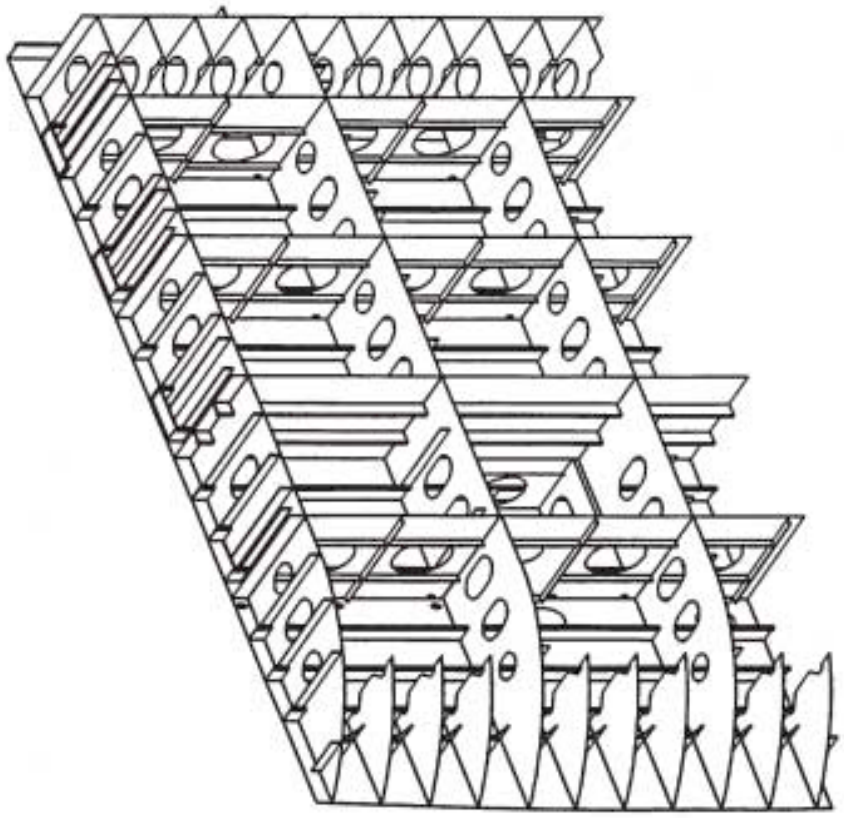


Figure 4 — Assembly

The data associated with an Assembly are the following:

— the_class.

4.2.8.1 the_class

The the_class specifies the class of an Assembly based on the where the Assembly is located in the range of the overall assembly tree. The value of the_class is either a human-interpretable name or is one of the following:

— preconstructed_group;

— preconstructed_section;

— production_panel;

— unit_assembly.

NOTE See 4.2.8.1.1 - 4.2.8.1.4 for the definition of each allowable value for the_class.

4.2.8.1.1 preconstructed_group

erection unit that can consist of Assembly objects and potentially unit Assembly objects and individual Part objects, which when completed is erected onto the ship.

4.2.8.1.2 preconstructed_section

an Assembly that may consist of Part, unit Assembly objects or production panels, which when completed is assembled into an erection unit.

4.2.8.1.3 production_panel

a panel that may consist of Part objects or unit Assembly objects which is defined within a production stage .

4.2.8.1.4 unit_assembly

an assembly consisting only of Part objects and no other Assembly objects.

EXAMPLE Built profile.

4.2.9 Assembly_bounding_box

An Assembly_bounding_box is the spatial cube circumscribing an Assembly. An Assembly_bounding_-box is defined by two cartesian points for opposite corners of the bounding box. The edges of the cube are parallel to the coordinate axes.

The data associated with an Assembly_bounding_box are the following:

— point_max;

— point_min.

4.2.9.1 point_max

The point_max specifies the opposite corner of the Assembly_bounding_box from the point_min, which has maximum coordinate values. See 4.3.4 for the application assertion.

4.2.9.2 point_min

The point_min specifies the corner of the Assembly_bounding_box, which has minimum coordinate values. See 4.3.4 for the application assertion.

4.2.10 Assembly_manufacturing_definition

An Assembly_manufacturing_definition is a type of Manufacturing_definition (see 4.2.131) that describes the definition of an Assembly (see 4.2.8) from the view of manufacturing.

The data associated with an Assembly_manufacturing_definition are the following:

— assembly_drawing;

— assembly_method;

— assembly_positions;

— assembly_stage;

— centre_of_gravity;

— defined_for;

— mass;

— production_date.

4.2.10.1 assembly_drawing

The assembly_drawing specifies the drawing on which the assembly is detailed for its planned stage and method of production. See 4.3.7 for the application assertion.

4.2.10.2 assembly_method

The assembly_method specifies the description of the technique to be used to assemble the Assembly.

4.2.10.3 assembly_positions

The assembly_positions define the set of positions in the Assembly_manufacturing_definition that the defined assembly is placed in during production operations. The assembly_positions need not be specified for a particular Assembly_manufacturing_definition. There may be more than one assembly_positions for an Assembly_manufacturing_definition. See 4.3.6 for the application assertion.

4.2.10.4 assembly_stage

The assembly_stage specifies the production stage at which the Assembly is planned for assembly.

4.2.10.5 centre_of_gravity

The centre_of_gravity specifies the center of gravity of an Assembly defined by the Assembly_ manufacturing_definition. See 4.3.8 for the application assertion.

4.2.10.6 defined_for

The defined_for specifies the Assembly_manufacturing objects that are defined by the Assembly_-manufacturing_definition. There may be more than one defined_for for an Assembly_manufacturing_-definition. See 4.3.5 for the application assertion.

4.2.10.7 mass

The mass specifies the mass of an Assembly defined by the Assembly_manufacturing_definition.

4.2.10.8 production_date

The production_date specifies the planned start of the work defined by the stage and method of production represented for an Assembly by the Assembly_manufacturing_definition.

4.2.11 Assembly_manufacturing_position

An Assembly_manufacturing_position specifies an orientation of the defined assembly during production operations for that assembly, and its resultant footprint and bounding box while in this orientation.

The data associated with an Assembly_manufacturing_position are the following:

— assembly_footprint;

— bounding_box;

— orientation.

4.2.11.1 assembly_footprint

The assembly_footprint specifies the shape of the downside of an Assembly as seen by the mechanism supporting the Assembly for the related Assembly_manufacturing_position. An assembly_footprint shall be a bounded curve. The bounded curve shall be a closed curve. The start point and the end point of the bounded curve are identical.

4.2.11.2 bounding_box

The bounding_box specifies the cube circumscribing the Assembly. The bounding_box is defined by the Assembly_manufacturing_position whose Z-axis points in the direction of orientations. See 4.3.9 for the application assertion.

4.2.11.3 orientation

The orientation specifies the vector pointing in the local +Z-axis direction relative to the global coordinate system.

4.2.12 Assembly_relationship

An Assembly_relationship is a type of Item_relationship (see 4.2.116) that describes the association of an Assembly (see 4.2.8) and an item allowed to be member in an Assembly. The related items are restricted to be either of type Assembly or Part. At least one of the related items is of type Assembly.

NOTE The relationship is realized by collecting all Part_relationship objects between item_1 and item_2 into a derived attribute atomic relationship. If one item is an Assembly and the other item is a Part atomic relationships

contains the Part_ relationship objects between this Part and all the Part objects held by the Assembly. If both, item_1 and item_2 are Assembly objects atomic relationships contains all Part_relationship objects between Part objects of the 1 Assembly and Part objects of the 2 Assembly.

The data associated with an Assembly_relationship are the following:

— item_1.

4.2.12.1 item_1

The item_1 specifies the re-declaration of item_1 attribute inherited from Item_relationship in order to constrain it to be of type Assembly that is related to a Part or another Assembly. See 4.3.10 for the application assertion.

4.2.13 Bar_profile_cross_section

A Bar_profile_cross_section is a type of Parametric_profile_cross_section (see 4.2.151) that is identified by a solid round, square, or rectangular shape. Each Bar_profile_ cross_section is one of the following: a Bulb_flat_cross_section (see 4.2.20), a Flat_bar_cross_ section (see 4.2.92), or a Round_bar_cross_-section (see 4.2.194).

NOTE ISO 1035 specifies dimensions for some standard bars.

4.2.14 Bending_moment

A Bending_moment is a type of Design_load (see 4.2.59) that is defined as the physical quantity moment. Bending_moment is either a Design_still_water_bending_monent (see 4.2.60), a Design_vertical_wave_-bending_moment (see 4.2.63), or a Maximum_permissible_still_water_bending_moment (see 4.2.132).

NOTE Figure 5 shows the bending moment in wave.

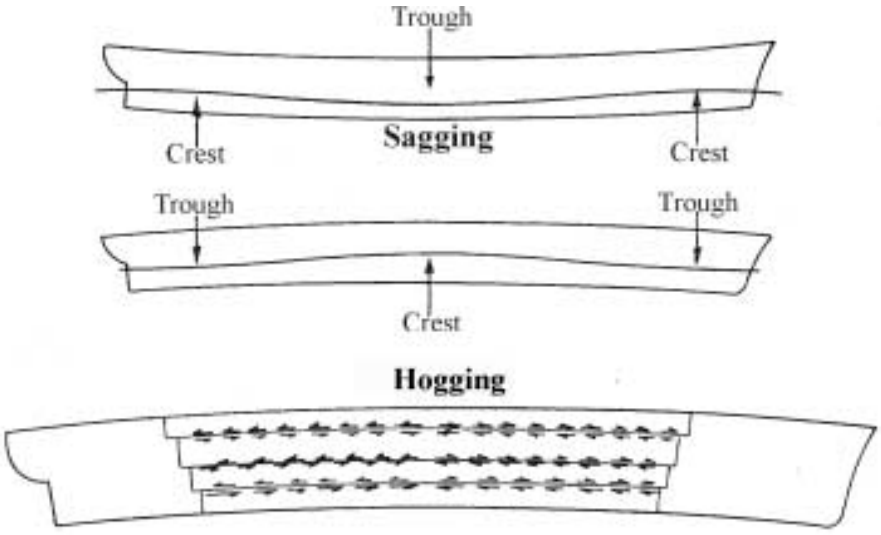


Figure 5 — Bending moment

4.2.15 Beveled_groove_weld

A Beveled_groove_weld is a type of Groove_weld (see 4.2.101) where the parts being welded are of significantly different thickness and the edge or edges are trimmed so as to provide a transition across the joint, with a curve that has a continuous derivative.

The data associated with a Beveled_groove_weld are the following:

— endcut_shape;

— taper;

— taper_angle.

4.2.15.1 endcut_shape

The endcut_shape specifies an indicator used to denote the particular shape of the cut form of plate parts being welded by the bevel groove weld where the parts are of the same or different thickness. The configuration includes the specification of whether the bevel groove weld has a flat, flare or curved faces.

The value of the endcut_shape shall be one of the following:

— double_bevel;

— double_j;

— double_v;

— double_u;

— flare_double_bevel;

— flare_double_v;

— flare_single_bevel;

— flare_single_v;

— single_bevel;

— single_j;

— single_v;

— single_u;

— user_defined.

NOTE See 4.2.15.1.1 - 4.2.15.1.13 for the definition of each allowable value for endcut shape.

4.2.15.1.1 double_bevel

the bevel form which is non-symmetrically open and has a flat cut on one plate part, at both sides of the welded edge also called K-form.

4.2.15.1.2 double_j

the bevel form which is non-symmetrically open and has a curved cut on one plate part, at both sides of the welded edge.

4.2.15.1.3 double_u

the bevel form which is symmetrically open and has curved cut on both plate parts, at both sides of the welded edge.

4.2.15.1.4 double_v

the bevel form which is symmetrically open and has a flat cut on both plate parts, at both sides of the welded edge also called X-form..

4.2.15.1.5 flare_double_bevel

the bevel form which is non-symmetrically open and spread outwards, has cut on one plate part, at both sides of the welded edge.

4.2.15.1.6 flare_double_v

the bevel form which is symmetrically open and spread outwards, has cuts on both plate parts, at both sides of the welded edge.

4.2.15.1.7 flare_single_bevel

the bevel form which is non-symmetrically open and spread outwards, has cut on one plate part, at one side of the welded edge.

4.2.15.1.8 flare_single_v

the bevel form which is symmetrically open and spread outwards, has cuts on both plate parts, at one side of the welded edge.

4.2.15.1.9 single_bevel

the bevel form which is non-symmetrically open and has a flat cut on one plate part, at one side of the welded edge.

4.2.15.1.10 single_j

the bevel form which is non-symmetrically open and has a curved cut on one plate part, at one side of the welded edge.

4.2.15.1.11 single_u

the bevel form which is symmetrically open and has curved cut on both plate parts, at one side of the welded edge.

4.2.15.1.12 single_v

the bevel form which is symmetrically open and has a flat cut on both plate parts, at one side of the welded edge.

4.2.15.1.13 user_defined

other possible bevel cut form of edge defined by the user.

4.2.15.2 taper

The taper specifies an indicator used to denote the particular taper cut at the welded surface of a bevel weld.

The value of the taper shall be one of the following:

— both_sides_taper;

— non_taper;

— one_side_taper.

NOTE See 4.2.14.2.1 - 4.2.14.2.3 for the definition of each allowable value for taper.

4.2.15.2.1 both_sides_taper

both sides taper cut at the welded surface of a bevel weld.

4.2.15.2.2 non_taper

no taper cut at the welded surface of a bevel weld.

4.2.15.2.3 one_side_taper

one side taper cut at the welded surface of a bevel weld.

4.2.15.3 taper_angle

The taper_angle specifies the the measure of the taper angle. The taper_angle need not be specified for a particular Beveled_groove_weld.

4.2.16 Bevel_design_definition

A Bevel_design_definition is a type of Corner_cutout_design_definition (see 4.2.44) that specifies the line or surface obtained when cutting parts of a corner with a line or plane not parallel with the edges or two of three surfaces of the corner. Each Bevel_design_definition may be a Shear_bevel_design_-definition (see 4.2.202).

NOTE See Figure 6 for parameters of a Bevel_design_definition.

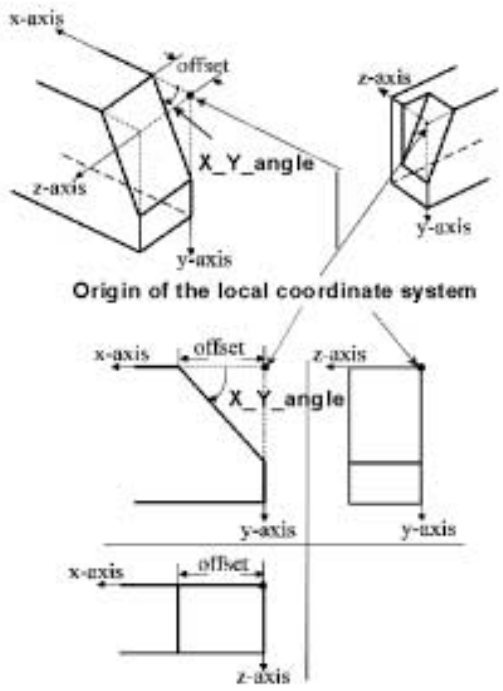


Figure 6 — Bevel_design_definition

The data associated with a Bevel_design_definition are the following:

— offset;

— x_y_angle.

4.2.16.1 offset

The offset specifies the distance from the origin of the local coordinate system along the X-axis.

4.2.16.2 x_y_angle

The x_y_angle specifies the angle at the offset, as shown in Figure 6, between x-axis and the beveled side in the x-y plane.

4.2.17 Boolean_property_value

A Boolean_property_value is a type of Property value (see 4.2.184) that provides a value for a property of type boolean.

The data associated with a Boolean_property_value are the following:

— val.

4.2.17.1 val

The val specifies the value of type boolean.

4.2.18 Bsu

A Bsu identifies a piece of information, which can be a supplier, a class or a property. Each Bsu is either a Supplier_bsu (see 4.2.248), a Class_bsu (see 4.2.37), or a Property_bsu (see 4.2.183).

The data associated with a Bsu are the following:

— code.

4.2.18.1 code

The code specifies the designation of the identification of the information piece.

4.2.19 Built_profile

A Built_profile is a type of both Structural_system (see 4.2.239) and Profile (see 4.2.171) that is made of at least two Structural_part objects (see 4.2.229) by an aggregation process; it can nevertheless be used like a profile.

The data associated with a Built_profile are the following:

— the_class.

4.2.19.1 the_class

The the_class specifies the category of a Built_profile.

The value of the the_class may be the following:

— built_profile_class.

NOTE See 4.2.19.1.1 for the definition of each allowable value for the_class.

4.2.19.1.1 built_profile_class

a Structural_system that is made of at least two Structural_part objects by an aggregation process; can nevertheless be used like a profile.

4.2.20 Bulbflat_cross_section

A Bulbflat_cross_section is a type of Bar_profile_cross_section (see 4.2.13) consisting of a standard manufactured shape.

NOTE 1 Figure 7 shows the shape and definition of geometric parameters of a Bulbflat_cross_section.

NOTE 2 ISO 657-19 specifies dimensions for some standard bulb flats.

The data associated with a Bulbflat_cross_section are the following:

— bulb_radius;

— bulb_width;

— depth;

— tw.

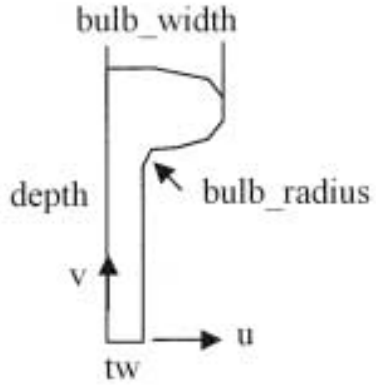


Figure 7 — Bulbflat_cross_section

4.2.20.1 bulb_radius

The bulb_radius specifies the radius of bulb. See ISO 657-19.

4.2.20.2 bulb_width

The bulb_width specifies the bulb width. See ISO 657-19.

4.2.20.3 depth

The depth specifies the overall depth or height of the section. See ISO 657-19.

4.2.20.4 tw

The tw specifies the thickness of the web. See ISO 657-19.

4.2.21 Butt_groove_weld

A Butt_groove_weld is a type of Groove_weld (see 4.2.101) where the parts being welded are of the same approximate thickness and no edge preparation is required.

NOTE An edge preparation is a bevel or a chamfer.

The data associated with a Butt groove weld are the following:

— face_shape.

4.2.21.1 face_shape

The face_ shape specifies an indicator used to denote the particular shape of the cut form of plate parts being welded by the Butt_groove_weld where the parts are of same approximate thickness.

NOTE Figure 8 shows a one-side butt weld, and Figure 9 shows both-sides butt weld.

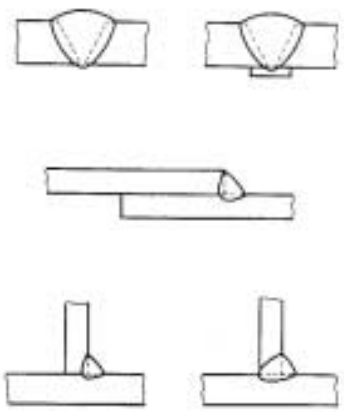


Figure 8 — One side butt weld

The value of the face_shape is one of the following:

— scarf;

— square;

— user_defined.

NOTE See 4.2.21.1.1 - 4.2.21.1.3 for the definition of each allowable value for face_shape.

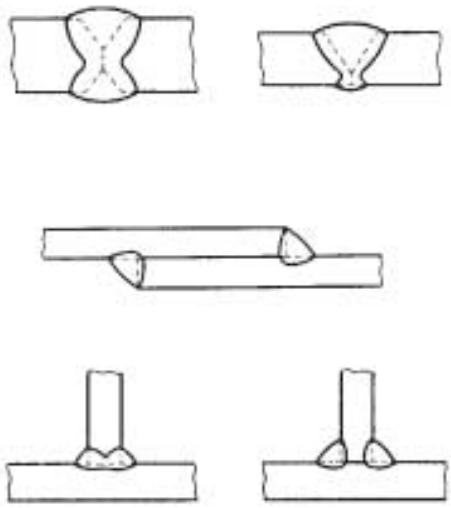


Figure 9 — Both sides butt weld

4.2.21.1.1 scarf

butt shape of parts whose welding surface is broadened by chamfering or cutting away the base metal.

EXAMPLE An I-shape groove, used for thick plate parts, or brazing and forge welding.

4.2.21.1.2 square

butt shape with no edge preparation is required, used for relatively thin plate parts with same approximate thickness.

4.2.21.1.3 user_defined

other butt cut form defined by the user.

4.2.22 Buttock_table

A Buttock_table is a type of Transversal_table (see 4.2.254) that has positions that reference the location of buttocks that are located on the global Y-axis.

4.2.23 Carrier

A Carrier is a type of a Shiptype (see 4.2.207) that represents a Ship that transports goods or passengers.

The data associated with a Carrier is the following:

— has_type.

4.2.23.1 has_type

The has_type specifies the type of the carrier ship.

The value of has_type shall be one of the following:

— barge;

— barge_for_deck_loading;

— barge_for_liquifiedgas;

— barge_for_oil;

— barge_pontoon;

— bulk_carrier;

— car_carrier;

— car_ferry;

— cargo_ship_carrying_passengers;

— chemical_tanker;

— chemical_tanker_type_1;

— container_carrier;

— cruise_liner;

— dry_cargo_vessel;

— ferry;

— gas_carrier;

— general_cargo_carrier;

— highspeedcraft_cargo;

— highspeedcraft_passenger;

— hydrofoil;

— liquified_gas_tanker;

— LNG_carrier;

— LPG_carrier;

— oil_tanker;

— ore_carrier;

— passenger_vessel;

— product_tanker;

— refrigerated_cargo_carrying_ship

— roro_vessel;

— tanker_for_refrigerated_fruit_juice;

— user_defined.

NOTE See 4.2.23.1.1 - 4.2.23.1.31 for definition of allowable values for has_type.

4.2.23.1.1 barge

a Carrier that has no machinery for self-propulsion and is designed for transporting cargo .

4.2.23.1.2 barge_for_deck_loading

a Carrier designed as a service platform for loading decks on Ships.

4.2.23.1.3 barge_for_liquefied_gas

a Carrier that has no machinery for self-propulsion and designed for transporting liquefied gas.

4.2.23.1.4 barge_for_oil

a Carrier that has no machinery for self-propulsion and is designed for transporting oil .

4.2.23.1.5 barge_pontoon

a Carrier with pontoons coupled to a barge serving as a floating utility platform.

4.2.23.1.6 bulk_carrier

a cargo Carrier constructed for the transportation of bulk cargo.

4.2.23.1.7 car_carrier

a Carrier used for transporting automobiles from manufacturer to end-user.

4.2.23.1.8 car_ferry

a Carrier constructed to transport automobiles across water.

4.2.23.1.9 cargo_ship_carrying_passengers

a cargo Carrier constructed for the transport of cargo and passengers.

4.2.23.1.10 chemical_tanker

a cargo Carrier specially designed transport chemicals.

4.2.23.1.11 chemical_tanker_type_1

a cargo Carrier specially designed to transport Type 1 chemicals.

4.2.23.1.12 container_carrier

a cargo Carrier constructed to transport containers.

4.2.23.1.13 cruise_liner

a Carrier constructed for the transport of passengers for pleasure.

4.2.23.1.14 dry_cargo_vessel

a cargo Carrier constructed to transport dry cargo that is not packed but carried loose.

EXAMPLE Types of dry_cargo_vessel are grain or coal.

4.2.23.1.15 ferry

a Carrier constructed to transport passengers, cars, coaches, lorries and trains across water.

NOTE The ferry could be designed for only one type of cargo e.g., passengers.

4.2.23.1.16 gas_carrier

a cargo Carrier specially designed for transporting gaseous products.

4.2.23.1.17 general_cargo_carrier

a cargo Carrier specially designed for transporting general cargo.

NOTE The general_cargo_carrier carries any dry cargo that is not packed but carried loose such as grain or coal.

4.2.23.1.18 highspeedcraft_cargo

a Carrier designed for carrying cargo at high speeds.

4.2.23.1.19 highspeedcraft_passenger

a Carrier designed for carrying passengers at high speeds.

4.2.23.1.20 hydrofoil

a Carrier designed to achieve speed by raising the hull of the vessel out of the water on hydrofoil surfaces, wings, thus avoiding hull drag and wave resistance.

4.2.23.1.21 liquefied_gas_tanker

a cargo Carrier specially designed for transporting liquefied gas products.

4.2.23.1.22 LNG_carrrier

a cargo ship specially designed to transport liquid natural gas

4.2.23.1.23 LPG_carrier

a cargo ship specially designed to transport liquid petroleum gas

4.2.23.1.24 oil_tanker

a cargo Carrier specially designed for transporting oil.

4.2.23.1.25 ore_carrier

a cargo Carrier constructed for transporting ore

EXAMPLE A types of ore is iron ore.

4.2.23.1.26 passenger_vessel

a Carrier constructed for transporting passengers.

4.2.23.1.27 product_tanker

a cargo Carrier specially designed for transporting of oil products.

4.2.23.1.28 refrigerated_cargo_carrying_ship

a cargo Carrier constructed for transporting freight requiring cooling or chilling.

4.2.23.1.29 roro_vessel

a cargo Carrier constructed for the transport of roll-on and roll-off type vehicles.

4.2.23.1.30 tanker_for_refrigerated_fruit_juice

a cargo Carrier specially designed for transporting fruit juice requiring cooling or chilling.

4.2.23.1.31 user_defined

any Carrier type that is not in the list mentioned above (See 4.2.23.1.1 - 4.2.23.1.28). Details should be found in the description attribute for Shiptype (see 4.2.207).

4.2.24 Centre_location

The Centre_location is a location in terms of the three principal axis directions of the Ship global coordinate system.

The data associated with a Centre_location are the following:

— longitudinal_location;

— transversal_location;

— vertical_location.

4.2.24.1 longitudinal_location

The longitudinal_location specifies the distance along the X axis from the origin of the global coordinate system of the Ship.

4.2.24.2 transversal_location

The transversal_location specifies the distance along the Y axis from the origin of the global coordinate system of the Ship.

4.2.24.3 vertical_location

The vertical_location specifies the distance along the Z axis from the origin of the global coordinate system of the Ship.

4.2.25 Change

A Change is a type of Item (see 4.2.115) that represents the focus of all stages associated with a potential or actual change to the product model resulting from a customer or design organization change order. The change may or may not result in modifications to the product model data. Any planned or actual changes to the product model are documented in the associated Change_definition objects.

The data associated with a Change are the following:

— the_class.

4.2.25.1 the_class

The the_class specifies the qualification of the organizational role of the change.

EXAMPLE Headquarter Modification Request or Engineering Change Proposal.

4.2.26 Change_definition

A Change_definition is a type of Definition (see 4.2.56) that is the generalization of the major discrete stages of a Change. A Change_definition is either a Change_request (see 4.2.30), a Change_plan (see 4.2.28), or a Change_realization (see 4.2.29).

The data associated with a Change_definition are the following:

— author;

— date time;

— defined_for;

— local_units.

4.2.26.1 author

The author specifies the person or organization responsible for the Change activities during the period lasting from the end of the previous, if it exists, up to the end of this Change_definition.

4.2.26.2 date_time

The date_time specifies the date and time when the state of the Change_definition was reached

4.2.26.3 defined_for

The defined_for specifies the Change to which the Change_definition is applicable. There may be more than one defined_for for a Change_definition. See 4.3.11 for the application assertion.

4.2.26.4 local units

The local_units specifies a Change_definition is not to define local units.

4.2.27 Change_impact

A Change_impact defines the effect a Change (see 4.2.25) will cause or has caused.

The data associated with a Change_impact are the following:

— impact.

4.2.27.1 impact

The impact specifies the Change_impact of a Change in terms of the creation, modification or deletion of some Definition objects, Item_structure objects, or Item_relationship objects. There may be more than one impact for a Change_impact. See 4.3.12 for the application assertion.

4.2.28 Change_plan

A Change_plan is a type of Change_definition (see 4.2.26) that defines the proposed solution for a Change (see 4.2.25) . A Change_plan is the basis for the activities necessary to implement the Change in the product model.

The data associated with a Change_plan are the following:

— checks;

— chosen_solution_for;

— planned_impact.

4.2.28.1 checks

The checks specifies the verifications defined for the Change_plan. The checks need not be specified for a particular Change_plan. There may be more than one checks for a Change_plan. See 4.3.15 for the application assertion.

4.2.28.2 chosen_solution_for

The chosen_solution_for specifies identification of the Change_request for which a Change_plan is applicable. See 4.3.14 for the application assertion.

4.2.28.3 planned_impact

The planned_impact specifies the estimated or calculated effects of the Change_plan. See 4.3.13 for the application assertion.

NOTE This impact is usually chosen from the set of solution alternatives of a Change_request.

4.2.29 Change_realization

A Change_realization is a type of Change_definition (see 4.2.26) that defines the actual, observed effects of a Change (see 4.2.25).

The data associated with a Change_realization are the following:

— checks;

— impact;

— realization_of.

4.2.29.1 checks

The checks specifies the organizational approval of the product model revisions made to implement the Change_realization. The checks need not be specified for a particular Change_realization. There may be more than one checks for a Change_realization. See 4.3.18 for the application assertion.

4.2.29.2 impact

The impact specifies the identification of the revisions made to the product model under this Change_-realization. See 4.3.16 for the application assertion.

4.2.29.3 realization_of

The realization_of specifies the Change_plan for which a product model Change is being implemented. See 4.3.17 for the application assertion.

4.2.30 Change_request

A Change_request is a type of Change_definition (see 4.2.26) that is the first phase of a Change (see 4.2.25), where the need for a Change and possible solution alternatives are established.

The data associated with a Change_request are the following:

— addressee;

— initiator;

— problem;

— solution_alternatives;

— solution_description.

4.2.30.1 addressee

The addressee specifies the person or organization the Change_request is addressed to. The addressee need not be specified for a particular Change_request.

4.2.30.2 initiator

The initiator specifies the person or organization the Change_request is coming from.

4.2.30.3 problem

The problem specifies a description of the problem that induced the Change_request.

4.2.30.4 solution_alternatives

The solution_alternatives specifies the alternative solutions proposed to solve the problem induced by the Change_request. A solution is described in terms of the effect on Versionable_object objects (see 4.2.263). The solution_alternatives need not be specified for a particular Change_request. There may be more than one solution_alternatives for a Change_request. See 4.3.19 for the application assertion.

4.2.30.5 solution_description

The solution_description specifies a description of one or more possible solutions for the problem. This textual description should be present, if the solution_alternatives are not yet established, or may enhance the information provided by the solution_alternatives. The solution_description need not to be specified for a particular Change_request.

4.2.31 Channel_cross_section

A Channel_cross_section is a type of Channel_profile_cross_section (see 4.2.32) that provides the parameters for a channel's shape.

NOTE ISO 657-11 specifies dimensions for some standard channels.

The data associated with a Channel_cross_section are the following:

— depth;

— flange_thk;

— k;

— radius;

— web_thk;

— width.

4.2.31.1 depth

The depth specifies the depth of the web.

4.2.31.2 flange_thk

The flange_thk specifies the flange thickness.

4.2.31.3 k

The k specifies the distance from outside flange to fillet on web.

4.2.31.4 radius

The radius specifies the radius at the connection of the web and the flange.

4.2.31.5 web_thk

The web_thk specifies the web thickness.

4.2.31.6 width

The width specifies the width of the flange.

4.2.32 Channel_profile_cross_section

A Channel_profile_cross_section is a type of Parametric_profile_cross_section (see 4.2.151) that is identified by a channel's shape, consisting of a web and two flanges. Each Channel_profile_cross_-section is a Channel_cross_section (see 4.2.31).

NOTE ISO 657-11 specifies dimensions for some standard channels.

4.2.33 Check

A Check is a type of Event (see 4.2.79) that defines the details of a planned or fulfilled approval within an organization for a Change_plan (see 4.2.28) or a Change_realization (see 4.2.29).

4.2.34 Circular_cutout_design_definition

A Circular_cutout_design_definition is a type of Interior_cutout_design_definition (see 4.2.111) that is provided with the radius of the circle for the circular cutout.

The data associated with a Circular_cutout_design_definition are the following:

— radius.

4.2.34.1 radius

The radius specifies the radius of the circle that defines the shape of this circular cutout.

NOTE Figure 10 illustrates the radius for a circular cutout.

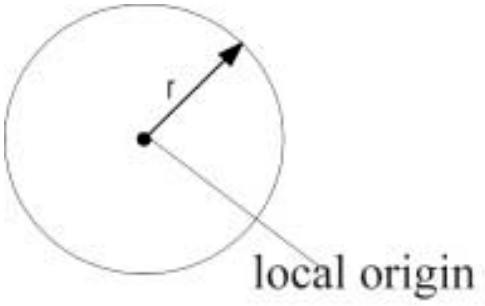


Figure 10 — Circular cutout

4.2.35 Circular_hollow_profile_cross_section

A Circular_hollow_profile_cross_section is a type of Parametric_profile_cross_section (see 4.2.151) that is circular in shape and has a non-solid core.

NOTE ISO 657-14 [5] specifies dimensions for some standard circular hollow sections.

EXAMPLE Pipe used as a structural stanchion would be represented using a Circular_hollow_profile_cross_section.

The data associated with a Circular_hollow_profile_cross_section are the following:

— inner_diameter;

— outer_diameter.

4.2.35.1 inner_diameter

The inner_diameter specifies the maximum dimension across the inner void.

4.2.35.2 outer_diameter

The outer_diameter specifies the maximum dimension between the outer faces.

4.2.36 Class_and_statutory_designation

A Class_and_statutory_designation is a type of General_characteristics_definition (see 4.2.97) that specifies the identification given to the Ship (see 4.2.204) by the classification society for the purpose of design, manufacture and in-service approval.

The data associated with a Class_and_statutory_designation are the following:

— class_number;

— local_units;

— the_class;

— the_statutory.

4.2.36.1 class_number

The class_number specifies the classification society specific identifier to a Ship.

4.2.36.2 local_units

The local_units specifies that the Class_and_statutory_designation is not to define local units.

4.2.36.3 the_class

The the_class specifies the applicable Class_notation with information about the Ship type and the cargo. See 4.3.20 for the application assertion.

4.2.36.4 the_statutory

The the_statutory specifies the set of national and international regulations and standards with which the Ship is intended to comply because of its Class_and_statutory_designation. See 4.3.21 for the application assertion.

4.2.37 Class_bsu

A Class_bsu is a type of Bsu (see 4.2.18) that identifies a class in a parts library.

NOTE 1 bsu is an accronym for “basic semantical unit”.

NOTE 2 The combination of supplier identification, code, and version of a class is to be unique.

The data associated with a Class_bsu are the following:

— absolute_id;

— code;

— delivered_by;

— dic_identifier;

— version.

4.2.37.1 absolute_id

The absolute_id specifies the identifier of the information piece constructed from the code and the dic_-identifier.

4.2.37.2 code

The code specifies the designation of the identificatoin of the class.

4.2.37.3 delivered_by

The delivered_by specifies the library supplier who delivers the library class with a Class_bsu. See 4.3.22 for the application assertion.

4.2.37.4 dic_identifier

The dic_identifier specifies the identifier of the information piece constructed from the code and the version data..

4.2.37.5 version

The version specifies the designation of the version of the information piece.

4.2.38 Class_notation

A Class_notation specifies the notations given to the hull and machinery of the Ship by the classification society as a result of its approval activities during the design, manufacture and in-service maintenance of the Ship.

The data associated with a Class_notation are the following:

— approval_required_for_heavy_cargo;

— approval_required_for_oil_cargo;

— approval_required_loading_unloading_aground;

— approval_required_loading_unloading_grabs;

— class_notations_hull;

— class_notations_machinery;

— class_society;

— ice_class_notation;

— service_area;

— service_factor.

4.2.38.1 approval_required_for_heavy_cargo

The approval_required_for_heavy_cargo specifies the flag indicating whether or not approval for special strengthening for heavy cargoes is necessary. These notations are valid for bulk carriers to indicate the distribution of loads across the cargo holds. The approval_required_for_heavy_cargo need not be specified for a particular Class_notation.

The value of the approval_required_for_heavy_cargo is to be one of the following:

— HC;

— HC_E;

— HC_EA.

NOTE See 4.2.38.1.1 - 4.2.38.1.3 for the definition of each allowable value for approval_required_for_heavy_-cargo.

4.2.38.1.1 hc

strengthened for heavy cargo. Heavy bulk cargo may be unevenly distributed among the cargo holds.

4.2.38.1.2 hc_e

strengthened for heavy cargo. In addition to hc a non-homogeneous loading condition with empty holds on full draught is approved. The approved combination of empty holds is added to the notation.

EXAMPLE Holds 2, 3, and 5 are empty.

4.2.38.1.3 hc_ea

strengthened for heavy cargo. Any cargo hold may be empty at full draught. The approved combinations of empty holds are added to the notation.

EXAMPLE Holds 2, 3, and 5 are empty, or... are empty.

4.2.38.2 approval_required_for_oil_cargo

The approval_required_for_oil_cargo specifies the flag indicating whether or not approval is required for the carriage of oil cargoes.

4.2.38.3 approval_required_loading_unloading_aground

The approval_required_loading_unloading_aground specifies a flag indicating whether or not approval for loading and unloading aground is necessary.

4.2.38.4 approval_required_loading_unloading_grabs

The approval_required_loading_unloading_grabs specifies a flag indicating whether or not approval for loading and unloading using grabs is necessary.

4.2.38.5 class_notations_hull

The class_notations_hull specifies the notation given to the hull of the Ship by the classification society as a result of its approval activities done on the hull. Class_notations_hull may have one or many values contained in it, without any duplications.

4.2.38.6 class_notations_machinery

The class_notations_machinery specifies the notation given to the machinery on the Ship by the classification society as a result of its approval activities done on the machinery. Class_notations_-machinery may have one or many values contained in it, without any duplications.

4.2.38.7 class_society

The class_society specifies the name and organizational details of the classification society whose rules and regulations are being used to assess the Ship.

4.2.38.8 ice_class_notation

The ice_class_notation specifies the type of class notation given to the Ship indicating the ice conditions in which the Ship has been approved to operate. The ice_class_notation need not to be specified for a particular Class_notation.

4.2.38.9 service_area

The service_area specifies the area or route in which the Ship operates.

NOTE This may include information about waterway, wave, weather and wind conditions.

4.2.38.10 service_factor

The service_factor specifies the service area of the Ship and the waves that occur in that area. The service factor should be in the range of 0.5 to 1.0. The service_factor need not to be specified for a particular Class_notation.

4.2.39 Class_parameters

A Class_parameters is a type of General_characteristics_definition (see 4.2.97) that Class_parameters specify the length and speed of the ship in accordance with classification society rules and statutory regulations.

The data associated with a Class_parameters are the following:

— block_coeffcient_class;

— design_speed_ahead;

— design_speed_astern;

— length_class;

— length_solas;

— scantlings_draught.

4.2.39.1 block_coefficient_class

The block_coefficient_class specifies the ratio of the moulded displacement volume to the volume of a block that has its length equal to the length class, its breadth equal to the moulded breadth and its depth equal to the scantlings draught.

4.2.39.2 design_speed_ahead

The design_speed_ahead specifies the forward speed at which the ship is designed to operate.

4.2.39.3 design_speed_astern

The design_speed_astern specifies the reverse speed at which the ship is designed to operate.

NOTE Froude_number : Ratio_measure := design_speed_ahead / (SQRT(9.81 * length_class)); a parameter of significance to the resistance and propulsion calculations of a ship. The Froude number is a nondimensional parameter which is defined by dividing the ship speed in metre per second by the square root of the product of the length of the ship in metre and the gravitational acceleration in metre per square second. In the derive clause the design_speed_ahead for the ship speed and the length_class for the length of the ship are used.

4.2.39.4 length_class

The length_class specifies the length measurement for the ship that is defined in classification society rules.

4.2.39.5 length_solas

The length_solas specifies the length measurement for the ship measured in accordance with the international convention on the safety of life at sea.

4.2.39.6 scantlings_draught

The scantlings_draught specifies the summer load draught used by the classification society in its calculations for structural integrity and strength.

4.2.40 Composite_feature

A Composite_feature is a type of Feature (see 4.2.85) that is built up of other Features.

NOTE 1 A Structural_feature (see 4.2.226) is made a composite feature by instantiating it as a complex subtype of Feature, that is combining itself with this Composite_feature subtype.

NOTE 2 A Composite_feature is intended for two applications:

— to model a complex Feature that is composed of a set of simple Features;

— to model Feature objects that are logically defined on an aggregated level like a panel, but physically implemented on the atomic parts constituting the composite object. A manhole logically defined on a panel can be decomposed into corner-cutouts on the plates realizing the panel. When such decomposition is done, knowledge about the affected connections is required. e.g. plate seems. The decomposition need not necessarily be defined before plating of the panel is frozen. If the plating is changed, the decomposition structure must be maintained.

NOTE 3 Figure 11 shows composite features on plate part

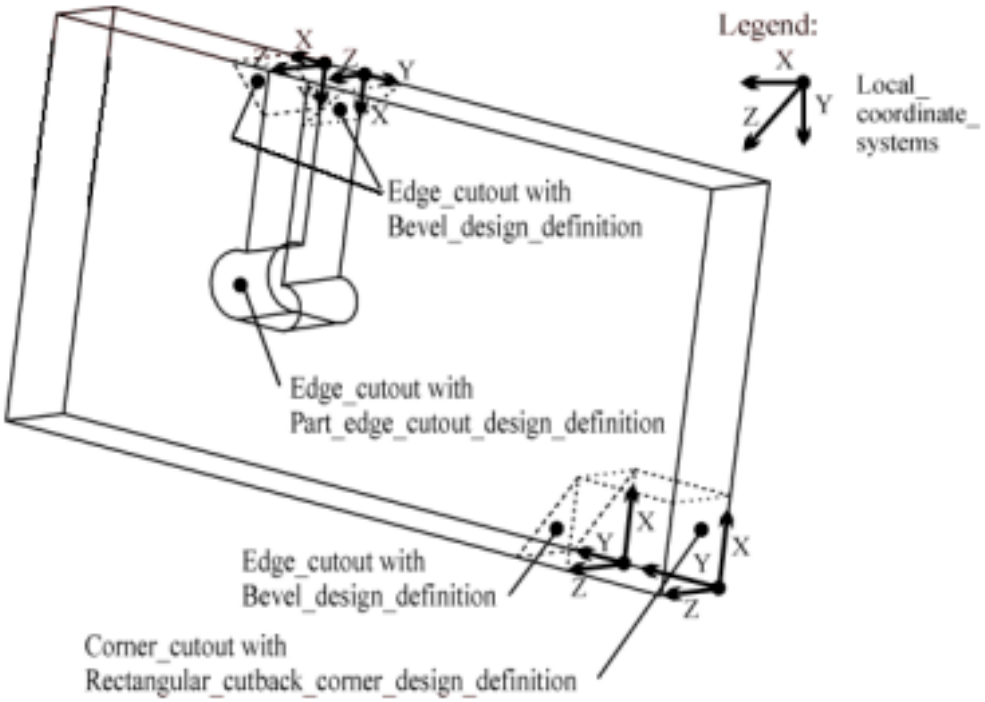


Figure 11 — Composite features on plate part

NOTE 4 Figure 12 shows composite features on profile part

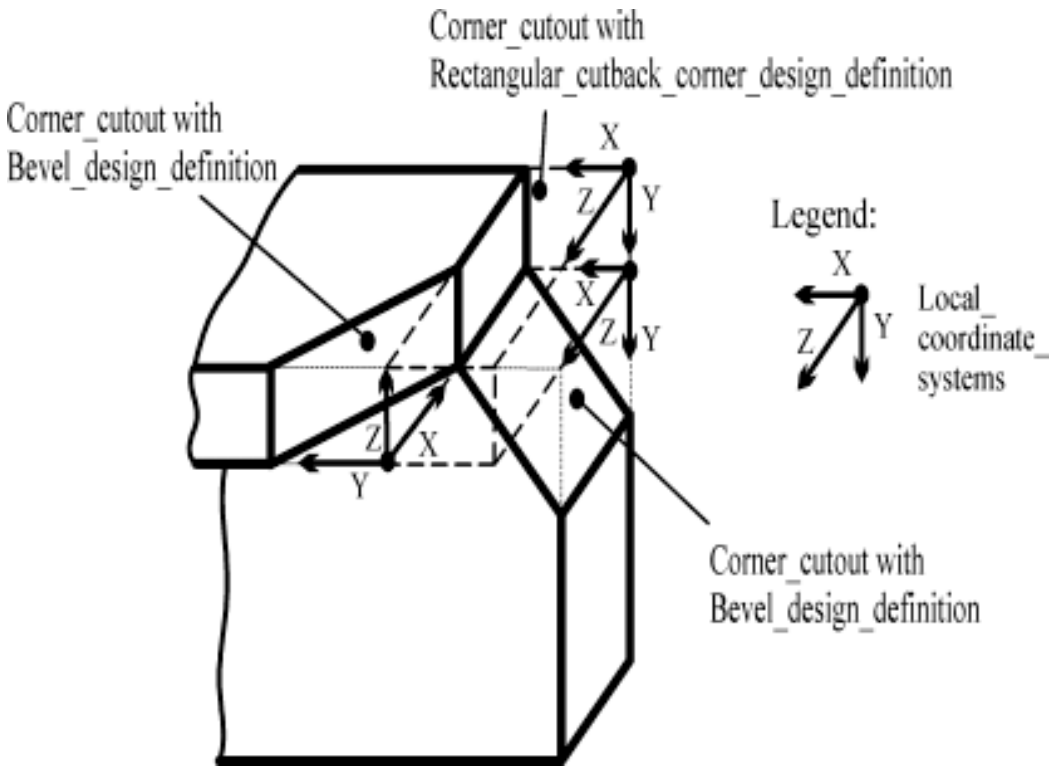


Figure 12 — Composite features on profile part

The data associated with a Composite_feature are the following:

— composed_of.

4.2.40.1 composed_of

The composed_of specifies the Feature objects which this Feature is built up of. There may be more than one composed_of for a Composite_feature. See 4.3.23 for the application assertion.

4.2.41 Continuous_fillet_weld

A Continuous_fillet_weld is a type of Fillet_weld (see 4.2.88) where the weld bead is uninterrupted over the entire length of the weld. Usually, this type of weld is associated with joints requiring a higher degree of tightness.

4.2.42 Corner_cutout

A Corner_cutout is a type of Structural_cutout (see 4.2.223) that is a feature located on the corner of a Structural_part (see 4.2.229). A Corner_cutout specifies removal of material at the corner along at least two edges of a Structural_part.

EXAMPLE Corner_cutout objects may be used to specify a snipe at the end of a flat bar profile or rounded cutouts at the corners of a plate.

NOTE Many types of Corner_cutout objects may be distinguished. These are specified using different corner cutout design definitions, such as:

— Round_corner_design_definition;

— Inward_round_corner_design_definition;

— Outward_round_corner_design_definition;

— Bevel_design_definition;

— Shear_bevel_design_definition;

— Rectangular_cutback_corner_design_definition.

4.2.43 Corner_cutout_boundary_relationship

A Corner_cutout_boundary_relationship is a type of Structural_cutout_boundary_relationship (see 4.2.225) that specifies the topological relationship between a Corner_cutout (see 4.2.42) and the boundary segment of a Plate (see 4.2.155) or Panel_system (see 4.2.144).

The data associated with a Corner_cutout_boundary_relationship are the following:

— boundary_index_2;

— item_1.

4.2.43.1 boundary_index_2

The boundary_index_2 specifies the index of the other boundary in the boundary list of the Panel_-system_design_definition or Plate_design_definition, depending on which one is referenced by item_2, that has the cutout. A corner involves two boundary segments.

4.2.43.2 item_1

The item_1 specifies the relating item of the relationship is a Corner_cutout. See 4.3.24 for the application assertion.

4.2.44 Corner_cutout_design_definition

A Corner_cutout_design_definition is a type of Feature_design_definition (see 4.2.86) that specifies the design definition for a cutout at the corner of a profile, plate, or structural system. Each Corner_cutout_-design_definition is either a Round_corner_design_definition (see 4.2.195), a Bevel_design_definition (see 4.2.16), or a Rectangular_cutback_corner_design_definition (see 4.2.188).

A Corner_cutout_design_definition may specify the design_definition for either a Corner_cutout (see 4.2.42) or an Edge_feature (see 4.2.73). Material is removed along the direction of the Z-axis. If the Z-axis is oriented along the part edge, the Corner_cutout_design_definition specifies an Edge_feature (see 4.3.26 for the application assertion). If the Z-axis is oriented perpendicular to the part edge, the Corner_cutout_design_definition specifies an Corner_cutout (see 4.3.25 for the application assertion).

The data associated with a Corner_cutout_design_definition are the following:

— z_depth.

4.2.44.1 z_depth

The z_depth specifies the extension of the Corner_cutout_design_definition in z-direction. If no z depth is specified, the cutout is along the entire z-direction. The z_depth need not to be specified for a particular Corner_cutout_design_definition.

NOTE The starting-point of a Corner_cutout_design_definition is given by the origin of the Local_-co_ordinate_system that always is assigned to a Structural_cutout.

4.2.45 Corrugated_part

A Corrugated_part is a type of Structural_part (see 4.2.229) that is made from corrugated stock material.

4.2.46 Corrugated_part_approval

A Corrugated_part_approval is a type of Structural_class_approval (see 4.2.219) that confirms the approval of a Corrugated_part_design_definition.

The data associated with a Corrugated_part_approval are the following:

— subject.

4.2.46.1 subject

The subject specifies the approved Corrugated_part_design_definition. See 4.3.27 for the application assertion.

4.2.47 Corrugated_part_design_definition

A Corrugated_part_design_definition is a type of Structural_part_design_definition (see 4.2.231) that describes the definition of a Corrugated_part (see 4.2.45) from the view of detailed design.

NOTE Figure 13 illustrates some parameters associated with the corrugated_part_design_definition.

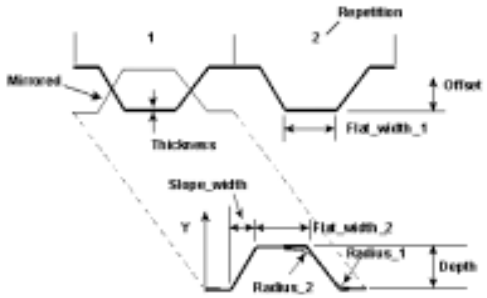


Figure 13 — Corrugated part parameters

The data associated with a Corrugated_part_design_definition are the following:

— border;

— defined_for;

— mirrored;

— offset;

— repetition;

— shape_aspect;

— thickness.

4.2.47.1 border

The border specifies that the Corrugated_part is assumed to be the extrusion of the cross section, made of repetition Corrugation objects, along the local X-axis limited by two boundaries. A boundary can be one of a given surface, or a boundary of a Corrugated_structure (see 4.2.49). The border need not be specified for a particular Corrugated_part_design_definition. There may be more than one border for a Corrugated_part_design_definition. See 4.3.29, 4.3.31 for the application assertion.

4.2.47.2 defined_for

The defined_for specifies the Corrugated_part objects that are defined by the Corrugated_part_-design_definition. There may be more than one defined_for for a Corrugated_part_design_definition. See 4.3.28 for the application assertion.

4.2.47.3 mirrored

The mirrored specifies if the Corrugation is mirrored with respect to its definition.

4.2.47.4 offset

The offset specifies offset information for the location of the Corrugation.

4.2.47.5 repetition

The repetition specifies the number of Corrugations that make this Corrugated_part.

4.2.47.6 shape_aspect

The shape_aspect specifies the shape defining aspect of the 'cross section' of a Corrugated_part. See 4.3.30 for the application assertion.

4.2.47.7 thickness

The thickness specifies the material thickness of the corrugated part.

NOTE If the Corrugated part belongs to a Structural system that got a Design_definition during the preliminary design stage like Corrugated structure thickness should match with the one specified there. For some reason however it might be necessary that a Corrugated part gets a thickness other than specified during preliminary design.

4.2.48 Corrugated_part_rejection

A Corrugated_part_rejection is a type of Structural_class_rejection (see 4.2.222) that confirms the Corrugated_part_design_definition is not approved. The reasons for rejection and existing requirements for the Corrugated_part are specified.

NOTE Corrugated_part rejection may specify additional information clarifying the rule requirements for the Corrugated_part. A yard has several options for action; the thickness of the part may be increased, the material quality may be improved, another stiffening may be applied. Although it is not mandatory, a class society may suggest improvements in these areas. This may then be done using Structural_class_rejection.proposed_-alternative.

The data associated with a Corrugated_part rejection are the following:

— subject.

4.2.48.1 subject

The subject specifies the rejected Corrugated_part_design_definition. See 4.3.32 for the application assertion.

4.2.49 Corrugated_structure

A Corrugated_structure is a type of Structural_system (see 4.2.239) that has the restriction to only consist of Corrugated_part objects (see 4.2.45) and Plate objects (see 4.2.155).

The data associated with a Corrugated_structure are the following:

— the_class.

4.2.49.1 the_class

The the_class specifies the class of a Corrugated_structure.

The value of the_class is one of the following:

— corrugated_structure_class.

NOTE See 4.2.49.1.1 for the definition of each allowable value for the_class.

4.2.49.1.1 corrugated_structure_class

corrugated_structure_class:special section that consists of only Corrugated_part objects and Plate objects.

4.2.50 Corrugated_structure_boundary_relationship

A Corrugated_structure_boundary_relationship is a type of Corrugated_structure_relationship (see 4.2.53) that defines a boundary of a Corrugated_structure (see 4.2.49) by referring to a structural item. Structural item can be one of the following:

— Corrugated_structure - the boundary is created by the intersection of the implicitly or explicitly given moulded surfaces of the two Corrugated_structure objects;

— Panel system - the boundary is created by the intersection of the implicitly or explicitly given moulded surfaces of the Corrugated_structure and the Panel systems;

— Seam - the Seam is the boundary.

The data associated with a Corrugated_structure_boundary_relationship are the following:

— displacement.

4.2.50.1 displacement

The displacement specifies the offset orthogonally off the referenced boundary. The displacement need not be specified for a particular Corrugated_structure_bounday_relationship. There may be more than one displacement for a Corrugated_structure_bounday_relationship.

4.2.51 Corrugated_structure_design_definition

The Corrugated_structure_design_definition is a type of Structural_system_design_definition (see 4.2.241) that describes the definition of a Corrugated_structure (see 4.2.49) for the life-cycle stage preliminary design.

NOTE Figure 14 shows parameters for corrugated structures.

The data associated with a Corrugated_structure_design_definition are the following:

— border;

— corrugation_width;

— defined_for;

— moulded_surface;

— thickness.

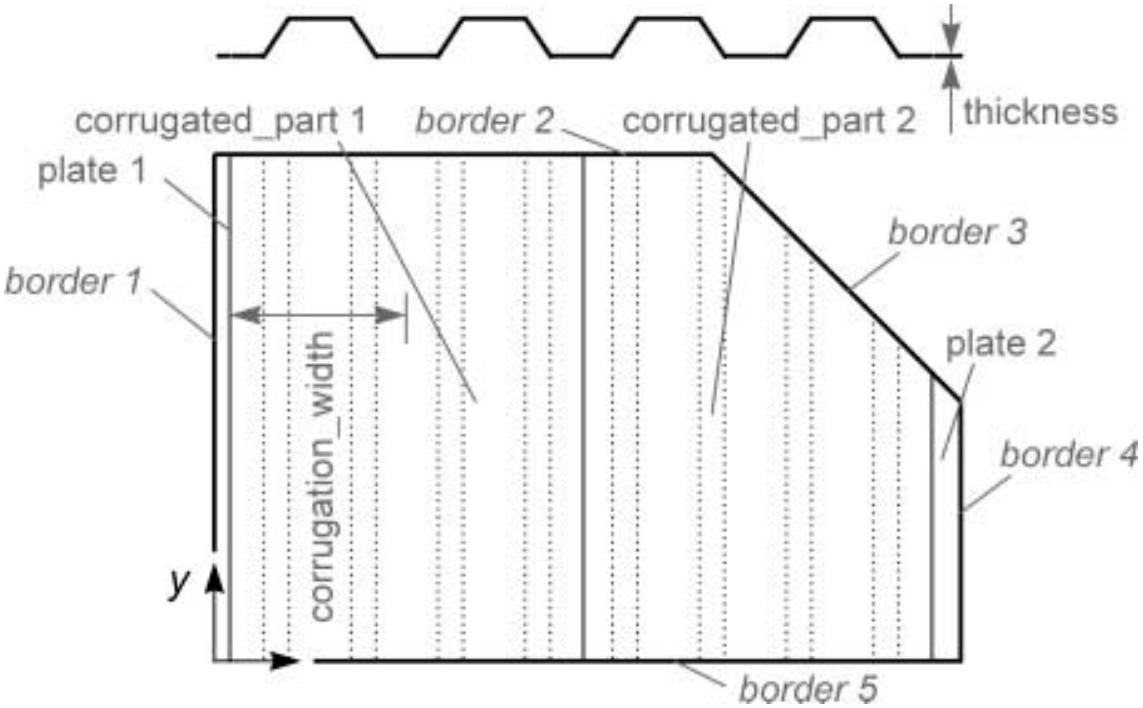


Figure 14 — Corrugated system parameters

4.2.51.1 border

The border specifies the border of a Corrugated_structure is made of a number of boundaries. The order
of the boundaries is significant and therefore should be preserved by the mapping. However there is no rule that requires end point of boundary[n] = start point of boundary[n+1]. Furthermore two boundaries may intersect or it might be necessary to extend one or both boundaries in order to make them intersecting.

A boundary can be one of the following;

— a line resulting from the intersection of the Corrugated_structure's moulded surface with the moulded surface of a given Structural_system;

— a curve parallel to a boundary of a given Structural system with a given displacement;

— a Seam.

4.2.51.2 corrugation width

The corrugation_width specifies the proposed length of the corrugation of the corrugated parts belonging to the corrugated structure. The corrugation_width need not be specified for a particular Corrugated_structure_design_definition.

4.2.51.3 defined_for

The defined_for specifies the Corrugated_structure objects that are defined by the Corrugated_-structure_design_definition. See 4.3.33 for the application assertion.

4.2.51.4 moulded_surface

The moulded_surface specifies the underlying moulded surface of the Corrugated_structure. If not specified, the local x-y-plane is implicitly assumed as underlying moulded surface. A moulded surface can either be a surface, a Non_manifold_surface_shape representation, or an External_instance_reference to a moulded form specified by ISO10303-216. The moulded surface need not to be specified for a particular Corrugated_structure_design_definition. See 4.3.34, 4.3.35 for the application assertion.

4.2.51.5 thickness

The thickness specifies the proposed material thickness of the Corrugated_structure.

NOTE The thickness specified here serves as a reference value for the thickness of the Corrugated_part objects and Plate objects defined for the Corrugated_structure referred to in a later stage of the design. For some reason it may happen that the thickness of the Corrugated_part objects and Plate objects differs from the one specified here.

4.2.52 Corrugated_structure_functional_definition

A Corrugated_structure_functional_definition is a type of Structural_system_functional_definition (see 4.2.242) that defines the function of a Corrugated_structure (see 4.2.49).

The data associated with a Corrugated_structure_functional_definition are the following:

— defined_for.

4.2.52.1 defined_for

The defined_for specifies the Corrugated_structure objects that are defined by the Corrugated_-structure_functional_definition. There may be more than one defined_for for a Corrugated_-structure_functional_definition. See 4.3.36 for the application assertion.

4.2.53 Corrugated_structure_relationship

A Corrugated_structure_relationship is a type of Structural_system_relationship (see 4.2.244) that describes the association of a Corrugated_structure (see 4.2.49) with another structural item. Structural item can be one of the following:

— Structural_system;

— Structural_part;

— Structural_feature.

Each Corrugated_structure_relationship is a Corrugated_structure_boundary_relationship (see 4.2.50). The data associated with a Corrugated_structure_relationship are the following:

— item_1.

4.2.53.1 item_1

The item_1 specifies a Corrugated_structure_relationship relates a Corrugated_structure with something else. See 4.3.37 for the application assertion.

NOTE Figure 15 shows a corrugated bulkhead.

4.2.54 Corrugation

A Corrugation is the shape defining aspect of the cross section of a Corrugated_part (see 4.2.45).

NOTE 1 The bottom half of Figure 15 is an example of a corrugated bulkhead.

NOTE 2 See Figure 13 for the illustration of the parameters for the corrugation.

The data associated with a Corrugation are the following:

— depth;

— flat_width_1;

— flat_width_2;

— radius_1;

— radius_2;

— slope_width.

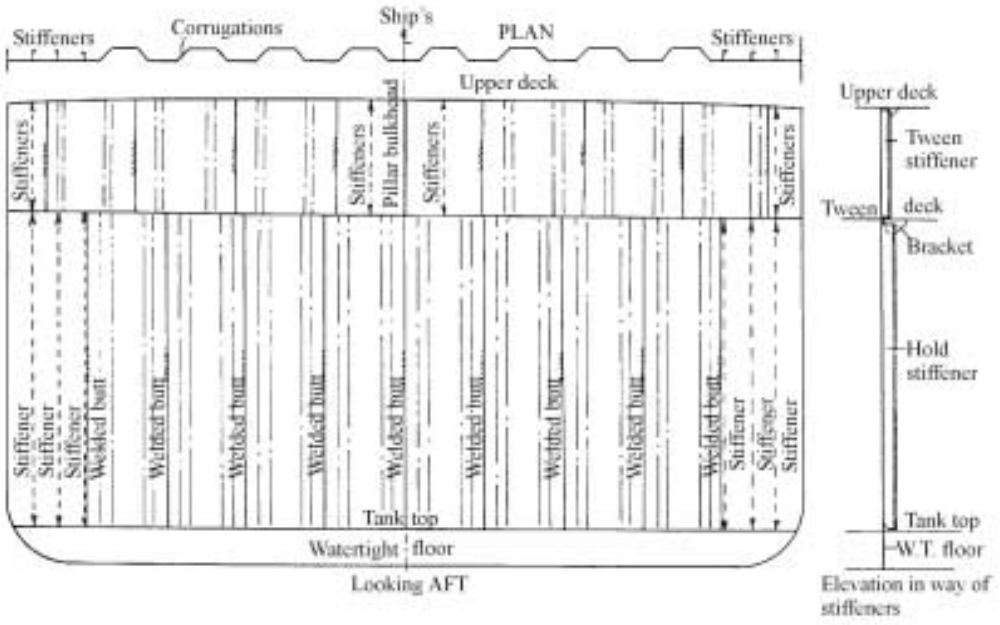


Figure 15 — Corrugated bulkhead

4.2.54.1 depth

The depth specifies the distance between the lower and the upper flat part of the corrugation shape.

4.2.54.2 flat_width_1

The flat_width_1 specifies the width of the lower flat part of the corrugation shape.

4.2.54.3 flat_width_2

The flat_width_2 specifies the width of the upper flat part of the corrugation shape.

4.2.54.4 radius_1

The radius_1 specifies the radius of the bend between the lower flat part and the slope part of the corrugation. The radius_1 need not to be specified for a particular Corrugation.

4.2.54.5 radius_2

The radius_2 specifies the radius of the bend between the upper flat part and the slope part of the corrugation. The radius_2 need not to be specified for a particular Corrugation.

4.2.54.6 slope_width

The slope_width specifies the width of the slope part of the corrugation shape.

4.2.55 Definable_object

A Definable_object is any type of product that can be defined. Each Definable_object is either an Item (see 4.2.115), an Item_relationship (see 4.2.116), or an Item_structure (see 4.2.117).

NOTE Being "defined" means - can be pointed to by a Definition (see 4.2.56).

The data associated with a Definable_object are the following:

— id.

4.2.55.1 id

The id specifies the global unique identifier for the Definable_object. See 4.3.38 for the application assertion.

4.2.56 Definition

A Definition is a type of Versionable_object (see 4.2.263) that is the basis for all types of Definable_objects (see 4.2.55) definitions. Each Definition is either a Change_definition (see 4.2.26), a Design_definition (see 4.2.58), a Functional_definition (see 4.2.96), a General_-characteristics_definition (see 4.2.97), a Library_definition (see 4.2.118), a Lightship_definition (see 4.2.123), a Local_co_ordinate_system (see 4.2.126), a Manufacturing_definition (see 4.2.131), a Plate_renewal_definition (see 4.2.163), a Profile_renewal_definition (see 4.2.181), a Ship_material_-property (see 4.2.206), or a Spacing_table (see 4.2.211). Definitions support the following concepts in shipbuilding design, function, manufacturing, general ship characteristics, design requirements, and parametric and library descriptions of objects.

The data associated with a Definition are the following:

— defined_for;

— id;

— local_units.

4.2.56.1 defined_for

The defined_for specifies the Definable_object objects that are defined by Definition.

4.2.56.2 id

The id specifies the global unique identifier for Definition. See 4.3.39 for the application assertion.

4.2.56.3 local_units

The local_units specifies the units that the Definition makes use of if they differ from the units globally defined for the Ship. Local_units may be either a Derived_unit (see 4.2.57) or a Named_unit (see 4.2.137). Local_units may have no value contained in it or it may have one or more values, without any duplications.

4.2.57 Derived_unit

A Derived_unit is a unit of measure that is composed of elements that are pre-defined Named_unit objects with exponents.

4.2.58 Design_definition

A Design_definition is a type of Definition (see 4.2.56) that is the basis for all types of design definitions. The ability to reference representations differentiates a Design_definition from a Definition. Each Design_definition is either a Feature_design_definition (see 4.2.86), a Hull_cross_-section_design_definition (see 4.2.106), Library_design_definition (see 4.2.119), a Structural_part_-design_definition (see 4.2.231), Structural_part_joint_design_definition (see 4.2.234), a System_-design_definition (see 4.2.250), or a Weld_design_definition (see 4.2.271).

The data associated with a Design_definition are the following:

— representations.

4.2.58.1 representations

The representations specify the representations of the design definition. It is possible for a Design_-definition to have multiple representations. Representations may have no value contained in it or it may have one or more values, without any duplications.

EXAMPLE A Design_definition may have multiple shape representations like wireframe, surface, or solid.

4.2.59 Design_load

A Design_load is a type of General_characteristics_definition (see 4.2.97) that is a load applied to the structure of a ship to prove its strength. The Design_load may be induced by an artificial load, such as a load required by a classification society, or an actual load, such as cargo. Design_load objects

maybe expressed as moments or forces. Each Design_load may be one of the following: a Bending_-moment (see 4.2.14), or a Shear_force (see 4.2.203).

The data associated with a Design_load are the following:

— defined_for.

4.2.59.1 defined_for

The defined_for specifies the Ship objects that a Design_load may be defined for. There may be more than one defined_for for a Design_load. See 4.3.40 for the application assertion.

4.2.60 Design_still_water_bending_moment

A Design_still_water_bending_moment is a type of Bending_moment (see 4.2.14) that is the bending moment amidship that the ship is designed for in still water. The values may be based on actual cargo, ballast conditions, and sea-going conditions. If these are not known, standard moments given by classification rules is applied.

NOTE Figure 16 illustrates Design_still_water_bending_moment.

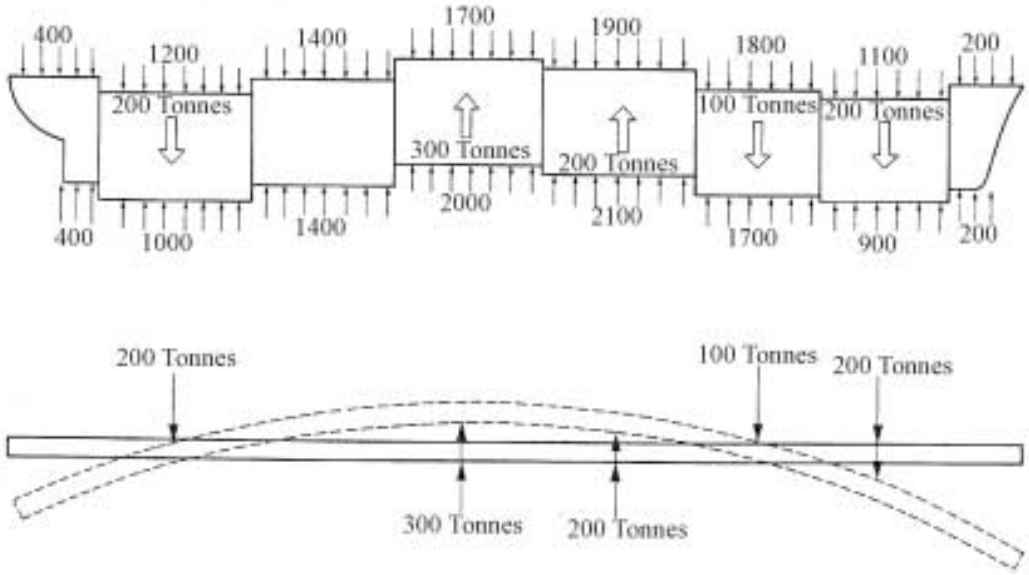


Figure 16 — Design_still_water_bending_moment

The data associated with a Design_still_water_bending_moment are the following:

— hogging_amidship;

— sagging_amidship.

4.2.60.1 hogging_amidship

The hogging_amidship specifies the hogging moment amidship.

4.2.60.2 sagging_amidship

The sagging_amidship specifies the sagging moment amidship.

4.2.61 Design_still_water_shear_force

A Design_still_water_shear_force is a type of Shear_force (see 4.2.203) that is a collection of shear force values distributed along the ship. These represent the shear force load that the ship is designed for when being in sea-going condition.

The data associated with a Design_still_water_shear_force are the following:

— table_of_shear_force_values.

4.2.61.1 table_of_shear_force_values

The table_of_shear_force_values specifies the set of values that defines the shear force distribution along the ship for design purposes. There may be more than one table_of_shear_force_values for a Design_still_water_shear_force. See 4.3.41 for the application assertion.

4.2.62 Design_swsf_values

The Design_swsf_values are design values for shear forces at specified positions along the ship. The values represent classification society requirements.

The data associated with a Design_swsf_values are the following:

— negative_value;

— position;

— positive_value.

4.2.62.1 negative_value

The negative_value specifies the negative shear force value that contributes to overall shear.

4.2.62.2 position

The position specifies distance from the aft-perpendicular from where the shear forces are taken, given as a frame or station position. See 4.3.42 for the application assertion.

4.2.62.3 positive_value

The positive_value specifies the positive shear force value that contributes to overall shear.

4.2.63 Design_vertical_wave_bending_moment

A Design_vertical_wave_bending_moment is a type of Bending_moment (see 4.2.14) that is the bending moment amidship that the ship is designed for being in sea-going conditions. Both hogging and sagging moments have distributions over the length of the ship. These can be derived from the values amidships and the length given by the classification society rules. The values will change for short voyages and sheltered water conditions. These can be derived from the sea-going conditions and so are not necessary for a model used for exchange purposes.

The data associated with a Design_vertical_wave_bending_moment are the following:

— hogging;

— sagging.

4.2.63.1 hogging

The hogging specifies the hogging moment amidships.

4.2.63.2 sagging

The sagging specifies the sagging moment amidships.

4.2.64 Design_vertical_wave_shear_force

A Design_vertical_wave_shear_force is a type of Shear_force (see 4.2.203) that is the shear force amidship that the ship is designed for being in sea-going conditions. Both positive and negative values have distributions over the length of the ship. These can be derived from the values amidships and the length given by the classification society rules. The values will change for short voyages and sheltered water conditions. These can be derived from the sea-going conditions and so are not necessary for a model used for exchange purposes.

The data associated with a Design_vertical_wave_shear_force are the following:

— negative_value;

— positive_value.

4.2.64.1 negative_value

The negative_value specifies the negative shear force value amidships.

4.2.64.2 positive_value

The positive_value specifies the positive shear force value amidships.

4.2.65 Document

A Document is a type of Versionable_object (see 4.2.263) that references an unambiguous identification of some human readable data item defined outside ISO 10303. A document has an author and may be versioned.

The data associated with a Document are the following:

— author;

— source_type;

— summary;

— title.

4.2.65.1 author

The author specifies the person and organization, or organization that authored the Document.

4.2.65.2 source_type

The source_type specifies the format of the document.

EXAMPLE The Document may be in a printed copy of a book or in a file format.

4.2.65.3 summary

The summary specifies the summary or abstract that describes the content of the Document. The summary need not to be specified for a particular Document.

4.2.65.4 title

The title specifies a description of the subject matter within the Document.

4.2.66 Document_portion

A Document_portion is the qualification of a specific subset of a Document in generic terms.

EXAMPLE A subsection of a report could be a range of pages identified by their numbers. Document_-portion.element_type page and Document_portion.element_value 1-10, 15.

The data associated with a Document_portion are the following:

— element_type;

— element_value;

— source.

4.2.66.1 element_type

The element_type specifies the name for this subset of the Document.

EXAMPLE Page, section, subsection, paragraph.

4.2.66.2 element_value

The element_value specifies the value for this subset of the Document.

EXAMPLE Page numbers = “1-10, 15", section numbers = “3.2.4", section title = “Introduction”.

4.2.66.3 source

The source specifies the Document to which the specified subset is related. See 4.3.43 for the application assertion.

4.2.67 Document_reference

A Document_reference is the qualification of a Document (see 4.2.65) or sections of a Document in terms of its source or location. Each Document_reference may be a Document_reference_with_-address (see 4.2.68).

EXAMPLE If the Document_reference source is a book, the pointer could be a section label or a page number.

The data associated with a Document_reference are the following:

— assigned_document.

4.2.67.1 assigned_document

The assigned_document specifies the Document, or Document_portion, that is to be associated with the product data. See 4.3.44, 4.3.45, 4.3.45 for the application assertion.

4.2.68 Document_reference_with_address

A Document_reference_with_address is a type of Document_reference (see 4.2.67) and External_-reference (see 4.2.83) that specifies a pointer to a location inside the source.

4.2.69 Drain_hole_cutout_design_definition

A Drain_hole_cutout_design_definition is a type of Feature_design_definition (see 4.2.86). A drain hole is a symmetric oblong opening with circular ends on both sides and a break-through along the symmetry line to the exterior of the work item. The origin of the local co-ordinate system is placed at the intersection of the work item boundary line and the symmetry axis. All length measures (width, depth, drain hole radius, gap and gap radius) are always positive. Depth is measured along the Y-axis (symmetry axis) whereas width and gap are measured along the X-axis. The Y-axis points into the interior of the work item.

NOTE 1 Figure 17 illustrates a Drain_hole_cutout_design_definition.

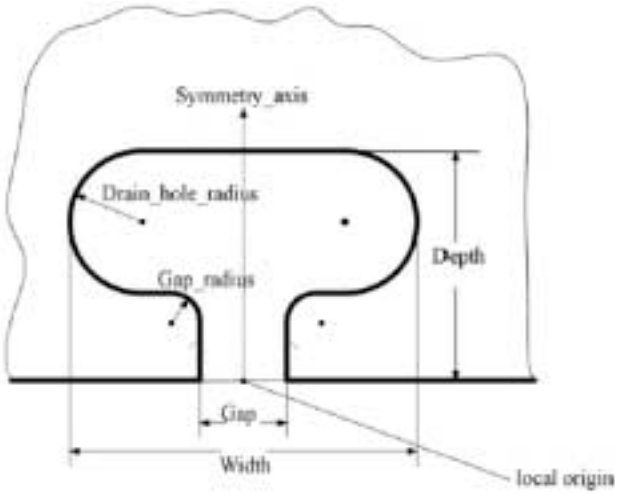


Figure 17 — Drain_hole_cutout

The design of the drain hole is defined parametrically with the following constraints

Depth > 2*drain_hole_readius

Width/2 > gap/2 + gap_radius + drain_hole_radius

The resulting shape is implicitly defined by a connected set of lines and circular arcs that is tangent continuous. The lines and arcs are connected at the following set of points along the positive X-axis (a symmetric set exists along the negative X-axis).

Point	X Coordinate	Y Coordinate
P1	gap/2	0
P2	gap/2	depth – 2*drain_hole_radius – gap_radius
P3	gap/2 + gap_radius,	depth – 2*drain_hole_radius
P4	width/2 – drain_hole_radius	depth – 2*drain_hole_radius
P5	width/2 – drain_hole_radius	Depth
P6	0	Depth

The set of lines and arcs along the positive X-axis is defined as follows:

Start Point	End Point	Geometry
P1	P2	Line
P2	P3	Quarter circle, radius of Gap_radius
P3	P4	Line
P4	P5	Half circle, radius drain_hole_radius
P5	P6	Line

NOTE 2 Figure 18 illustrates a drain_hole_cutout shape as a connected set of lines and arcs.

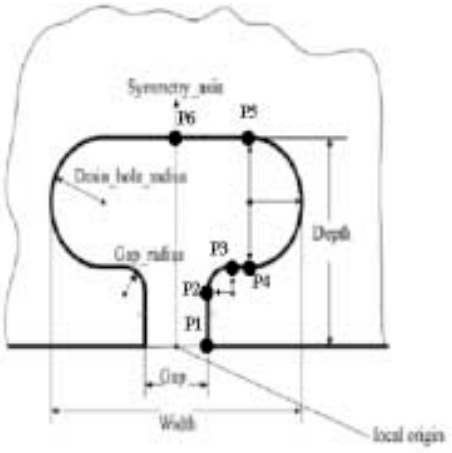


Figure 18 — Drain_hole_cutout shape

The data associated with a Drain_hole_cutout_design_definition are the following:

— defined_for;

— depth;

— drain_hole_radius;

— gap;

— gap_radius;

— width.

4.2.69.1 defined_for

The defined_for specifies the Edge_cutouts that a Drain_hole_cutout_design_definition is valid. There may be more than one defined_for for a Drain_hole_cutout_design_definition. See 4.3.46 for the application assertion.

4.2.69.2 depth

The depth specifies the distance measured from the exterior of the work item through the gap to the furtherest side of the drain hole.

4.2.69.3 drain_hole_radius

The drain_hole_radius specifies the radius of a half circle piece that determines the round shape of the drain hole. The circle is centered at (width/2 – drain_hole_radius, depth – drain_hole_radius).

4.2.69.4 gap

The gap specifies the width of the gap.

4.2.69.5 gap_radius

The gap_radius specifies the radius of the round corner that represents the transition from the drain hole to the gap. The circle is centered at (gap/2 + gap_radius, depth – 2*drain_hole_radius – gap_radius).

4.2.69.6 width

The width specifies the maximum diameter of the drain hole.

4.2.70 Edge_based_wireframe_shape

An Edge_based_wireframe_shape is a shape representation which conforms to ISO 10303-501.

4.2.71 Edge_cutout

An Edge_cutout is a type of Structural_cutout (see 4.2.223) that is a feature that interrupts a more or less straight edge of a Structural_part (see 4.2.229). An Edge_cutout specifies the removal of material for a certain purpose, such as for the penetration of another Structural part or for water escapes. Many types of Edge_cutout objects may be distinguished. These are specified using different edge cutout design definitions, such as:

— Part_edge_cutout_design_definition;

— Drain_hole_cutout_design_definition.

NOTE See 4.2.152 for definition of Part_edge_cutout_design_definition and 4.2.68 for Drain_hole_cutout_-design_definition

4.2.72 Edge_cutout_functional_definition

An Edge_cutout_functional_definition is a type of Functional_definition (see 4.2.96) that defines the function that an Edge_cutout (see 4.2.71) may be designed for.

The data associated with an Edge_cutout_functional_definition are the following:

— defined_for;

— the_function.

4.2.72.1 defined_for

The defined_for specifies the Edge_cutout and Edge_feature that this function is assigned to. There may be more than one defined_for for an Edge_cutout_functional_definition. See 4.3.47 for the application assertion.

4.2.72.2 the_function

The the_function specifies the purpose of the Edge_cutout in the ship structure. The value of the_function is one of the following:

— access_hole;

— air_escape;

— air_liquid_escape;

— liquid_escape;

— penetration;

— user_defined.

NOTE See 4.2.72.2.1 - 4.2.72.2.6 for the definition of each allowable value for the_function.

4.2.72.2.1 access_hole

a cutout that permits entry to an area from another area.

4.2.72.2.2 air_escape

a cutout that permits air to be dispelled or removed.

4.2.72.2.3 air_liquid_escape

a cutout that permits air or liquid to be released, dispelled, or removed.

4.2.72.2.4 liquid_escape

a cutout that permits liquid to be released, dispelled, or removed.

4.2.72.2.5 penetration

a cutout that permits something to pass through.

EXAMPLE A cutout for a stiffener or a pipe that passes through a bulkhead.

4.2.72.2.6 user_defined

the Edge_cutout is defined within the Functional_definition.user_def_function attribute, by the user.

NOTE The user_defined option may only be used when the Edge_cutout cannot be defined by the other function codes.

4.2.73 Edge_feature

An Edge_feature is a type of Structural_cutout (see 4.2.223) that is a feature along an edge of a Structural_part (see 4.2.229).

EXAMPLE An Edge_featur objects may specify the removal of material along an entire edge or parts of it for the purpose of weld preparation, e.g. the beveled edge of a plate or the beveled web of a profile.

NOTE Many types of Edge_features may be distinguished. These can be specified using different Corner_-cutout_design_definition objects, such as:

— Round_corner_design_definition;

— Inward_round_corner_design_definition;

— Outward_round_corner_design_definition;

— Bevel_design_definition;

— Shear_bevel_design_definition;

— Rectangular_cutback_corner_design_definition.

4.2.74 Edge_feature_functional_definition

An Edge_feature_functional_definition is a type of Functional_definition (see 4.2.96) that defines the function that an Edge_feature may be designed for.

The data associated with an Edge_feature_functional_definition are the following:

— defined_for;

— the_function.

4.2.74.1 defined_for

The defined for specifies that the Edge_feature that this function is assigned to. See 4.3.48 for the application assertion.

4.2.74.2 the_function

The the_function specifies the purpose for the Edge_feature in the ship structure.

The value of the_function is one of the following:

— bevel;

— chamfer;

— user_defined.

NOTE See 4.2.74.2.1 - 4.2.74.2.3 for the definition of each allowable value for the_function.

4.2.74.2.1 bevel

the result of removal of material for the purpose of weld preparation.

4.2.74.2.2 chamfer

the result of removal of material for the purpose of thickness transition between plates.

4.2.74.2.3 user_defined

the Edge_feature is defined within the Functional_definition.user_def_function attribute, by the user.

NOTE The user_defined option may only be used when the Edge_feature cannot be defined by the other function codes.

4.2.75 Electrode_chemical_composition

An Electrode_chemical_composition specifies the chemical composition of the wire electrode. All chemical elements within an electrode wire, except iron, are listed as a percentage.

The data associated with an Electrode_chemical_composition are the following:

— aluminum;

— carbon;

— chromium;

— copper;

— manganese;

— molybdenum;

— nickel;

— phosphorus;

— silicon;

— sulphur;

— titanium;

— vanadium;

— zirconium.

4.2.75.1 aluminum

The aluminum specifies the percentage of the chemical element aluminum (Al).

4.2.75.2 carbon

The carbon specifies the percentage of the chemical element carbon (C).

4.2.75.3 chromium

The chromium specifies the percentage of the chemical element chromium (Cr).

4.2.75.4 copper

The copper specifies the percentage of the chemical element copper (Cu).

4.2.75.5 manganese

The manganese specifies the percentage of the chemical element manganese (Mn).

4.2.75.6 molybdenum

The molybdenum specifies the percentage of the chemical element molybdenum (Mo).

4.2.75.7 nickel

The nickel specifies the percentage of the chemical element nickel (Ni).

4.2.75.8 phosphorus

The phosphorus specifies the percentage of the chemical element phosphorus (P).

4.2.75.9 silicon

The silicon specifies the percentage of the chemical element silicon (Si).

4.2.75.10 sulphur

The sulphur specifies the percentage of the chemical element sulphur (S).

4.2.75.11 titanium

The titanium specifies the percentage of the chemical element titanium (Ti).

4.2.75.12 vanadium

The vanadium specifies the percentage of the chemical element vanadium (V).

4.2.75.13 zirconium

The zirconium specifies the percentage of the chemical element zirconium (Zr).

4.2.76 Elliptical_cutout_design_definition

An Elliptical_cutout_design_definition is a type of Interior_cutout_design_definition (see 4.2.111) that is given by the formula $(x/a)^2 + (y/b)^2 = 1$, with a and b being the major and minor semi-axes of the elliptical opening. x is the abscissa and y is the ordinate of any point on the elliptical arc of the opening.

NOTE Figure 19 illustrates an Elliptical_cutout.

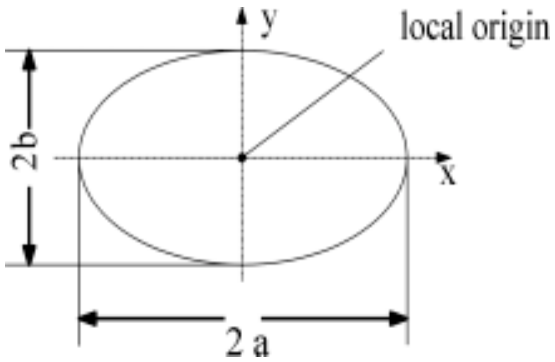


Figure 19 — Eliptical_cutout

The data associated with an Elliptical_cutout_design_definition are the following:

— half_axis_a;

— half_axis_b.

4.2.76.1 half_axis_a

The half_axis_a specifies major semi-axis of the elliptical_cutout_design, which is “a” in the X-axis.

4.2.76.2 half_axis_b

The half_axis_b specifies minor semi-axis of the elliptical_cutout_design, which is “b” in the Y-axis.

4.2.77 Elongated_oval_cutout_design_definition

An Elongated_oval_cutout_design_definition is a type of Interior_cutout_design_definition (see 4.2.111) that is an opening either egg-shaped or shaped like a slot.

NOTE Figure 20 illustrates an Elongated_oval_cutout.

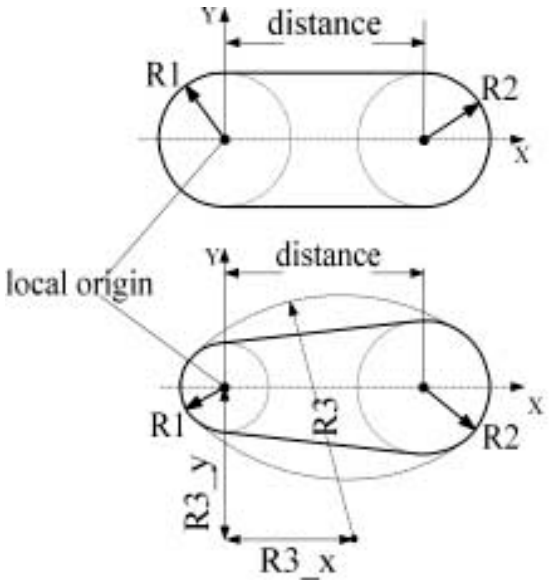


Figure 20 — Elongated_oval_cutout

The data associated with an Elongated_oval_cutout_design_definition are the following:

— distance;

— r1;

— r2;

— r3;

— r3_x;

— r3_y.

4.2.77.1 distance

The distance specifies distance from circle centre with radius r1 at origin to circle centre with radius r2 in direction of positive X-axis.

4.2.77.2 r1

The r1 specifies the radius of circle arc with centre at the origin.

4.2.77.3 r2

The r2 specifies the radius of circle arc to the right along X-axis of circle given by r1.

4.2.77.4 r3

The r3 specifies the radius of the "sides" of the egg shaped opening. The r3 need not to be specified for a particular Elongated_oval_cutout_design_definition.

4.2.77.5 r3_x

The r3_x specifies the offset in x-direction of centre of circle with radius r2. The r3_x need not to be specified for a particular Elongated_oval_cutout_design_definition.

4.2.77.6 r3_y

The r3_y specifies the offset in y-direction of centre of circle with radius r2. The r3_y need not to be specified for a particular Elongated_oval_cutout_design_definition.

4.2.78 Envisaged_version_creation

An Envisaged_version_creation is a type of Versionable_object_change_event (see 4.2.264) that is the event leading to a new Versionable_object. The event is an envisaged event and has not yet happened. The Definition (see 4.2.56), Item_structure (see 4.2.117) or Item_relationship (see 4.2.116) as the subject of the Event does not yet exist and is described in terms of descriptive, non-formal properties.

The data associated with an Envisaged_version_creation are the following:

— base;

— category.

4.2.78.1 base

The base specifies the Versionable_object objects the envisaged new version is derived from. The base need not be specified for a particular Envisaged_version_creation. There may be more than one base for an Envisaged_version_creation. See 4.3.49 for the application assertion.

4.2.78.2 category

The category specifies the category the envisaged Versionable_object belongs to.

4.2.79 Event

An Event is identification that something has happened at a certain time, activated by a certain person or organization for a certain reason. Each Event may be one of the following: an Approval_event (see 4.2.6), a Check (see 4.2.33), or a Versionable_object_change_event (see 4.2.111).

The data associated with an Event are the following:

— caused_by;

— caused_when;

— description.

4.2.79.1 caused_by

The caused_by specifies the person or organization creating the Event.

4.2.79.2 caused_when

The caused_when specifies the date and time that the Event occurred.

4.2.79.3 description

The description specifies a description for the reason of the Event.

4.2.80 Explicit_feature_design_definition

An Explicit_feature_design_definition is a type of Feature_design_definition (see 4.2.86) that describes the design of a Feature (see 4.2.85) by the explicit representation of its shape. The shape is given as a geometric model without parameters and without a reference to moulded geometry. This Feature_design_definition is best suited for the purely visual presentation of the Feature. The functionality of the specific Feature is not included in this description.

The data associated with an Explicit_feature_design_definition are the following:

— representations.

4.2.80.1 representations

The representations specifies the geometric shape of the Explicit_feature_design_definition. Only one shape representation is in the set of representations. The representations need not be specified for a particular Explicit_feature_design_definition. There may be more than one representations for an Explicit_feature_design_definition. See 4.3.50, 4.3.51, 4.3.52, 4.3.53 for the application assertions. The valid shape representations are the following:

— Advanced_brep_shape (see 4.2.1);

— Edge_based_wireframe_shape (see 4.2.70);

— Geometrically_bounded_wireframe_shape (see 4.2.98);

— Non_manifold_surface_shape (see 4.2.140).

4.2.81 Explicit_profile_cross_section

An Explicit_profile_cross_section is a type of Profile_cross_section (see 4.2.174) that is the explicit geometric definition of the cross-section of a rolled profile.

The data associated with an Explicit_profile_cross_section are the following:

— cross_section_geometry;

— local_coordinate_system.

4.2.81.1 cross_section_geometry

The cross_section_geometry specifies the closed planar curve that defines the outer boundary of the profile.

4.2.81.2 local_coordinate_system

The local_coordinate_system specifies the point and pair of axes on the cross section geometry that specifies the point and orientation that is used to place the explicit cross-section geometry onto the extrusion trace.

4.2.82 External_instance_reference

An External_instance_reference represents an instance of an object that does not exist in the same scope.

NOTE The object who's instance is represented must be a subtype of either Definable_object or Definition in order to be referable via a global unique identifier.

The data associated with an External_instance_reference are the following:

— entity_type;

— schema_name;

— target_guid.

4.2.82.1 entity_type

The entity_type specifies the name of the type of the externally referenced instance.

4.2.82.2 schema_name

The schema_name specifies the schema in which the externally referenced instance is defined.

4.2.82.3 target_guid

The target_guid specifies the global unique identifier of the externally referenced instance. See 4.3.54 for the application assertion.

4.2.83 External_reference

An External_reference is the abstract denotation of a data source external to the data set where an instance of this object exists. Each External_reference may be a Document_reference_with_address (see 4.2.68).

EXAMPLE An Universal_resource_locator denotes such a data source.

The data associated with an External_reference are the following:

— description;

— location.

4.2.83.1 description

The description specifies the External_reference..

4.2.83.2 location

The location specifies the location of an external reference. In the case of an Universal_resource_-locator, the location is computer accessible by a specified transmission protocol. The location need not be specified for a particular External_reference. See 4.3.55 for the application assertion.

4.2.84 External_storage

An External_storage describes the location of physical documents or items external to the current exchange. This object may be relevant, when identifying the location of CD-ROMs, floppy disks, or video tapes.

EXAMPLE A libary or a company archive may serve as External_storage.

The data associated with an External_storage are the following:

— location.

4.2.84.1 location

The location specifies the identification of an external storage place, typically without possibility for direct computer network access.

4.2.85 Feature

A Feature is a type of Item (see 4.2.115) that are design and production details of a ship, such as corner-features, edge-features or interior-features (openings). Each Feature is either a Composite_-Feature (see 4.2.40), or a Structural_feature (see 4.2.226).

NOTE 1 A Feature represents just the function or idea of a feature. Its details may be given by adding Definition subtypes, such as Explicit_feature_design_definition that reference this Feature.

NOTE 2 No provision is made for a Feature on this level to reference other Items, such as an owner of the feature.

4.2.86 Feature_design_definition

A Feature_design_definition is a type of Design_definition (see 4.2.58) that describes the definition of a Feature (see 4.2.85) for the life-cycle phase of design. Each Feature_design_definition may be one of the following: a Corner_cutout_design_definition (see 4.2.44), a Drain_hole_cutout_design_-definition (see 4.2.69), an Explicit_feature_design_definition (see 4.2.80), an Interior_cutout_-design_definition (see 4.2.111), a Part_edge_cutout_design_definition (see 4.2.153), a Position_-feature_design_definition (see 4.2.167), or a Structural_weld_shrinkage_allowance_feature_design_-definition (see 4.2.247).

The data associated with a Feature_design_definition are the following:

— defined_for.

4.2.86.1 defined_for

The defined_for specifies the Feature that the Feature_design_definition is defined for. There may be more than one defined_for for a Feature_design_definition.

4.2.87 Feature_relationship

A Feature_relationship is a type of Item_relationship (see 4.2.116) that defines the association of a Feature (see 4.2.85) with another Item (see 4.2.115). Each Feature_relationship is a Structural_-feature_relationship (see 4.2.227).

EXAMPLE Feature_relationship may be applied to indicate that two Seams are parallel or that one Seam is parallel to a plate boundary. Feature_relationship does not replace the parent attribute of Structural_feature.

The data associated with a Feature_relationship are the following:

— item_1.

4.2.87.1 item_1

The item_1 specifies the Feature for which the relationship is defined. See 4.3.57 for the application assertion.

4.2.88 Fillet_weld

A Fillet_weld is a type of Welded_joint_design_definition (see 4.2.276) that is a weld where the weld bead is laid down along the outside of the part. The degree of penetration achieved by this type of weld is dependent on the thickness and arrangement of the parts being welded and usually, is less than that attainable with a groove weld. The advantage it offers in elimination of edge preparation, makes this a highly desirable weld detail. Each Fillet_weld may be one of the following: a Continuous_fillet_weld (see 4.2.41), or a Intermittent_fillet_weld (see 4.2.113).

The data associated with a Fillet_weld are the following:

— endcut_shape_type;

— sidedness.

4.2.88.1 endcut_shape_type

The endcut_shape_type specifies an indicator used to denote the particutar shape of the edge cut form of plate or end cut form of profile parts being welded by fillet weld.

The value of the endcut_shape_type is one of the following:

— fillet_j;

— fillet_v;

— user_defined.

NOTE See 4.2.88.1.1 - 4.2.88.1.3 for the definition of each allowable value for endcut_shape_type.

4.2.88.1.1 fillet_j

particular shape of the edge curved cut form of plate or end cut form of profile parts being welded by fillet weld.

4.2.88.1.2 fillet_v

particular shape of the edge plane cut form of plate or end cut form of profile parts being welded by fillet weld.

4.2.88.1.3 user_defined

other fillet cut form defined by the user.

4.2.88.2 sidedness

The sidedness specifies the identifier as to whether the fillet weld is to be made at one side or at both sides of the welded joint.

The value of the sidedness is one of the following:

— both_sides;

— one_side.

NOTE See 4.2.88.2.1 - 4.2.88.2.2 for the definition of each allowable value for sidedness.

4.2.88.2.1 both_sides

both sides of a weld joint are welded.

4.2.88.2.2 one_side

one side of a weld joint is welded.

4.2.89 Flanged_plate_cross_section

A Flanged_plate_cross_section is a type of Proprietary_profile_cross_section (see 4.2.185) commonly used in shipbuilding in place of an angle cross section. A Flanged_plate_cross_section is made from a flat plate knuckled into an angular shape.

The data associated with a Flanged_plate_cross section are the following:

— depth;

— radius;

— thk;

— width.

4.2.89.1 depth

The depth specifies the depth of the web.

4.2.89.2 radius

The radius specifies the radius between the flange and the web.

4.2.89.3 thk

The thk specifies the thickness of the plate from which the cross-section is manufactured.

4.2.89.4 width

The width specifies the width of the flange.

4.2.90 Flanged_profile_cross_section

A Flanged_profile_cross_section is a type of Parametric_profile_cross_section (see 4.2.151) that is a generic type of standard shape that can be distinquished as having an identifiable flange and web. Each Flanged_profile_cross_section may be one of the following: a T_bar_cross_section (see 4.2.252) or a W_shape_cross_section (see 4.2.267).

4.2.91 Flare_area

The Flare_area describes properties of the fore ship that indicate the danger for buckling in the fore ship due to upwards pushing sea.

NOTE Flare_area may be a critical aspect especially for slim ships that go at high speeds.

The data associated with a Flare_area are the following:

— area;

— height.

4.2.91.1 area

The area specifies the difference between the area of the weather deck forward of 0.2 of the rule length of the ship measured from forward perpendicular and the corresponding area at the summer waterline.

4.2.91.2 height

The height specifies the distance from the baseline of the ship to the weather deck measured at forward perpendicular and on the ship centreline.

4.2.92 Flat_bar_cross_section

A Flat_bar_cross_section is a type of Bar_profile_cross_section (see 4.2.13) identified by a solid rectangular shape.

NOTE 1 ISO 1035-3 specifies dimensions for some flat bars.

NOTE 2 Figure 21 illustrates a Flat_bar_cross_section.

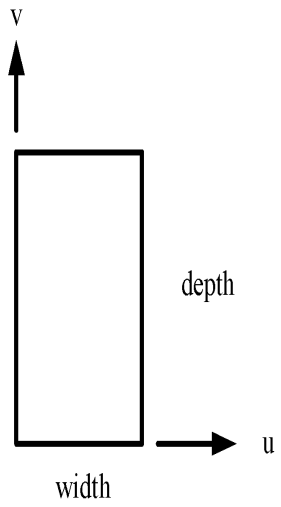


Figure 21 — Flat_bar_cross_section

The data associated with a Flat_bar_cross_section are the following:

— depth;

— width.

4.2.92.1 depth

The depth specifies the dimension of the longer of the two sides, if unequal.

4.2.92.2 width

The width specifies the dimension of the shorter of the two sides, if unequal.

4.2.93 Frame_table

A Frame_table is a type of Longitudinal_table (see 4.2.130) that has positions that reference the locaation of frames that are located along the global X-axis.

NOTE Frames are used for the internal structure of the ship and they are structural elements. A ship can have more than 100 frames. The intersection curve between a frame and the hull moulded form is a curve of transversal section through the hull of the Ship.

4.2.94 Free_form_interior_cutout_design_definition

A Free_form_interior_cutout_design_definition is a type of Interior_cutout_design_definition (see 4.2.111) that specifies an opening that can not be represented by the basic openings. A closed bounding curve is defined that represents the shape of the opening.

The data associated with a Free_form_interior_cutout_design_definition are the following:

— bounding_curve.

4.2.94.1 bounding_curve

The bounding_curve specifies the closed curve representing the opening.

4.2.95 Freeboard_characteristics

A Freeboard_characteristics is a type of General_characteristics_definition (see 4.2.97) that give details of the assignment of freeboard of a ship. Freeboard is the distance measured from the waterline to the upper edge of the deck plating at the side of the freeboard deck amidships. The freeboard deck is the uppermost continuous deck exposed to the weather and the sea which has permanent means for the watertight closure of all exposed openings on the deck and in the side shell below. The minimum freeboard is required princdipally to ensure that the ship is seaworthy when loaded and provides the ship with sufficient bouancy reserves to rise as it passes through waves and thus to remain largely dry on its decks.

The data associated with a Freeboard_characteristics are the following:

— applicable_loadline;

— assigned_code;

— date_freeboard_assigned;

— freeboard;

— freeboard_assigned_by.

4.2.95.1 applicable_loadline

The applicable_loadline specifies the resulting Loadline for the assigned freeboard. See 4.3.58 for the application assertion.

4.2.95.2 assigned_code

The assigned_code specifies the type of loadline according to ship type. These are defined fully in the International Convention on Load Lines, 1966 Regulation 27.

The value of the assigned_code is one of the following:

— a;

— a_plus;

— b;

— b_pluse;

— b_60;

— b_100;

— other.

NOTE See 4.2.95.2.1 - 4.2.95.2.7 for the definition of each allowable value for assigned_code.

4.2.95.2.1 a

a Type A ship. Those which are designed to carry only liquid cargoes in bulk, and in which the cargo tanks have only small access openings closed by watertight gasketed covers of steel or equivalent material.

4.2.95.2.2 a_plus

a Type A ship over 150 metres in length.

4.2.95.2.3 b

a Type b ship. Those which do not fall into the category of Type A ships.

4.2.95.2.4 b_plus

a Type B ship with hatchways fitted with portable beams and covers on exposed freeboard or raised quarterdecks, and within 25 percent of the ship superstructure decks. These have the basic freeboard increased.

4.2.95.2.5 b_60

a Type B ship which comply with not such severe sub-division requirements as the B-100 type. It may be assigned a basic freeboard reduced by up to 60 percint of the difference between B and A basic values.

4.2.95.2.6 b_100

a Type B ship which is effectively adopting a Type A freeboard by virtue of having steel weathertight covers fitted with gaskets and clamping devices, improved measures for the protection of the crew, better freeing arrangements, and satisfactory sub-division characteristics. It may be assigned a basic freeboard reduced by up to 100 percent of the difference between B and A basic values.

4.2.95.2.7 other

an unspecified type of vessel for the freeboard calculation.

4.2.95.3 date_freeboard_assigned

The date_freeboard_assigned specifies the date and time the freeboard is assigned.

4.2.95.4 freeboard

The freeboard specifies the assigned freeboard, which is the difference between the actual depth at side including the thickness of the stringer plate and wooden deck sheathing, if fitted, and the summer load draught.

4.2.95.5 freeboard assigned by

The freeboard_assigned_by specifies the organization which made the freeboard assignment.

4.2.96 Functional_definition

A Functional_definition is a type of Definition (see 4.2.56) that is the basis for all types of functional definitions. It provides the capability to specify a role or purpose for a Definition. Each Functional_-definition is either an Edge_cutout_functional_definition (see 4.2.72), an Edge_feature_functional_-definition (see 4.2.74), an Interior_cutout_functional_definition (see 4.2.112), a Shiptype (see 4.2.207), a Structural_part_functional_definition (see 4.2.232), or a Structural_system_functional_-definition (see 4.2.242).

The data associated with a Functional_definition are the following:

— local_units;

— user_def_function.

4.2.96.1 local_units

The local_units specifies that a Functional_definition is not to define local units.

4.2.96.2 user_def_function

The user_def_function specifies a user-defined role or purpose of the Functional_definition. The user_def_function need not to be specified for a particular Functional_definition.

4.2.97 General_characteristics_definition

A General_characteristics_definition is a type of Definition (see 4.2.56) that provides a major part of the documentation of a Ship. It includes primary dimensions and capacities due to the contract of the product (Ship). Each General_characteristics_definition is either a Class_and_statutory_designation (see 4.2.36), a Class_parameters (see 4.2.39), a Design_load (see 4.2.59), a Freeboard_characteristics (see 4.2.95), a Global_axis_placement (see 4.2.99), a Owner_designation (see 4.2.143), a Principal_-characteristics (see 4.2.170), a Ship_designation (see 4.2.205), or a Shipyard_designation (see 4.2.208).

The data associated with a General_characteristics_definition are the following:

— defined_for.

4.2.97.1 defined_for

The defined_for specifies a Ship or set of Ships for which the General_characteristics_definition applies. There may be more than one defined_for for a General_characteristics_definition. See 4.3.59 for the application assertion.

4.2.98 Geometrically_bounded_wireframe_shape

A Geometrically_bounded_wireframe_shape is a shape representation which conforms to 10303-510.

4.2.99 Global_axis_placement

A Global_axis_placement is a type of General_characteristics_definition (see 4.2.97) that defines a fixed system of right-handed orthogonal axes to which geometric data are referred. A Global_axis_-placement is to have a positive Z-axis in an upwards direction starting from the base of the Ship and a positive X-axis running along the Ship on the intersection of the centreline with the base. In one case it is directed from the aft part of the Ship to the forward part of the Ship, in the other, it is directed from the forward part of the Ship to the aft part of the Ship. The origin of the Global_axis_placement can be any point on the X-axis. The distance of the after perpendicular from the origin and the orientation of the X-axis is specified for the Ship. If any other system of axes is used, local or global, then the transformation relations between it and the Global_axis_placement is specified.

NOTE Figure 22 illustrates different Global_axis_placement objects.

The data associated with a Global_axis_placement are the following:

— after_perpendicular_offset;

— orientation.

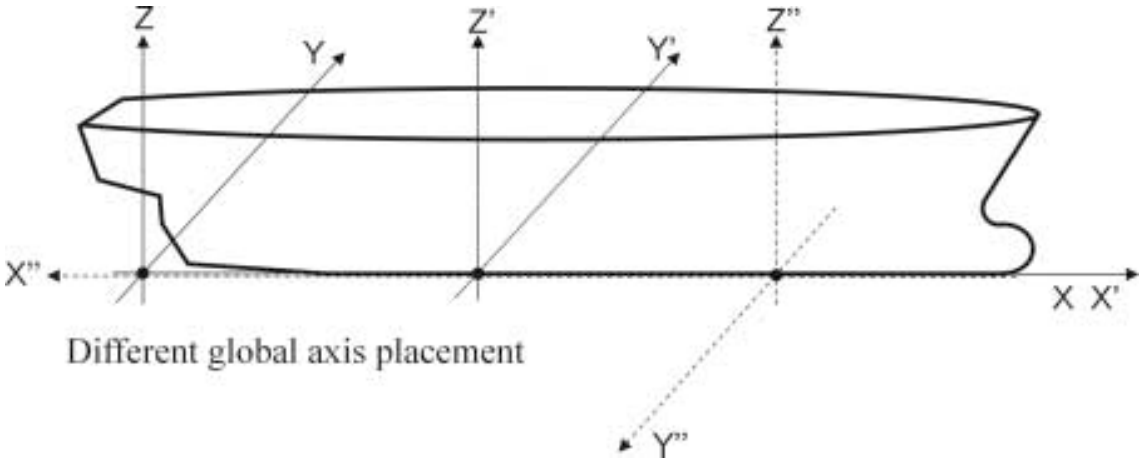


Figure 22 — Global axis placement

4.2.99.1 after_perpendicular_offset

The after_perpendicular_offset specifies the distance from the origin of the Global_axis_placement to the after perpendicular.

4.2.99.2 orientation

The orientation specifies the direction of the X-axis.

The value of orientation is one of the following:

— aft_pointing;

— forward_pointing.

4.2.99.2.1 aft_pointing

an orientation of a right-handed ship coordinate system that has the positive X-axis from the forward part of the Ship directed to the aft part of the Ship.

4.2.99.2.2 forward_pointing

an orientation of a right-handed ship coordinate system that has the positive X-axis from the aft part of the Ship directed to the forward part of the Ship.

4.2.100 Global_id

A Global_id is a persistent, global identifier which uniquely identifies the product data.

NOTE A guideline as to the format for Global_id is provided in annex L, section 4.

The data associated with a Global_id are the following:

— id.

4.2.100.1 id

The id specifies a unique, persistent identifier generated by the company that creates the product data.

4.2.101 Groove_weld

A Groove_weld is a type of Welded_joint_design_definition (see 4.2.276) that is a weld between the structural parts where, either due to increased part thickness or highly stressed joints, deeper penetration of weld materials is required. In order to achieve this penetration, the edges of the parts to be joined require edge preparation such as bevel chamfer features or grinding features. Each Groove_weld may be one of the following: a Beveled_groove_weld (see 4.2.15), or a Butt_groove_-weld (see 4.2.21).

NOTE 1 This type of weld is usually stronger but more costly than a fillet weld. It is usually used in highly stressed and critical areas of the ship, such as the shell plating, decks, bulkheads.

NOTE 2 Figure 23 illustrates groove weld types.

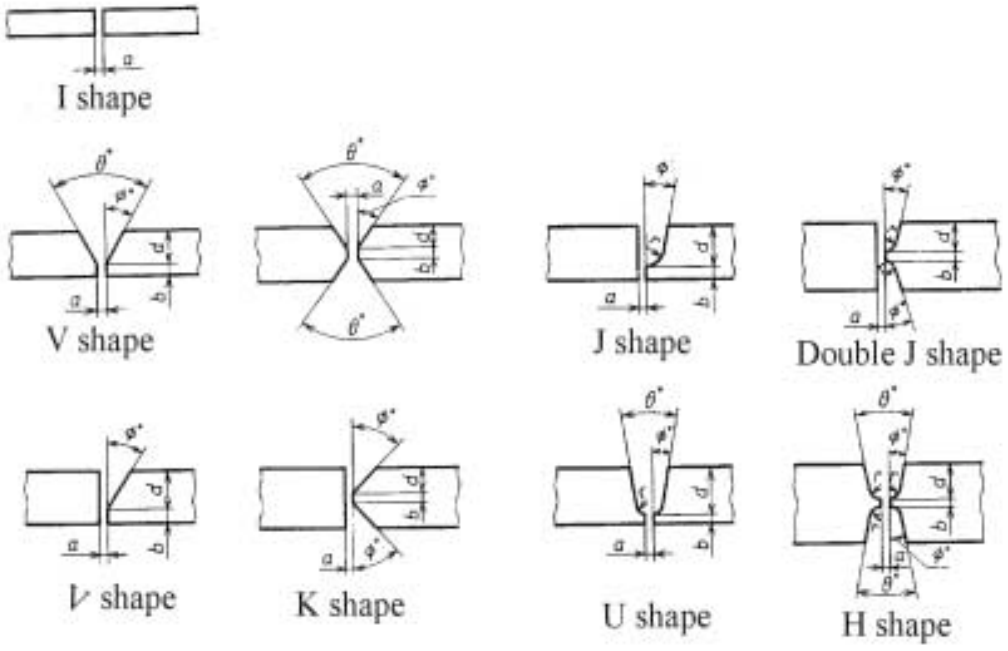


Figure 23 — Groove weld types

The data associated with a Groove_weld are the following:

— sidedness;

— weld_backing_type;

— weld_joint_spacer.

4.2.101.1 sidedness

The sidedness specifies the identifier as to denote whether the groove weld is to be made at one side or at both sides of the welded joint.

The value of the sidedness is one of the following:

— both_sides;

— one_side.

NOTE See 4.2.101.1.1 - 4.2.101.1.2 for the definition of each allowable value for sidedness.

4.2.101.1.1 both_sides

both sides of a weld joint are welded.

4.2.101.1.2 one_side

one side of a weld joint is welded.

4.2.101.2 weld_backing_type

The weld_backing_type specifies the type of required underside or back of the weld in order to prevent blow through or eliminate the need for back gouging. Proper selection of backing reduces manufacturing costs by making it possible to joint parts by welding from a single side. The weld backing type need not to be specified for a particular Groove weld.

EXAMPLE A cut-down channel bar used as a deck beam, the upper flange providing the backing bar for a deck panel butt weld, made by machine above.

The value of the weld_backing_type is one of the following:

— ceramic_tape;

— copper_backing_bar;

— flare_backing_ring;

— none;

— permanent;

— permanent_backing_ring;

— removable_backing_ring;

— user_defined.

NOTE See 4.2.101.2.1 - 4.2.101.2.8 for the definition of each allowable value for weld backing type.

4.2.101.2.1 ceramic_tape

ceramic_tape: a backing tape made of ceramic.

4.2.101.2.2 copper_backing_bar

copper_backing_bar: a backing bar made of copper.

4.2.101.2.3 flare_backing_ring

flare_backing_ring: a backing ring of flare form.

4.2.101.2.4 none

none: no backing is defined.

4.2.101.2.5 permanent

permanent: a backing attached at the welded joint permanently.

4.2.101.2.6 permanent_backing_ring

permanent_backing_ring: a backing ring attached at the welded joint permanently .

4.2.101.2.7 removable_backing_ring

removable_backing_ring: a removable backing ring.

4.2.101.2.8 user_defined

user_defined: a backing defined by the user.

4.2.101.3 weld_joint_spacer

The weld_joint_spacer specifies an identifier as to whether or not the joint requires the use of additional material to maintain the weld dimensions before or during the welding process. This is a Boolean identifier that indicates that a spacer is required when TRUE, or a spacer is not required when FALSE. The geometric definition of the spacer must be derived from the joint detailed dimensions.

4.2.102 Homogeneous_ship_material_property

A Homogeneous_ship_material_property is a type of Ship_material_property (see 4.2.206) that is used to specify the raw material of this class by its physical properties.

The data associated with a Homogeneous_ship_material_property are the following:

— defined_for;

— poisson_ratio;

— stress_of_fracture;

— thermal_expansion_coeffcient;

— yield_point;

— youngs_modulus.

4.2.102.1 defined_for

The defined_for specifies re-declaration of the defined_for attribute in order to allow the Homogeneous_ship_material_property, as a type of definition, to point to Part objects only. There may be more than one defined_for for a Homogeneous_ship_material_property. See 4.3.60 for the application assertion.

4.2.102.2 poisson_ratio

The poisson_ratio specifies poisson's ratio (nu) is the ratio of lateral unit strain to longitudinal unit strain under the condition of uniform and uniaxial longitudinal stress within the proportional limit.

4.2.102.3 stress_of_fracture

The stress_of_fracture specifies the ultimate strength at which a brittle material fractures and a ductile material continues to stretch.

NOTE Ultimate strength of material is the maximum load divided by the original cross sectional area of the material.

4.2.102.4 thermal_expansion_coefficient

The thermal_expansion_coefficient specifies the change in length of the material per unit length per degree change in temperature.

4.2.102.5 yield_point

The yield_point specifies the minimum stress of the material at which the material deforms appreciably without an increase in stress.

4.2.102.6 youngs_modulus

The youngs_modulus specifies the Young's modulus (modulus of elasticity, E) is the rate of change of unit tensile or compressive stress with respect to unit tensile or compressive strain for the condition of uniaxial stress within the proportional limit (Hooke's law)

NOTE The shear modulus (modulus of rigidity, G) is derivable from G = E/(2 * (1 + nu)) with the Young's modulus (E) and the Poisson's ratio (nu).

4.2.103 Hull_applicability

Hull_applicability is the identification of a ship hull, or a range of hulls within a class of ships, for which a particular product data are applicable. The data associated with a Hull_applicability are the following:

— definitions_for_hulls;

— end_hull;

— items_for_hulls;

— start_hull.

4.2.103.1 definitions_for_hulls

definitions_for_hulls specifies the Definition objects (see 4.2.56) that are applicable for the range of hulls specified in start_hull and end_hull. The definitions_for_hulls need not be specified for a particular Hull_applicability. There may be more than one definitions_for_hulls for a Hull_applicability. See 4.3.61 for the application assertion.

4.2.103.2 end_hull

The end_hull specifies the final hull in a range of hulls for which the product data is applicable. The end_hull need not be specified for a particular Hull_applicability. If the end_hull is not specified, the product data is applicable to only the start_hull. If the end_hull and the start_hull are both not specified, the product data is applicable to all hulls.

4.2.103.3 items_for_hulls

The items_for_hulls specifies the Item objects (see 4.2.115) that are applicable for the range of hulls specified in start_hull and end_hull. . The items_for_hulls need not be specified for a particular Hull_applicability. There may be more than one items_for_hulls for a Hull_applicability. See 4.3.62 for the application assertion.

4.2.103.4 start_hull

The start_hull specifies the first hull in a range of hulls for which the product data is applicable.

4.2.104 Hull_cross_section

A Hull_cross_section is a type of both Item (see 4.2.115) and Item_structure (see 4.2.117) that is a collection of all those parts of the structure of a ship which are relevant to its longitudinal strength.

NOTE 1 A Hull_cross_section is an Item in the context of a Ship. It has a location along the global longitudinal axis with transverse and vertical axis aligned with the corresponding ones in the global hull coordinate system. The location is to be specified by a Local_co_ordinate_system_with_station_ reference definition. A Hull_-cross_section is also an Item_structure that references compartments and structural systems. These items are to have Definitions that specify the information that is necessary for the Hull_cross_section approval by classification society. These Definitions are class compartment requirement definition and class deck load requirement definition. All the definitions that together describe a Hull_cross_section are collected by Revision_with_context with Revision_with_context .members being these definitions, one of which is a Hull_-cross_section_design_definition, and Revision_ with_context.context_of_revision being the Hull_cross_section.

NOTE 2 For a Hull_cross_section the ship is viewed as a structural beam, with a length of approximately the length between perpendiculars and with cross sectional properties varying over its length. A Hull_cross_section includes all data that are necessary for its design and its approval with respect to class rule requirements. The model is applicable to any kind of ship.

NOTE 3 Figure 25 illustrates a hull cross section of a cargo ship.

NOTE 4 From the view of ship structures Figure 26 illustrates a hull cross section of a bulk carrier, Figure 24 illustrates its double bottom.

EXAMPLE Panels in bottom, side, deck, bulkheads, girders, and stringers are examples of Structural_system objects that may be relevant in this context.

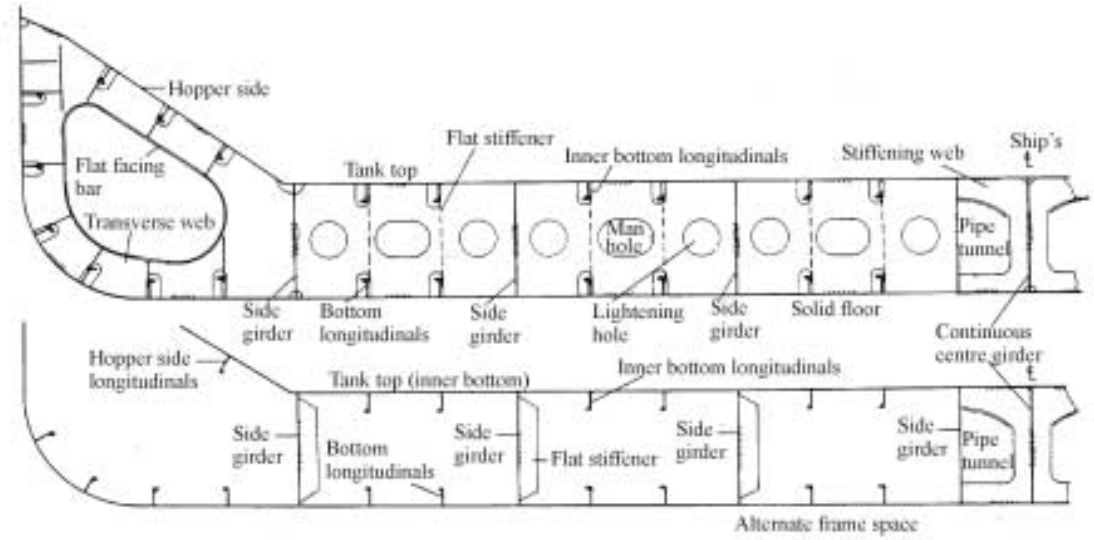


Figure 24 — Double bottom of a bulk carrier

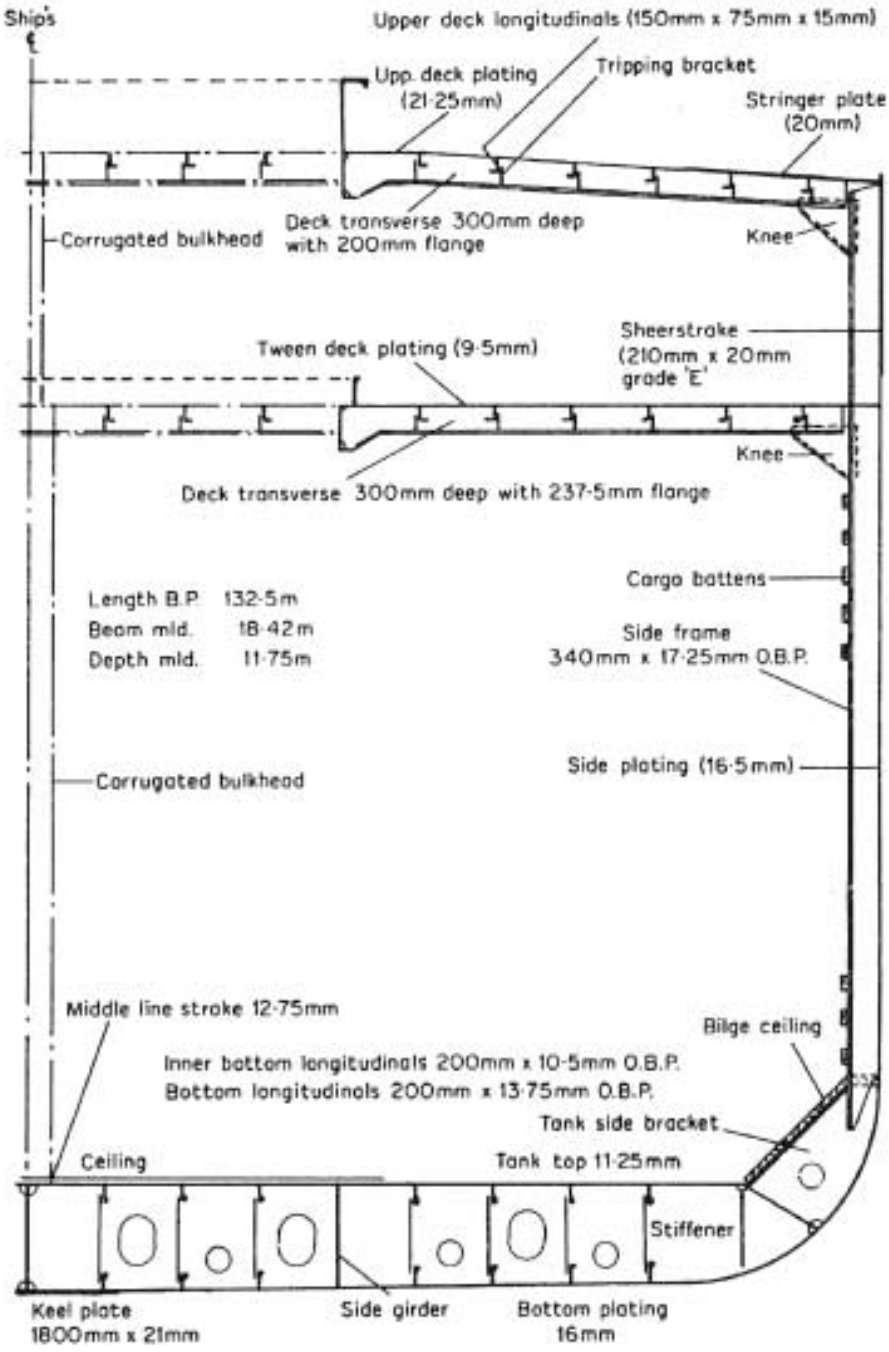


Figure 25 — Hull cross section of a cargo ship

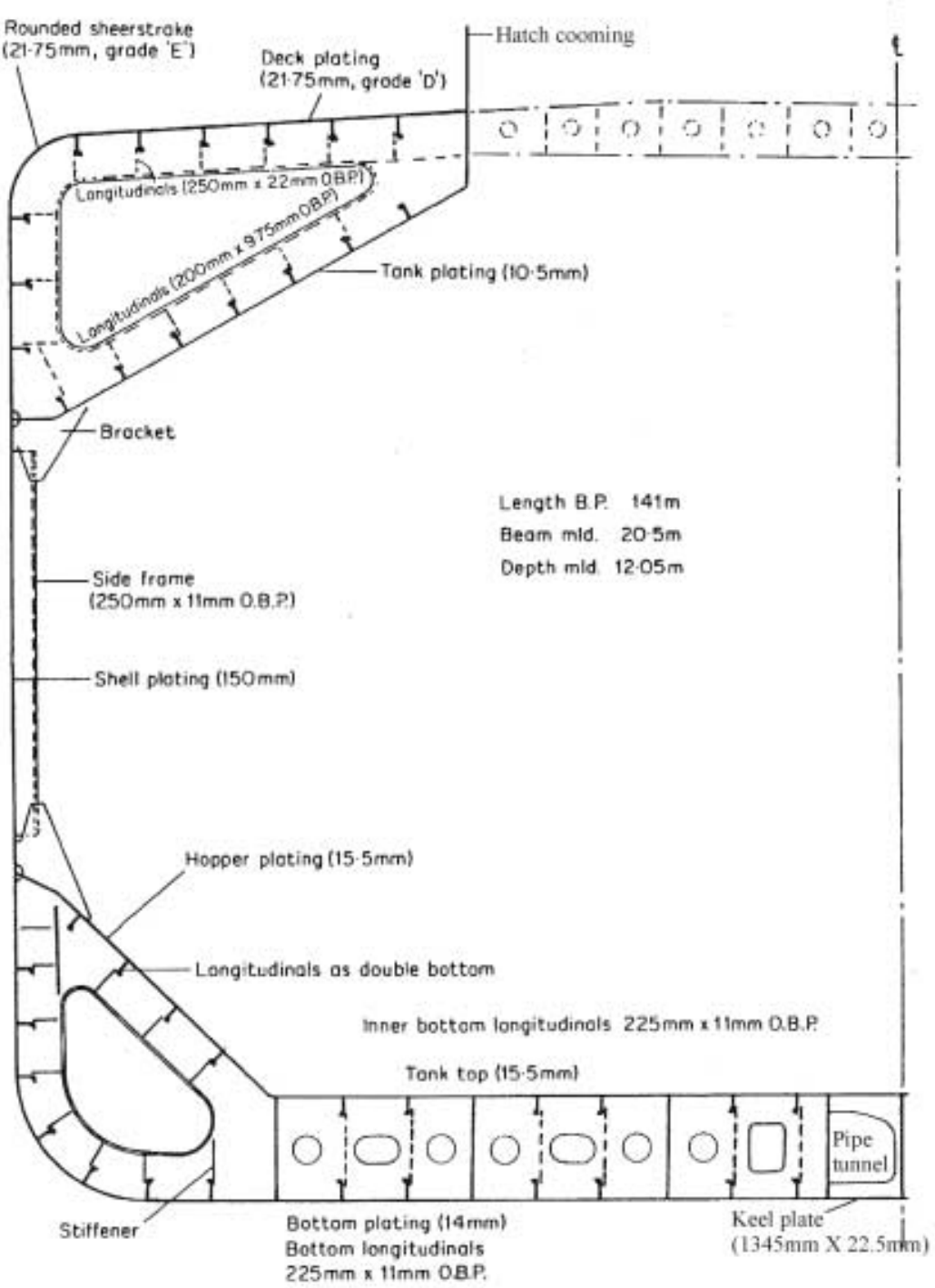


Figure 26 — Hull cross section of a bulk carrier

4.2.105 Hull_cross_section_approval

A Hull_cross_section_approval is a type of Structural_class_approval (see 4.2.219) and Hull_cross_-section result (see 4.2.108) that confirms the approval of a Hull_cross_section_design_definition (see 4.2.106).

NOTE A Hull_cross_section_design_definition is approved if all its constituents are approved and if the requirements as stated in Hull_cross_section_result are fulfilled.

4.2.106 Hull_cross_section_design_definition

A Hull_cross_section_design_definition is a type of Design_definition (see 4.2.58) that describes the definition of a Hull_cross_section (see 4.2.104) from the view of design for classification approval.

NOTE The representations of a Hull_cross_section_design_definition is either an Edge_based_wireframe_-shape, or a Non_manifold_surface_shape. There is at least one Revision_with_context with a Hull_cross_-section_design_definition. Each Revision_with_context that has a Hull_cross_section_design_definition as one of its members is to have only one.

The data associated with a Hull_cross_section_design_definition are the following:

— defined_for;

— flare_area_buckling;

— mirrored_symmetry.

4.2.106.1 defined_for

The defined_for specifies the one Hull_cross_section item that a Hull_cross_section_design_-definition is defined for. See 4.3.64 for the application assertion.

NOTE A Hull_cross_section may have many Hull_cross_section_design_definitions, each describing a different version.

4.2.106.2 flare_area_buckling

The flare_area_buckling specifies the part of the foreship that is to withstand the forces that are introduced by upwards pushing sea. See 4.3.63 for the application assertion.

4.2.106.3 mirrored_symmetry

The mirrored_symmetry specifies the indication whether the hull cross section is symmetrical to the mid-ship plane. The Boolean value is set to TRUE if mirrored_symmetry exists.

NOTE If mirrored symmetry is false, the entire hull cross section shall be provided.

4.2.107 Hull_cross_section_rejection

A Hull_cross_section_rejection is a type of Structural_class_rejection (see 4.2.222) and Hull_cross_-section_result (see 4.2.108) that confirms that a Hull_cross_section_design_definition (see 4.2.106) is not approved.

4.2.108 Hull_cross_section_result

A Hull_cross_section_result is a type of Structural_class_approval_result (see 4.2.220) that specifies the result of the assessment of a Hull_cross_section_design_definition (see 4.2.106) against for example the rules of a classification society. Subtypes of this object indicate whether the item passed the verification or not. If the Hull_cross_section_design_definition contains any item that is not approved, the whole Hull_cross_section (see 4.2.104) is disapproved. In addition there is a set of requirements that the Hull_cross_section_design_definition as a whole is fulfill to be approved. These are specified by this object. Each Hull_cross_section_result may be one of the following: a Hull_cross_section_approval (see 4.2.105), or a Hull_cross_section_rejection (see 4.2.107).

The data associated with a Hull_cross_section_result are the following:

— actual_first_moment_horizontal;

— actual_first_moment_vertical;

— actual_inertia_cross;

— actual_inertia_horizontal;

— actual_inertia_torsion;

— actual_inertia_vertical;

— actual_neutral_axis_position_horizontal;

— actual_neutral_axis_position_vertical;

— actual_section_area;

— actual_section_modulus_bottom;

— actual_section_modulus_coaming;

— actual_section_modulus_deck;

— actual_stress_bottom;

— actual_stress_coaming;

— actual_stress_deck;

— actual_twist_centre_horizontal;

— actual_twist_centre_vertical;

— direction_largest_inertia;

— direction_least_inertia;

— max_moment;

— max_shear_force;

— meter_weight;

— min_moment;

— min_shear_force;

— required_section_modulus_bottom;

— required_section_modulus_coaming;

— required_section_modulus_deck;

— required_stress_bottom;

— required_stress_coaming;

— required_stress_deck;

— subject.

4.2.108.1 actual_first_moment_horizontal

The actual_first_moment_horizontal specifies calculated first moment about the horizontal neutral axis.

4.2.108.2 actual_first_moment_vertical

The actual_first_moment_vertical specifies calculated first moment about the vertical neutral axis.

4.2.108.3 actual_inertia_cross

The actual_inertia_cross specifies calculated cross moment of inertia.

4.2.108.4 actual_inertia_horizontal

The actual_inertia_horizontal specifies calculated moment of inertia about the horizontal neutral axis.

4.2.108.5 actual_inertia_torsion

The actual_inertia_torsion specifies calculated torsion moment of inertia.

4.2.108.6 actual_inertia_vertical

The actual_inertia_vertical specifies calculated moment of inertia about the vertical neutral axis.

4.2.108.7 actual_neutral_axis_position_horizontal

The actual_neutral_axis_position_horizontal specifies the horizontal distance in this cross section between the longitudinal axis of the global coordinate system and the neutral axis.

4.2.108.8 actual_neutral_axis_position_vertical

The actual_neutral_axis_position_vertical specifies the vertical distance in this cross section between the longitudinal axis of the global coordinate system and the neutral axis.

4.2.108.9 actual_section_area

The actual_section_area specifies the calculated area of the hull_cross_section_design_definition as determined by the analysis.

4.2.108.10 actual_section_modulus_bottom

The actual_section_modulus_bottom specifies actual section modulus for the bottom as a result of the proposed structure.

4.2.108.11 actual_section_modulus_coaming

The actual_section_modulus_coaming specifies actual section modulus for the coaming as a result of the proposed structure.

4.2.108.12 actual_section_modulus_deck

The actual_section_modulus_deck specifies actual section modulus for the deck as a result of the proposed structure.

4.2.108.13 actual_stress_bottom

The actual_stress_bottom specifies calculated stress in the bottom.

4.2.108.14 actual_stress_coaming

The actual_stress_coaming specifies calculated stress in the coaming.

4.2.108.15 actual_stress_deck

The actual_stress_deck specifies calculated stress in the weather deck.

4.2.108.16 actual_twist_centre_horizontal

The actual_twist_centre_horizontal specifies the horizontal distance in this cross section between the longitudinal axis of the global coordinate system and the centre of twisting.

4.2.108.17 actual_twist_centre_vertical

The actual_twist_centre_vertical specifies the vertical distance in this cross section between the longitudinal axis of the global coordinate system and the centre of twisting.

4.2.108.18 direction_largest_inertia

The direction_largest_inertia specifies calculated direction angle of the largest moment of inertia.

4.2.108.19 direction_least_inertia

The direction_least_inertia specifies calculated direction angle of the least moment of inertia.

4.2.108.20 max_moment

The max_moment specifies maximum allowable still water bending moment.

4.2.108.21 max_shear_force

The max_shear_force specifies maximum allowable shear force.

4.2.108.22 meter_weight

The meter_weight specifies weight of longitudinal structural members per meter.

4.2.108.23 min_moment

The min moment specifies required minimum still water bending moment.

4.2.108.24 min_shear_force

The min_shear_force specifies required minimum shear force.

4.2.108.25 required_section_modulus_bottom

The required_section_modulus_bottom specifies required section modulus for the bottom.

4.2.108.26 required_section_modulus_coaming

The required_section_modulus_coaming specifies required section modulus for the coaming.

4.2.108.27 required_section_modulus_deck

The required_section_modulus_deck specifies required section modulus for the deck.

4.2.108.28 required_stress_bottom

The required_stress_bottom specifies required stress in the bottom.

4.2.108.29 required_stress_coaming

The required_stress_coaming specifies required stress in the coaming.

4.2.108.30 required_stress_deck

The required_stress_deck specifies required stress in the weather deck.

4.2.108.31 subject

The subject specifies the approved rejected Hull_cross_section_design_definition. See 4.3.65 for the application assertion.

4.2.109 Integer_property_value

An Integer_property_value is a type of Property_value (see 4.2.184) that provides a value for a property of type integer.

The data associated with an Integer_property_value are the following:

— val.

4.2.109.1 val

The val specifies the value of type integer.

4.2.110 Interior_cutout

An Interior_cutout is a type of Structural_cutout (see 4.2.223) that specifies holes in structural parts, such as manholes, access holes, lightening holes. Many types of Interior cutouts may be distinguished. These are to be specified using different interior cutout design definitions, such as:

— Circular_cutout_design_definition (see 4.2.34),

— Elliptical_cutout_design_definition (see 4.2.76),

— Elongated_oval_cutout_design_definition (see 4.2.77),

— Free_form_interior_cutout_design_definition (see 4.2.94),

— Rectangular_cutout_design_definition (see 4.2.189),

— Round_corner_rectangular_cutout_design_definition (see 4.2.196),

— Round_edge_rectangular_cutout_design_definition (see 4.2.197),

— Triangular_cutout_design_definition (see 4.2.255).

4.2.111 Interior_cutout_design_definition

An Interior_cutout_design_definition is a type of Feature_design_definition (see 4.2.86) that specifies a design definition for an opening. Each Interior_cutout_design_definition may be one of the following: a Circular_cutout_design_definition (see 4.2.34), an Elliptical_cutout_design_-definition (see 4.2.76), an Elongated_oval_cutout_design_definition (see 4.2.77), a Free_form_-interior_cutout_design_definition (see 4.2.94), a Rectangular_cutout_design_definition (see 4.2.189), or a Triangular_cutout_design_definition (see 4.2.255).

The data associated with an Interior_cutout_design_definition are the following:

— defined_for;

4.2.111.1 defined_for

The defined_for specifies the Interior_cutout feature that an Interior_cutout_design_definition is valid for. There may be more than one defined_for for an Interior_cutout_design_definition. See 4.3.66 for the application assertion.

4.2.112 Interior_cutout_functional_definition

An Interior_cutout_functional_definition is a type of Functional_definition (see 4.2.96) that defines the function that an Interior_cutout (see 4.2.110) may be designed for.

The data associated with an Interior_cutout_functional_definition are the following:

— defined_for;

— the_function.

4.2.112.1 defined_for

The defined_for specifies the Interior_cutout feature that this function is assigned to. There may be more than one defined_for for an Interior_cutout_functional_definition. See 4.3.67 for the application assertion.

4.2.112.2 the_function

The the_function specifies the purpose for the Interior_cutout in a ship structure.

The value of the_function is one of the following:

— access_hole;

— air_escape;

— air_liquid_escape;

— foothold;

— lightening_hole;

— liquid_escape;

— penetration;

— user_defined.

NOTE See 4.2.112.2.1 - 4.2.112.2.8 for the definition of each allowable value for the_function.

4.2.112.2.1 access_hole

a void that permits entry to an area from another area

4.2.112.2.2 air_escape

a void that permits air to be released, dispelled, or removed

4.2.112.2.3 air_liquid_escape

a void that permits air or liquid to be released, dispelled, or removed

4.2.112.2.4 foothold

a cutout to provide footing or a hold for the feet.

4.2.112.2.5 lightening_hole

material removed to reduce weight

4.2.112.2.6 liquid_escape

a void that permits liquid to be released, dispelled, or removed

4.2.112.2.7 penetration

a void that permits something to pass through

EXAMPLE A cutout for a stiffener or a pipe that passes through a bulkhead

4.2.112.2.8 user_defined

the Interior_cutout is defined within the user_def_function attribute, by the user .

NOTE The user_defined option may only be used when the Interior_cutout cannot be defined by the other function codes

4.2.113 Intermittent_fillet_weld

An Intermittent_fillet_weld is a type of Fillet_weld (see 4.2.88) where the weld bead is deposited in a sequence of short continuous length, spaced at regular intervals. The combination of a weld length and spacing forms a pattern that is repeated along the length of the weld. The primary use of intermittent welds is to reduce distortion in thinner structural parts and to reduce manufacturing costs in lightly stressed members.

The data associated with an Intermittent_fillet_weld are the following:

— cutout_rules;

— end_rules;

— fillet_alignment;

— fillet_weld_length;

— fillet_weld_spacing;

— penetration_rules.

4.2.113.1 cutout_rules

The cutout_rules specifies the aspect of an Intermittent_fillet_weld that defines the rule of length of weld that must be made on each side of a cutout.

The value of the cutout_rules is one of the following:

— fixed_rule;

— member_depth;

— none;

— percent_length.

NOTE See 4.2.113.1.1 - 4.2.113.1.4 for the definition of each allowable value for cutout_rules.

4.2.113.1.1 fixed_rule

fixed_rule: fixed intermittent weld rule is defined.

4.2.113.1.2 member_depth

member_depth: intermittent weld rule depending on the welded metals.

4.2.113.1.3 none

none: no intermittent weld rule is defined.

4.2.113.1.4 percent_length

percent_length: intermittent weld rule which is expressed as a percent of the joint length of the base metal being welded.

4.2.113.2 end_rules

The end_rules specifies the an aspect of an Intermittent_fillet_weld that defines the rule of length of weld that must be made on each end.

The value of the end_rules is one of the following:

— fixed_rule;

— member_depth;

— none;

— percent_length.

NOTE See 4.2.113.2.1 - 4.2.113.2.4 for the definition of each allowable value for end_rules.

4.2.113.2.1 fixed_rule

fixed_rule: fixed intermittent weld rule is defined.

4.2.113.2.2 member_depth

member_depth: intermittent weld rule depending on the welded metals.

4.2.113.2.3 none

none: no intermittent weld rule is defined.

4.2.113.2.4 percent_length

percent_length: intermittent weld rule which is expressed as a percent of the joint length of the base metal being welded.

4.2.113.3 fillet_alignment

The fillet_alignment specifies the an indicator as to whether or not each one of paired single fillet weld making up an intermittent double fillet weld are aligned with each other or offset from each other.

The value of the fillet_alignment is one of the following:

— chained;

— staggered.

NOTE See 4.2.113.3.1 - 4.2.113.3.2 for the definition of each allowable value for fillet_alignment.

4.2.113.3.1 chained

chained: the intermittent fillet welds aligned with each other.

4.2.113.3.2 staggered

staggered: the intermittent fillet welds with offset from each other.

4.2.113.4 fillet_weld_length

The fillet_weld_length specifies the measure of the length of continuous weld bead to be deposited. These lengths are spaced at intervals defined by fillet weld spacing and are adjusted at the ends of the weld and at the location of cutouts.

4.2.113.5 fillet_weld_spacing

The fillet_weld_spacing specifies the measure of the distance between intermittent fillet weld lengths. The value represents the greatest distance between weld lengths that is not welded.

EXAMPLE If the weld length is 75 cm and the spacing is 150 cm, then only 33%of 75/(75+150) of physical weld joint is actually welded.

4.2.113.6 penetration_rules

The penetration_rules specifies an aspect of an Intermittent fillet weld that defines the rule of penetration of weld that must be made on the intermittent weld.

The value of the penetration_rules is one of the following:

— fixed_rule;

— member_depth;

— none;

— percent_length.

NOTE See 4.2.113.6.1 - 4.2.113.6.4 for the definition of each allowable value for penetration_rules.

4.2.113.6.1 fixed_rule

fixed_rule: fixed intermittent weld rule is defined.

4.2.113.6.2 member_depth

member_depth: intermittent weld rule depending on the welded metals.

4.2.113.6.3 none

none: no intermittent weld rule is defined.

4.2.113.6.4 percent_length

percent_length: intermittent weld rule which is expressed as a percent of the joint length of the base metal being welded.

4.2.114 Inward_round_corner_design_definition

An Inward_round_corner_design_definition is a type of Round_corner_design_definition (see 4.2.195) that specifies the radius and location of the centre of the circular arc representing the inward round corner. The transition between the circular arc and the adjacent edges has a discontinuous derivative.

NOTE Figure 27 illustrates an Inward_round_corner_cutout.

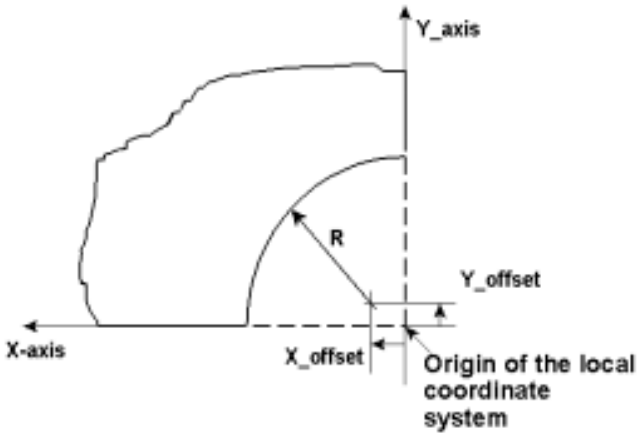


Figure 27 — Inward_round_corner_cutout

The data associated with an Inward_round_corner_design_definition are the following:

— x_offset;

— y_ offset.

4.2.114.1 x_offset

The x_offset specifies the distance of the centre of the circular arc from the origin of the local coordinate system in positive x-direction.

4.2.114.2 y_offset

The y_offset specifies the distance of the centre of the circular arc from the origin of the local coordinate system in positive y-direction.

4.2.115 Item

An Item is a type of Definable_object (see 4.2.55) that is discrete, identifiable object used in one or more design, production, or operational activities. An Item is something to be created by a physical or mental activity or automatically derived from one or more other Items. An Item needs not to represent a physically realizable object; it may also represent some abstract concept. Item provides the functionality to have relationships to other Items and to be member in an Item structure. Each Item is either an Assembly (see 4.2.8), a Change (see 4.2.25), a Feature (see 4.2.85), a Hull_cross_-section (see 4.2.104), a Part (see 4.2.152), a Ship (4.2.204), a Structural_part_connection_-implementation (see 4.2.230), or a System (see 4.2.249).

NOTE An Item may represent some abstract concept such as an activity or task.

The data associated with an Item are the following:

— description;

— documentation;

— name;

— ship_context.

4.2.115.1 description

The description specifies the description for Item. The description need not to be specified for a particular Item.

4.2.115.2 documentation

The documentation specifies the documentation available for Item. The documentation need not be specified for a particular Item. See 4.3.68 for the application assertion.

4.2.115.3 name

The name specifies the human readable name of the concept that is represented by an Item.

4.2.115.4 ship_context

The ship_context specifies the context of Item in terms of its applicability to a Ship. The ship_-context need not to be specified for a particular Item. There may be more than one ship_context for an Item. See 4.3.69 for the application assertion.

4.2.116 Item_relationship

An Item_relationship is a type of both Definable_object (see 4.2.55) and Versionable_object (see 4.2.263) that defines the association of two Item objects (see 4.2.115). The related Item objects may share a common function or activity, or are dependent on each other. Each Item_relationship is either an Assembly_relationship (see 4.2.12), a Feature_relationship (see 4.2.87), a Part_relationship (see 4.2.154), a Structural_part_joint (see 4.2.233), a Structural_part_penetration_relationship (see 4.2.236), a Structural_system_penetration_relationship (see 4.2.243), or a System_relationship (see 4.2.243)

The data associated with an Item_relationship are the following:

— external_item_1;

— external_item_2;

— item_1;

— item_2.

4.2.116.1 external_item_1

The external_item_1 specifies the relating Item of the relationship in the case where it is an External_instance_reference of an Item. The external_item_1 need not be specified for a particular Item_relationship. See 4.3.70 for the application assertion.

4.2.116.2 external_item_2

The external_item_2 specifies the related Item of the relationship in the case where it is an External_instance_reference of an Item. The external_item_2 need not be specified for a particular Item_relationship. See 4.3.70 for the application assertion.

4.2.116.3 item_1

The item_1 specifies the relating Item of the relationship in the case where it is in the same instance model as the Item. The item_1 need not be specified for a particular Item_relationship. it is in the same instance model as itself. See 4.3.71 for the application assertion.

NOTE The items related by Item may be either local instances or external instances; constraints ensure that either the local or the external instances exist.

4.2.116.4 item_2

The item_2 specifies the related Item of the relationship in the case where it is in the same instance model as the Item. The item_2 need not be specified for a particular Item_relationship. See 4.3.71 for the application assertion.

4.2.117 Item_structure

An Item_structure is a type of both Definable_object (see 4.2.55) and Versionable_object (see 4.2.263) that is a collection of Item objects (see 4.2.115) possibly related by Item_relationship objects (see 4.2.116). An Item structure forms a graph without any restriction neither regarding the number of entries, the connectivity nor the cyclicity. Each Item_structure is either an Assembly (see 4.2.8), a Hull_cross_section (see 4.2.104), or a System (see 4.2.249).

The data associated with an Item_structure are the following:

— external_items;

— external_relationships;

— items;

— relationships.

4.2.117.1 external_items

The external_items specifies the Item objects belonging externally to an Item_structure. The external_items need not be specified for a particular Item_structure. There may be more than one external_items for an Item_structure. See 4.3.72 for the application assertion.

4.2.117.2 external_relationships

The external_relationships specifies the external relationships between the Item objects belonging to an Item_structure. The external_relationships need not be specified for a particular Item_structure. There may be more than one external_relationships for an Item_structure. See 4.3.72 for the application assertion.

4.2.117.3 items

The items specifies the Item objects belonging locally to an Item_structure. The items need not be specified for a particular Item_structure. There may be more than one items for an Item_structure. See 4.3.73 for the application assertion.

4.2.117.4 relationships

The relationships specifies the local relationships between the Item objects belonging to an Item_-structure. The relationships need not be specified for a particular Item_structure. There may be more than one relationships for an Item_structure. See 4.3.74 for the application assertion.

4.2.118 Library_definition

A Library_definition is a type of Definition (see 4.2.56) that is the basis for all types of library definitions. It provides the capability to reference a Definition from a library source.

The data associated with a Library_definition are the following:

— source.

4.2.118.1 source

The source specifies the reference to the Definition in a library. See 4.3.75 for the application assertion.

4.2.119 Library_design_definition

A Library_design_definition is a type of both Design_Definition (see 4.2.58) and Library_definition (see 4.2.118) that is a definition specified within a library and that is concerned with the life-cycle phase of design.

4.2.120 Library_element_reference

A Library_element_reference provides the means to reference information about a class within a P-Lib dictionary. Each Library_element_reference may be a Library_profile_cross_section (see 4.2.122).

The data associated with a Library_element_reference are the following:

— library_identifier;

— property_value_pairs.

4.2.120.1 library_identifier

The library_identifier specifies the identification of the component within a parts library as defined by ISO-13584. See 4.3.76 for the application assertion.

4.2.120.2 property_value_pairs

The property_value_pairs specifies the set of pairs property bsu and Property_value, defining the properties of the class. The property_value_pairs need not be specified for a particular Library_-element_reference. There may be more than one property_value_pairs for a Library_element_-reference. See 4.3.77 for the application assertion.

4.2.121 Library_manufacturing_definition

A Library_manufacturing_definition is a type of both Manufacturing_definition (see 4.2.131) and Library_definition (see 4.2.118) that is a definition specified within a library and that is concerned with the life-cycle phase of manufacturing.

4.2.122 Library_profile_cross_section

A Library_profile_cross_section is a type of Profile_cross_section (see 4.2.174) and Library_-element_reference (see 4.2.120) that is the reference to a standard profile cross section standardized in an ISO 13584 library.

4.2.123 Lightship_definition

A Lightship_definition is a type of Definition (see 4.2.56) that specifies the weight of the hull structure of the Ship, including the weight to any installed machinery and outfitting, but excluding the weight of the crew, any passengers and cargoes.

The data associated with a Lightship_definition are the following:

— defined_for;

— lightship_centre_of_gravity;

— lightship_items;

— lightship_weight.

4.2.123.1 defined_for

The defined_for specifies the redefinition of the defined_for attribute; a Lightship_definition is only valid for types of Ship objects. See 4.3.80 for the application assertion.

4.2.123.2 lightship_centre_of_gravity

The lightship_centre_of_gravity specifies the centre of gravity of the lightweight in the global coordinate system of the Ship. See 4.3.78 for the application assertion.

4.2.123.3 lightship_items

The lightship_items specifies the components which make up the lightweight definition. The lightship_items need not be specified for a particular Lightship_definition. There may be more than one lightship_items for a Lightship_definition. See 4.3.79 for the application assertion.

4.2.123.4 lightship_weight

The lightship_weight specifies the lightweight of the ship expressed in units of mass.

4.2.124 Lightship_weight_item

A Lightship_weight_item is a type of Weight_and_centre_of_gravity (see 4.2.269) that identifies the component that is a part of the total lightship weight. It may included the hull structure of the Ship, machinery or outfitting, but does not include cargo, crew, or passengers.

The data associated with a Lightship_weight_item are the following:

— aft_weight_extent;

— fwd_weight_extent;

— lightship_item_description.

4.2.124.1 aft_weight_extent

The aft_weight_extent specifies the length ordinate in the local coordinate system of the Lightship_-weight_ item identifying the aft extent.

4.2.124.2 fwd_weight_extent

The fwd_weight_extent specifies the length ordinate in the local coordinate system of the Lightship_-weight_item identifying the forward extent.

4.2.124.3 lightship_item_description

The lightship_item_description specifies the descriptive label of the Lightship_weight_item.

4.2.125 Loadline

A Loadline is a mark placed on the side of the ship in the form of a ring of 300mm outside diameter and 25mm wide, intersected by a horizontal line 450mm long and 25mm wide. The upper edge of this line passes through the centre of the ring. The ring is positioned at midships and at a distance below the upper edge of the deck line which corresponds to the assigned minumum freeboard. This value may not be less than 50mm. A series of load line marks are situated forward of this mark and these denote the minimum freeboards within certain geographical zones or in fresh water. The summer load line is level with the centre of the ring and marked S.

The data associated with a Loadline are the following:

— load_line_block_coefficient;

— load_line_depth;

— load_line_displacement;

— load_line_draught;

— load_line_length;

— load_line_regulation.

4.2.125.1 load_line_block_coefficient

The load_line_block_coefficient specifies the ratio measure obtained when the load_line_-displacement is divided by the product of load_line_length, load_line_depth and the moulded breadth of the ship.

4.2.125.2 load_line_depth

The load_line_depth specifies the least moulded depth of the ship.

4.2.125.3 load_line_displacement

The load_line_displacement specifies the moulded displacement of the ship measured at the load_-line_draught.

4.2.125.4 load_line_draught

The load_line_draught specifies the draught corresponding to 0.85 times the load_line_depth of the ship

4.2.125.5 load_line_length

The load_line_length specifies the length of the ship measured in accordance with the applicable IMO load line convention.

4.2.125.6 load_line_regulation

The load_line_regulation specifies the relevant IMO load line convention containing the rules to which the load line has been calculated..

The value of the load_line_regulation is one of the following:

— illc_1930;

— illc_1966;

— other.

NOTE See 4.2.125.6.1 - 4.2.125.6.3 for the definition of each allowable value for load_line_regulation.

4.2.125.6.1 illc_1930

illc_1930: the 1930 International loadline convention.

4.2.125.6.2 illc_1966

illc_1966: the 1966 International loadline convention.

4.2.125.6.3 other

other: another set of regulations for the calculation of the load lines.

4.2.126 Local_co_ordinate_system

A Local_co_ordinate_system is a type of Definition (see 4.2.56) that is used to locate an object in space. A Local_co_ordinate_system is always defined with respect to another coordinate system, this might be the Global_axis_placement or another Local_co_ordinate_system that is a member in the same hierarchy. Each Local_co_ordinate_system may be a Local_co_ordinate_system_with_-position_reference (see 4.2.127).

EXAMPLE Figure 28 illustrates a local coordinate system.

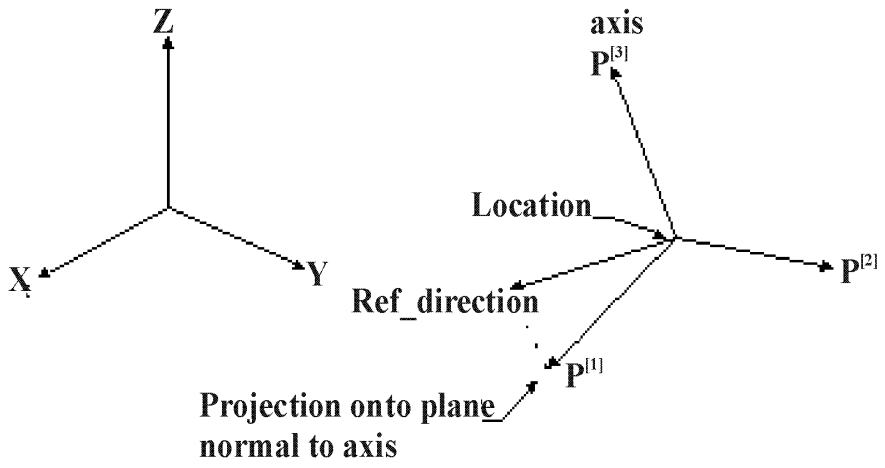


Figure 28 — Local coordinate system

NOTE 1 Local axes and origin are handled in the same way as for axis2 placement 3d.

NOTE 2 A local coordinate system is from always a right-handed system.

The data associated with a Local_co_ordinate_system are the following:

— parent.

4.2.126.1 parent

The parent specifies the underlying coordinate system which serves as definition space for the current coordinate system. The parent need not be specified for a particular Local_co_ordinate_system. There may be more than one parent for a Local_co_ordinate_system. See 4.3.81, 4.3.82, 4.3.82 for the application assertions.

4.2.127 Local_co_ordinate_system_with_position_reference

A Local_co_ordinate_system_with_position_reference is a type of Local_co_ordinate_system (see 4.2.126) that directly refers to the unique Global_axis_placement as its parent. Its location is defined by references to longitudinal, vertical or transversal frames, possibly using an additional offset value-a distance. Alternatively absolute coordinates may be specified. Also, combinations of coordinates and references are valid. A Local_co_ordinate_system_with_position_reference is not to specify rotations as transformations to the global system, that is its axes are required to be parallel to the axes of the Global_axis_placement.

The data associated with a Local_co_ordinate_system_with_position_reference are the following:

— longitudinal_ref;

— transversal_ref;

— vertical_ref.

4.2.127.1 longitudinal_ref

The longitudinal_ref specifies a Longitudinal_position, possibly with an offset value, or an absolute coordinate value, along the longitudinal axis of the global coordinate system. See 4.3.83 for the application assertion.

4.2.127.2 transversal_ref

The transversal_ref specifies a Transversal_position, possibly with an offset value, or an absolute coordinate value, along the transversal axis of the global coordinate system. See 4.3.83 for the application assertion.

4.2.127.3 vertical_ref

The vertical_ref specifies a Vertical_position, possibly with an offset value, or an absolute coordinate value, along the vertical axis of the global coordinate system. See 4.3.83 for the application assertion.

4.2.128 Logical_property_value

A Logical_property_value is a type of Property_value (see 4.2.184) that provides a value for a property of type logical.

The data associated with a Logical_property_value are the following:

— val.

4.2.128.1 val

The val specifies the value of type logical.

4.2.129 Longitudinal_position

A Longitudinal_position is a type of Spacing_position (see 4.2.209) that is located on the global X-axis.

4.2.130 Longitudinal_table

A Longitudinal_table is a type of Spacing_table (see 4.2.211) that has positions that are along the longitudinal axis of the global coordinate system that is the global X-axis. Each Longitudinal_-position may be one of the following: a Frame_table (see 4.2.93), or a Station_table (see 4.2.214).

The data associated with a Longitudinal_table are the following:

— spacing_table_representations.

4.2.130.1 spacing_table_representations

The spacing_table_representations specifies the longitudinal positions that make up the longitudinal table. The spacing_table_representations need not be specified for a particular Longitudinal_table. There may be more than one spacing_table_representations for a Longitudinal_table. See 4.3.84 for the application assertion.

4.2.131 Manufacturing_definition

A Manufacturing_definition is a type of Definition (see 4.2.56) that is the basis for all types of manufacturing definitions. The ability to reference representations differentiates a Manufacturing_-definition from a Definition. Each Manufacturing_definition is either an Assembly_manufacturing_-definition (see 4.2.10), a Library_manufacturing_definition (see 4.2.121), a Structural_part_-manufacturing_definition (see 4.2.235), or Weld_manufacturing_definition (see 4.2.273).

The data associated with a Manufacturing_definition are the following:

— representations.

4.2.131.1 representations

The representations specifies the representations of the design definition. It is possible for a Manufacturing_definition to have multiple representations. Representations may have no value contained in it or it may have one of more values, without any duplications.

4.2.132 Maximum_permissible_still_water_bending_moment

A Maximum_permissible_still_water_bending_moment is a type of Bending_moment (see 4.2.14) that is the maximum valid bending moments for a ship under still water conditions as assigned by the classification society. These values will change for short voyages and sheltered water conditions. However, these can be derived from the sea-going conditions and so are not necessary for a model used for exchange purposes.

The data associated with a Maximum_permissible_still_water_bending_moment are the following:

— table_of_max_values.

4.2.132.1 table_of_max_values

The table_of_max_values specifies the set of maximum hogging and sagging moments at specified positions along the ship. There may be more than one table_of_max_values for a Maximum_-permissible_still_water_bending_moment. See 4.3.85 for the application assertion.

4.2.133 Maximum_permissible_still_water_shear_force

A Maximum_permissible_still_water_shear_force is a type of Shear_force (see 4.2.203) that is the maximum valid shear force for a ship under still water conditions as assigned by the classification society. These values will change for short voyages and sheltered water conditions. However, these can be derived from the sea-going conditions and so are not necessary for a model used for exchange purposes.

The data associated with a Maximum_permissible_still_water_shear_force are the following:

— table_of_max_shear_force_values.

4.2.133.1 table_of_max_shear_force_values

The table_of_max_shear_force_values specifies the set of values that defines the maximum permissible shear force distribution along the ship. There may be more than one table_of_max_-shear_force_values for a Maximum_permissible_still_water_shear_force. See 4.3.86 for the application assertion.

4.2.134 Maximum_permissible_swbm_values

The Maximum_permissible_swbm_values are maximum values for hogging and sagging moments at specified positions along the ship. The values are to represent classification society requirements.

The data associated with a Maximum_permissible_swbm_values are the following:

— hogging;

— position;

— sagging.

4.2.134.1 hogging

The hogging specifies the maximum hogging value.

4.2.134.2 position

The position specifies the distance from the aft-perpendicular from where the sagging and hogging moments are taken, given as a frame or station position. See 4.3.87 for the application assertion.

4.2.134.3 sagging

The sagging specifies the maximum sagging value.

4.2.135 Maximum_permissible_swsf_values

The Maximum_permissible_swsf_values are maximum values for positive and negative shear force at specified positions along the ship. The values are to represent classification society requirements.

The data associated with a Maximum_permissible_swsf_values are the following:

— negative_value;

— position;

— positive_value.

4.2.135.1 negative_value

The negative_value specifies the maximum negative shear force value that contributes to overall shear.

4.2.135.2 position

The position specifies the distance from the aft-perpendicular from where the shear forces are taken, given as a frame or station position. See 4.3.88 for the application assertion.

4.2.135.3 positive_value

The positive_value specifies the maximum positive shear force value that contributes to overall shear.

4.2.136 Moment_3d

A Moment_3d is a collection of moment components at three major coordinate system axes, vertical, transverse and longitudinal.

The data associated with a Moment_3d are the following:

— longitudinal_moment;

— origin;

— transverse_moment;

— vertical_moment.

4.2.136.1 longitudinal_moment

The longitudinal_moment specifies the moment component along the longitudinal axis.

4.2.136.2 origin

The origin specifies the point in the local coordinate system about which the moment component is defined. See 4.3.89 for the application assertion.

4.2.136.3 transverse_moment

The transverse_moment specifies the moment component along the transverse axis.

4.2.136.4 vertical_moment

The vertical_moment specifies the moment component along the vertical axis.

4.2.137 Named_unit

A Named_unit is a pre-defined unit object specified in ISO 10303-41.

4.2.138 Navy_ship

A Navy_ship is a type of Shiptype (see 4.2.207) that is a Ship operating under military command.

The data associated with a Navy_ship are the following:

— has_type.

4.2.138.1 has_type

The has_type specifies the type of Navy_ship.

The value of the has_type is one of the following:

— aircraft_carrier;

— auxiliary_oiler;

— corvette;

— cruiser;

— destroyer;

— fleet_auxiliary_vessel;

— frigate;

— landing_platform_dock;

— landing_platform_helicopter;

— mine_warfare_ship;

— patrol_force_vessel;

— service_craft;

— submarine;

— user_defined;

NOTE See 4.2.138.1.1 - 4.2.138.1.14 for the definition of each allowable value for has_type.

4.2.138.1.1 aircraft_carrier

a Navy_ship that, apart from providing a Command, Control and Communications facility, is to deploy aircraft and or helicopters in sustained anti-submarine warfare operations and fighter protection, with full reconnaissance and strike capability.

EXAMPLE Deployed aircraft may include short take off and vertical landing (STOVAL), short take off but arrested recovery (STOBAR), or conventional take off (CTOL).

4.2.138.1.2 auxiliary_oiler

a ship specially designed to replenish ships at sea with liquids during world-wide operations, with vertical-replenishment (VERTREP) services for the transfer of solids also being provided.

NOTE The Ship has a stable platform suitable for helicopters, including stowage and maintenance facilities plus emergency landing of other helicopters.

4.2.138.1.3 corvette

corvette: a Navy_ship specifically constructed for the purpose of performing the role of a small escort vessel to a task group. Corvettes are fitted primarily to fulfill an anti-submarine warfare role.

4.2.138.1.4 cruiser

cruiser: a Navy_ship typically used to screen carrier task forces. Cruisers, many with guided missiles or carrying a helicopter provide anti-air warfare and anti-submarine capabilities. Cruisers also provide protection against anti-ship cruise missile threats at extended ranges, particularly in the presence of enemy electronic countermeasures.

4.2.138.1.5 destroyer

destroyer: a Navy_ship that is a major surface combatant that is typically used to conduct operations with strike, antisubmarine warfare and amphibious forces, and to perform screening and convoy duties. A destroyer may also be equipped with helicopters, providing an enhanced capability.

4.2.138.1.6 fleet_auxiliary_vessel

fleet_auxiliary_vessel: a Navy_ship with the primary task of supplying warships at sea with fuel, food, stores and ammunition. An auxiliary also provides aviation platforms, amphibious support for the navy and marines and sea transport for army units. There are many types of fleet auxiliary vessels.

4.2.138.1.7 frigate

frigate: a Navy_ship that can be generally described as a general purpose, ocean escort vessel. Operational requirements necessitate that frigates perform the duties of area defense ships, capable of defending a task group against modern day air threats. Frigates may also provide command facilities and accommodation. Secondary capabilities include anti-surface warfare, naval gunfire support and anti-submarine warfare.

4.2.138.1.8 landing_platform_dock

landing_platform_dock: a Navy_ship specifically designed to transport a large embarked military force plus support equipment across open oceans; support a flexible landing on hostile shores using onboard helicopters and landing craft, and coordinate the naval, air and land aspects of amphibious operations through command, control and communications_facilities.

4.2.138.1.9 landing_platform_helicopter

landing_platform_helicopter: a Navy_ship that serves as an amphibious helicopter carrier designed to enable the rapid tactical deployment of airborne troops and equipment to spearhead amphibious operations ashore. It can also stand off a coast at strategic range in a deterrent operational role. Peacetime roles include troop and equipment transport, and humanitarian tasks such as disaster relief.

4.2.138.1.10 mine_warfare_ship

mine_warfare_ship: a mine countermeasures Navy_ship (MCMV) that is specially constructed for the hunting, sweeping by mechanical, acoustic or magnetic means, and clearance of mines in both inshore, coastal and deep, exposed waters. Designed to have minimal magnetic signature, MCMV hulls are generally constructed of low-magnetic steel, GRP or laminated wood.

4.2.138.1.11 patrol_force_vessel

patrol_force_vessel: a Navy_ship that can vary in size according to the roles for which they have been designed. Larger offshore patrol vessels (OPVs) are used for firefighting, rescue or supply tasks, others are used as protection vessels, sometimes carrying a marine detachment and semi-rigid craft to act as a rapid response squadron. Some OPVs have ice-strengthened hulls and a helicopter landing deck for operation as survey vessels in the arctic or antarctic regions. The size, maneuverability and other operational characteristics of smaller coastal patrol craft indicate that they

are ideally suited for patrol, search and rescue duties in coastal areas, or ports, harbors and other restricted waters.

4.2.138.1.12 service_craft

service_craft: typically a small Navy_ship that has been specially designed to provide services to a fleet in harbors and ports. These include tugs, tenders, barges, patrol craft, recovery vessels, floating docks, etc. Other service craft providing support are larger ocean-going vessels, such as transports, survey and research ships, repair vessels, cargo ships, hospital ships, etc.

4.2.138.1.13 submarine

submarine: a Navy_ship that is a high-capability, ocean-going vessel designed typically to perform both anti-submarine and anti-surface warship tasks. To perform these functions, generally the weapon system of a submarine is capable of stowing, firing and controlling heavyweight torpedoes, underwater-to-surface guided weapons or submarine-laid mines. Submarines are able to operate submerged in shallow waters as well as the open ocean, whilst also being capable of operation environmentally in waters from the tropics to the arctic.

4.2.138.1.14 user_defined

user_defined: can be any Shiptype that is not in the enumeration list. Details should be found in the description attribute for Shiptype.

4.2.139 Non_circular_hollow_profile_cross_section

A Non_circular_hollow_profile_cross_section is a type of Parametric_profile_cross_section (see 4.2.151) identified by having a hollow core and closed perimeter. Each Non_circular_hollow_-profile_cross_section is a Square_tube_cross_section (see 4.2.213).

NOTE ISO 657-14 [5] specifies dimensions for some standard square and rectangular hollow sections.

4.2.140 Non_manifold_surface_shape

A Non_manifold_surface_shape is a shape representation whch conforms to 10303-508.

4.2.141 Number_property_value

A Number_property_value is a type of Property_value (see 4.2.184) that provides a value for a property that is a numeric value.

The data associated with a Number_property value are the following:

— val.

4.2.141.1 val

The val specifies a numeric value.

4.2.142 Outward_round_corner_design_definition

An Outward_round_corner_design_definition is a type of Round_corner_design_definition (see 4.2.195) that specifies the radius and location the centre of the circular arc representing the outward round corner.

NOTE Figure 29 illustrates an Outward_round_corner_cutout.

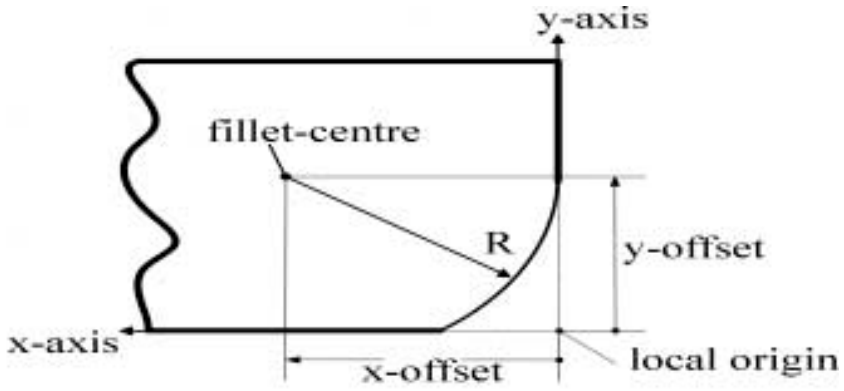


Figure 29 — Outward_round_corner_cutout

The data associated with an Outward_round_corner_design_definition are the following:

— x_offset;

— y_ offset.

4.2.142.1 x_offset

The x_offset specifies the distance of the centre of the circular arc from the origin of the local coordinate system in positive x-direction.

4.2.142.2 y_offset

The y_offset specifies the distance of the centre of the circular arc from the origin of the local coordinate system in positive y-direction.

4.2.143 Owner_designation

An Owner_designation is a type of General_characteristics_definition (see 4.2.97) that specifies the organizations that order, own and manage the Ship.

The data associated with an Owner_designation are the following:

— local_units;

— managing_company;

— ordering_company;

— owner_approval;

— owning_company.

4.2.143.1 local_units

The local_units specifies that an Owner_designation is not to define local units.

4.2.143.2 managing_company

The managing_company specifies the organization that is responsible for managing and operating the Ship.

4.2.143.3 ordering_company

The ordering_company specifies the organization that ordered the Ship at a shipyard.

4.2.143.4 owner_approval

The owner_approval specifies an indication that the ship owner has approved the design of the ship.

4.2.143.5 owning_company

The owning_company specifies the organization that legally owns the Ship.

4.2.144 Panel_system

A Panel_system is a type of Structural_system (see 4.2.239) that is a special section that may contain zero or many Plate objects (see 4.2.155) reinforced by Profile objects running in the same main direction; the Plate objects lie on the same planar or non-planar moulded surface. Item structure containing structural parts relating to panel system does not need to be populated, but if it is, at least one of the structural parts must be a plate.

The data associated with a Panel_system are the following:

— the_class.

4.2.144.1 the_class

The the_class specifies the class of a Panel_system.

The value of the_class is one of the following:

— panel_system_class.

NOTE See 4.2.144.1.1 for the definition of each allowable value for the_class.

4.2.144.1.1 panel_system_class

panel_system_class: special section that consists of zero to many Plate objects reinforced by Profile objects running in the same main direction; the Plate objects lie on the same planar or non-planar moulded surface.

4.2.145 Panel_system_boundary_relationship

A Panel_system_boundary_relationship is a type of Panel_system_relationship (see 4.2.150) that defines a boundary of a Panel_system (see 4.2.144) by referring to a structural item.

Structural item can be one of the following:

— Panel_system - the boundary is created by the intersection of the implicitly or explicitly given moulded surfaces of the two Panel_systems;

— Profile - the trace line of the Profile must either be parallel or orthogonal to the implicitly or explicitly given moulded surface of Panel_system_boundary_relationship. In the first case the boundary is created by the intersection of the Profile top line with the moulded surface of Panel_system_boundary_relationship. In the second case it is created by the web of the cross section of the Profile;

— Seam - the Seam is the boundary.

The data associated with a Panel_system_boundary_relationship are the following:

— displacement.

4.2.145.1 displacement

The displacement specifies the offset orthogonally off the referenced boundary. The displacement need not be specified for a particular Panel_system_boundary_relationship.

4.2.146 Panel_system_curve_boundary

A Panel_system_curve_boundary is a simple geometric boundary for a Panel_system (see 4.2.144).

The data associated with a Panel_system_curve_boundary are the following:

— curve;

— displacement.

4.2.146.1 curve

The curve defines the boundary geometry.

4.2.146.2 displacement

The displacement specifies the offset of the boundary orthogonal off the curve. The displacement need not be specified for a particular Panel_system_curve_ boundary.

4.2.147 Panel_system_design_definition

A Panel_system_design_definition is a type of Structural_system_design_definition (see 4.2.241) that describes the definition of a Panel_system (see 4.2.144) for the life-cycle stage preliminary design.

The design of the panel_system plating is defined parametrically by a basis surface, implicitly defined by moulded_surface and border, which is then offset, based on material_offset and thickness, to form the two major faces of the panel_system. The basis surface is represented implicitly as a bounded_surface defined by moulded_surface and border. The resulting plate faces are implicitly represented as offset surfaces from the basis_surface. The offset surface distances are (material_offset multiplied by thickness) and ((material_offset – 1) multiplied by thickness).

NOTE The major faces of the panel_system are parallel, separated by the given thickness. Since every point on the offset surface is a fixed distance from the basis_surface along the surface normal, the edge surfaces of the panel_system are always normal to both major faces. If the basis_surface is planar and rectangular, then the representation of the resulting panel_system shape is a rectangular parallelepiped.

The data associated with a Panel_system_design_definition are the following:

— border;

— defined_for;

— material_offset;

— moulded_surface;

— stiffener_spacing;

— thickness.

4.2.147.1 border

The border of a Panel_system is made of a number of boundaries. The order of the boundaries is significant and therefore should be preserved by the mapping. However there is no rule that requires end point of boundary[n] = start point of boundary[n+1]. Furthermore two boundaries may intersect or it might be necessary to extend one or both boundaries in order to make them intersecting. A boundary can be one of the following:

— a curve parallel to a given curve with a given displacement;

— a line resulting from the intersection of the Panel_system's x-y-plane with a given plane;

— a line resulting from the intersection of the Panel_system's x-y-plane with the x-y-plane of a given Panel_system;

— a curve parallel to a boundary of a given Panel_system with a given displacement;

— a curve parallel to the top line of a given Profile with a given displacement;

— a line parallel to the mould line of the web of the cross section of a given Profile with a given displacement;

— a seam.

The border need not be specified for a particular Panel_system_design_definition. There may be more than one border for a Panel_system_design_definition.

See 4.3.93, 4.3.94, 4.3.95, 4.3.96 for the application assertions.

4.2.147.2 defined_for

The defined_for specifies the Panel_system objects that are defined by the Panel_system_design_-definition. There may be more than one defined_for for a Panel_system_design_definition. See 4.3.92 for the application assertion.

4.2.147.3 material_offset

The material_offset specifies the location of the panel material with respect to the moulded_surface implicitly or explicitly given in fractions of the panel thickness. If the material_offset is between 0 and 1, then : material_offset multiplied by 100% defines the percentage of the material located on the positive side of the moulded surface, where positive indicates in the direction of the moulded surface normal. If the material_offset is greater than 1, then the panel material is located ((material_thickness – 1) multiplied by thickness) above the moulded_surface on the positive side. If the material_offset is negative, then the panel material is located (material_thickness multiplied by thickness) below the moulded_surface on the negative side.

EXAMPLE 1 material_offset < 0: material located completely below the negative side of the moulded surface.

EXAMPLE 2 material_offset = 0: material located completely on the negative side of the moulded surface.

EXAMPLE 3 0 < material_offset < 1: material located in the moulded surface.

EXAMPLE 4 material_offset = 1 : material located completely on the positive side of the moulded surface.
EXAMPLE 5 material_offset > 1 : material located completely above the positive side of the moulded surface.

NOTE The material_offset specified here serves as a reference value for the material_offset of the Plate objects defined for the Panel_system referred to in a later stage of the design. For some reason it may happen that the material_offset of the Plate objects differs from the one specified here.

4.2.147.4 moulded_surface

The moulded_surface specifies the underlying moulded surface of the Panel_system. If not specified, the local x-y-plane is implicitly assumed as underlying moulded surface with normal along the Z-axis. The local x-y plane is defined by the local coordinate system of the panel system , if it exists,

otherwise, by the parent panel_system . The moulded_surface need not be specified for a particular Panel_system_design_definition. See 4.3.90 for the application assertion.

4.2.147.5 stiffener_spacing

The stiffener_spacing specifies the proposed spacing of the stiffeners belonging to the panel. The stiffener_spacing need not be specified for a particular Panel_system_design_definition.

NOTE 1 The stiffener_spacing is optional because not all Panel_system objects have a meaningful stiffener_-spacing.

NOTE 2 The stiffener_spacing specified here serves as a reference value for the spacing of the Profile objects defined on the Panel_system referred to in a later stage of the design. For some reason it may happen that the spacing of the Profile objects differs from the one specified here.

4.2.147.6 thickness

The thickness specifies the proposed material thickness of the Panel_system.

NOTE The thickness specified here serves as a reference value for the thickness of the Plate objects defined for the Panel_system referred to in a later stage of the design. For some reason it may happen that the thickness of the Plate objects differs from the one specified here.

4.2.148 Panel_system_functional_definition

A Panel_system_functional_definition is a type of Structural_system_functional_definition (see 4.2.242) that defines the function of a Panel_system (see 4.2.144).

The data associated with a Panel_system_functional_definition are the following:

— defined_for.

4.2.148.1 defined_for

The defined_for specifies the Panel_system objects that are defined by the Panel_system_-functional_definition. There may be more than one defined_for for a Panel_system_functional_-definition. See 4.3.97 for the application assertion.

4.2.149 Panel_system_plane_boundary

A Panel_system_plane_boundary specifies a boundary by intersecting implicitly or explicitly with the given moulded surface of a Panel_system (see 4.2.144).

The data associated with a Panel_system_plane_boundary are the following:

— plane.

4.2.149.1 plane

The plane specifies the boundary geometry by intersecting implicitly or explicitly with the given moulded surface of a Panel_system.

4.2.150 Panel_system_relationship

A Panel_system_relationship is a type of Structural_system_relationship (see 4.2.244) that describes the association of a Panel_system (see 4.2.144) with another structural item. Structural item can be one of the following:

— Structural_system;

— Structural_part;

— Structural_feature.

Each Panel_system_relationship is a Panel_system_boundary_relationship (see 4.2.145).

The data associated with a Panel_system_relationship are the following:

— item_1.

4.2.150.1 item_1

The item_1 specifies the Panel_system_relationship is to relate a Panel_system with something else. See 4.3.98 for the application assertion.

4.2.151 Parametric_profile_cross_section

A Parametric_profile_cross_section is a type of Profile_cross_section (see 4.2.174). A Parametric_-profile_cross_section is a standard shape that can be defined by specification of values for a set of pre-defined geometric parameters. Each Parametric_profile_cross_section may be one of the following: an Angle_profile_cross_section (see 4.2.5), a Bar_profile_cross_section (see 4.2.13), a Channel_profile_cross_section (see 4.2.32), a Flanged_profile_cross_section (see 4.2.90), a Non_-circular_hollow_profile_cross_section (see 4.2.139), or a Proprietary_profile_cross_section (see 4.2.185).

4.2.152 Part

A Part is a type of Item (see 4.2.115) that is the atomic element within a product structure representing a System (see 4.2.249), an Assembly (see 4.2.8) or a space. A Part is an Item created by a physical activity and made of a material.

4.2.153 Part_edge_cutout_design_definition

A Part_edge_cutout_design_definition is a type of Feature_design_definition (see 4.2.86) that specifies the removal of material along an edge due to the penetration of a part. The shape of the

opening is dependent on the part and therefore, not necessarily a symmetry axis. All measures of the Part_edge_cutout_design_definition are relative to the penetrating part, that is, they depend on the part ID-points at specific locations on the cross section of the part. The X-axis coincides with the part edge containing Part ID locations A and D.

NOTE 1 Figure 30 illustrates the parameters and shape of a Part_edge_cutout_design_definition.

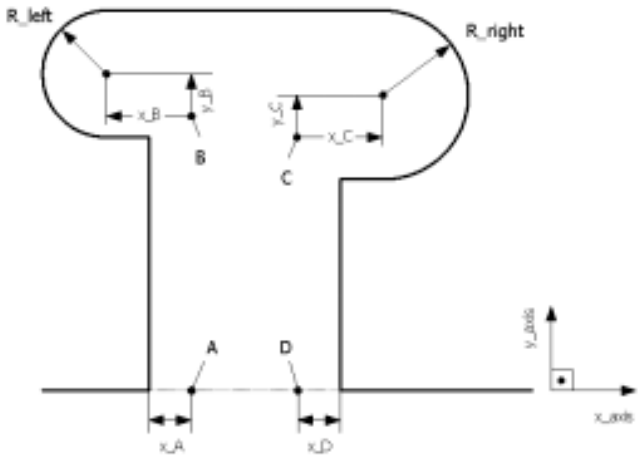


Figure 30 — Part edge cutout

The design of the part edge cutout is defined parametrically and the resulting shape is implicitly defined by a connected set of lines and circular arcs. The cutout depth and inner shape depend on the Part ID locations B and C. The width of the opening may depend on the part ID locations A, B, C and or D. The radii r_left and r_right, x_a and x_d are positive always, whereas x_b, y_b, x_c and y_c may be positive or negative. The center of each circular arcs is constrained to lay on or outside the opening. Hence, the following constraints apply:

$$A - x\_a \geq B - x\_b$$
$$D + x\_d \leq C + x\_c$$

NOTE 2 The connected set of lines and arcs is defined in the table below. P1 denotes a point on the left circular arc where the arc is tangent to the line containing Point P2. P2 denotes a point on the right circular arc where the arc is tangent to the line containing Point P1. If the line P1 to P2 is parallel to the local X-axis, then both circular arcs are half circles.

Start Point		End Point		Geometry
X	Y	X	Y	
A – x_a	0	A – x_a	B + y_b – r_left	Line
A – x_a	B + y_b – r_left	B – x_b	B + y_b – r_left	Line
B – x_b	B + y_b – r_left	P1	P1	Circular arc
P1	P1	P2	P2	Line
P2	P2	C + x_c	C + y_c – r_right	Circular arc
C + x_c	C + y_c – r_right	D + x_d	C + y_c – r_right	Line
D + x_d	C + y_c – r_right	D + x_d	0	Line

The Part ID locations are defined by the profile shape. Part ID location A and D are defined on a Profile_cross_section at v = 0 at the minimum and maximum u values respectively. Part ID location B is defined as the point with minimum u value and greatest v value, unless this coincides with Part ID location A. In this case, Part ID location B is the point on the left side (negative u direction) with maximum u value and greatest v value. Part ID location C is defined as the point with maximum u value and greatest v value, unless this coincides with Part ID location D. In this case, Part ID location C is the point on the right side (negative u direction) with minimum u value and greatest v value.

EXAMPLE 1 The Part ID locations for different types of Profile_cross_section are shown in Figure 31.

EXAMPLE 2 Examples of different part edge cutout shapes are shown in Figure 32.

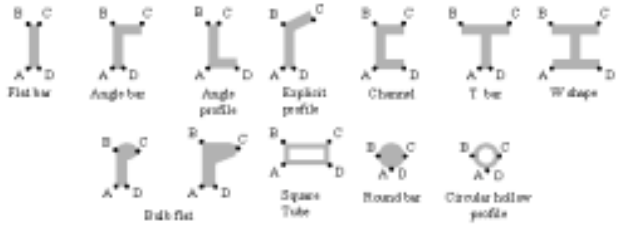


Figure 31 — Part ID locations for various Profile_cross_sections

Figure 32 — Examples of part edge cutout shapes

The data associated with a Part_edge_cutout_design_definition are the following:

— defined_for;

— r_left;

— r_right;

— x_a;

— x_b;

— x_c;

— x_d;

— y_b;

— y_c.

4.2.153.1 defined_for

The defined_for specifies the Edge_cutout objects that are defined by the Part_edge_cutout_design_-definition. There may be more than one defined_for for a Part_edge_cutout_design_definition. See 4.3.99 for the application assertion.

4.2.153.2 r_left

The r_left specifies the radius of a left circular arc that determines the round shape of the part edge cutout in the negative X direction. The origin of the left circular arc is offset (x_b, y_b) from Part ID location.

4.2.153.3 r_right

The r_right specifies the radius of a right circular arc that determines the round shape of the part edge cutout in the positive X direction. The origin of the right circular arc is offset (x_c, y_c) from Part ID location C.

4.2.153.4 x_a

The x_a specifies the offset from Part ID location A in the negative X direction.

4.2.153.5 x_b

The x_b specifies the center of the left circular arc as an offset from Part ID location B in the negative X direction.

4.2.153.6 x_c

The x_c specifies the center of the right circular arc as an offset from Part ID location C in the positive X direction.

4.2.153.7 x_d

The x_d specifies the offset from Part ID location D in the positive X direction.

4.2.153.8 y_b

The y_b specifies the center of the left circular arc as an offset from Part ID location B in the positive Y direction.

4.2.153.9 y_c

The y_c specifies the center of the right circular arc as an offset from Part ID location C in the positive Y direction.

4.2.154 Part_relationship

A Part_relationship is a type of Item_relationship (see 4.2.116) that defines the association of a Part (see 4.2.152) with another Item (see 4.2.115).

The data associated with a Part_relationship are the following:

— item_1.

4.2.154.1 item_1

The item_1 specifies the relating Part of the relationship. See 4.3.100 for the application assertion.

4.2.155 Plate

A Plate is a type of Structural_part (see 4.2.229) that forms the primary physical structure of a ship. Plate parts are manufactured from purchased flat stock material of standard thicknesses, and are welded, cut, drilled, bent, rolled, or otherwise formed to produce the resulting part for installation in the ship.

4.2.156 Plate_approval

A Plate_approval is a type of Structural_class_approval (see 4.2.219) that confirms the approval of a Plate_design_definition (see 4.2.158).

The data associated with a Plate_approval are the following:

— subject.

4.2.156.1 subject

The subject specifies the approved Plate_design_definition. See 4.3.101 for the application assertion.

4.2.157 Plate_boundary_relationship

A Plate_boundary_relationship is a type of Plate_relationship (see 4.2.162) that defines a boundary of a Plate (see 4.2.155) by referring to a Seam (see 4.2.198).

The data associated with a Plate_boundary_relationship are the following:

— item_2.

4.2.157.1 item_2

The item_2 specifies the binding Item is to be of type Seam. See 4.3.102 for the application assertion.

4.2.158 Plate_design_definition

The Plate_design_definition is a type of Structural_part_design_definition (see 4.2.231) that describes the definition of a Plate (see 4.2.155) from the view of detailed design.

The design of the plate is defined parametrically by a basis surface, implicitly defined by moulded_surface and border, which is then offset, based on material_offset and thickness, to form the two major faces of the plate. The basis surface is represented implicitly as a bounded_surface defined by moulded_surface and border. The resulting plate faces are implicitly represented as offset surfaces from the basis_surface. The offset surface distances are (material_offset multiplied by thickness) and ((material_offset – 1) multiplied by thickness).

NOTE The major faces of the plate are parallel, separated by the given thickness. Since every point on the offset_surface is a fixed distance from the basis_surface along the surface normal, the edge surfaces of the plate are always normal to both major faces. If the basis_surface is planar and rectangular, then the representation of the resulting plate shape is a rectangular parallelepiped.

The data associated with a Plate_design_definition are the following:

— border;

— defined_for;

— interior_point;

— material_offset;

— moulded_surface;

— thickness.

4.2.158.1 border

The border of a Plate (see 4.2.155) is made of a number of boundaries. The order of the boundaries is significant and therefore should be preserved by the mapping. However there is no rule that requires end point of boundary[n] = start point of boundary[n+1]. Furthermore two boundaries may intersect or it might be necessary to extend one or both boundaries in order to make them intersecting. A boundary can be one of a curve, a Seam belonging to the Panel_system the Plate belongs to, and a boundary of a Panel_system the Plate belongs to.

The border need not be specified for a particular Plate_design_definition. There may be more than one border for a Plate_design_definition.

See 4.3.105, 4.3.106, 4.3.107 for the application assertions.

4.2.158.2 defined_for

The defined_for specifies the Plate objects that are defined by the Plate_design_definition. There may be more than one defined_for for a Plate_design_definition. See 4.3.109 for the application assertion.

4.2.158.3 interior_point

The interior_point specifies a point within the boundaries of the Plate_design_definition, located on the moulded surface. The interior_point need not be specified for a particular Plate_design_definition. The interior_point is to be specified for any Plate_design_definition for which the resolution of the border attribute references does not constitute a single unambiguous plate.

4.2.158.4 material_offset

The material_offset specifies the location of the plate material with respect to the moulded surface, implicitly or explicitly given, in fractions of the Plate thickness. If the material_offset is between 0 and 1, then material_offset multiplied by 100% defines the percentage of the material located on the positive side of the moulded surface, where positive indicates in the direction of the moulded surface. If the material_offset is greater than 1, then the plate material is located ((material_thickness – 1) multiplied by thickness) above the moulded_surface on the positive side. If the material_offset is negative, then the plate material is located (material_thickness multiplied by thickness) below the moulded_surface on the negative side.

EXAMPLE 1 material_offset < 0: material located completely below the negative side of the moulded surface.

EXAMPLE 2 material_offset = 0: material located completely on the negative side of the moulded surface.

EXAMPLE 3 0 < material_offset < 1: material located in the moulded surface.

EXAMPLE 4 material_offset = 1: material located completely on the positive side of the moulded surface.

EXAMPLE 5 material_offset > 1: material located completely above the positive side of the moulded surface.

NOTE If the Plate belongs to a Structural_system that has a Design_definition during the preliminary design stage, like Panel_system, material_offset should match with the one specified there. For some reason however it might be necessary that a Plate gets a material_offset other than specified during preliminary design.

4.2.158.5 moulded_surface

The moulded_surface specifies the underlying moulded surface of the Plate. If the moulded_surface is not specified and the Plate_design_definition participates in a Panel_system, the underlying moulded surface of the Panel_system is taken into account, otherwise the local x-y-plane of the plate is implicitly assumed as underlying moulded surface with normal along the Z-axis. The local x-y plane is defined by the local coordinate system of the plate , if it exists, otherwise, by the parent panel_system . The moulded_surface need not be specified for a particular Plate_design definition. See 4.3.103 for the application assertion.

4.2.158.6 thickness

The thickness specifies the material thickness of the plate.

NOTE If the Plate belongs to a Structural_system that got a Design_definition during the preliminary design stage, like Panel_system, thickness should match with the one specified there. For some reason however, it might be necessary that a Plate gets a thickness other than specified during preliminary design.

4.2.159 Plate_functional_definition

A Plate_functional_definition is a type of Structural_part_functional_definition (see 4.2.232) that defines the functional role of a Plate; the role may be a pre-defined one or may be user-defined.

The data associated with a Plate_functional_definition are the following:

— defined_for.

4.2.159.1 defined_for

The defined_for specifies one or multiple Plate objects that are defined by the Plate_functional_-definition. There may be more than one defined_for for a Plate_functional_definition. See 4.3.111 for the application assertion.

4.2.160 Plate_manufacturing_definition

A Plate_manufacturing_definition is a type of Structural_part_manufacturing_definition (see 4.2.235) that is a production life cycle stage view of a plate part that represents characteristics and attributes directly related to the manufacturing of the part. An earlier detailed design part stage view may or may not exist for the part. At this stage, structural part features are identified that affect aspects of the part that have to do with processes that will be used to cut and form the part. Some examples of the kinds of features identified during this stage are edge grinds, layout marks, and part stock. A 2D or 3D representation may be defined for the manufacturing definition of the part, enabling it to be nested and cut from flat or corrugated raw material stock. This geometry is represented by the inherited representation attribute.

The data associated with a Plate_manufacturing_definition are the following:

— outer_contour.

4.2.160.1 outer_contour

The outer_contour specifies the edge based wireframe model that specifies the outer boundary of the part geometry that will be burned during the manufacturing process. This is the resulting model after the resolution of manufacturing features such as added material feature and weld shrinkage allowance.

4.2.161 Plate_rejection

The Plate_rejection is a type of Structural_class_rejection (see 4.2.222) that confirms that a Plate_-design_definition (see 4.2.158) is not approved. The reasons for rejection and existing requirements for the Plate are specified.

NOTE Plate_rejection may specify additional information clarifying the rule requirements for the Plate. A yard has several options for action; the thickness of the plate may be increased, the material quality may be improved, another stiffening may be applied. Although it is not mandatory, a class society may suggest improvements in these areas. This may then be done using Structural_class_rejection.proposed_alternative.

The data associated with a Plate_rejection are the following:

— subject.

4.2.161.1 subject

The subject specifies the rejected Plate_design_definition. See 4.3.112 for the application assertion.

4.2.162 Plate_relationship

A Plate_relationship is a type of Structural_part_relationship (see 4.2.237) that describes the relationship between a Plate (see 4.2.155) and another structural item. Structural item can be one of the following:
Structural_system (see 4.2.239), Structural_part (see 4.2.229), or Structural_feature (see 4.2.226).

Each Plate_relationship is a Plate_boundary_relationship (see 4.2.157).

The data associated with a Plate_relationship are the following:

— item_1.

4.2.162.1 item_1

The item_1 specifies the Plate_relationship is to relate a Plate with something else. See 4.3.113 for the application assertion.

4.2.163 Plate_renewal_definition

A Plate_renewal_definition is a type of Definition (see 4.2.56) that specifies the rule and the renewal thicknesses of a Plate. Rule values are values that are required by organizations such as classification societies for approval. These properties are specified during the newbuilding of the ship and are used during its operation, for example during classification society surveys.

The data associated with a Plate_renewal_definition are the following:

— defined_for;

— renewal_thickness;

— rule_thickness.

4.2.163.1 defined_for

The defined_for specifies the Plate objects that this Definition is valid for. There may be more than one defined_for for a Plate_renewal_definition. See 4.3.114for the application assertion.

4.2.163.2 renewal_thickness

The renewal_thickness specifies the minimum thickness of a Plate, below which it must be replaced or repaired.

4.2.163.3 rule_thickness

The rule_thickness specifies the minimum allowable thickness of a Plate during the life-time of the Plate. Rule_thickness is specified in the classification rules and regulations.

4.2.164 Plate_strake

A Plate_strake is a type of Structural_system (see 4.2.239) which forms a portion of a deck, bulkhead, or hull. A Plate_strake shall only consist of Plate (see 4.2.155) objects.

The data associated with a Plate_strake are the following:

— the_class.

4.2.164.1 the_class

The the_class specifies the class of a Plate_strake.

The value of the_class is one of the following:

— plate_strake_class.

NOTE See 4.2.164.1.1 for the definition of each allowable value for the_class.

4.2.164.1.1 plate_strake_class

plate_strake_class: special block that consists of only Plate objects; has a tightness function.

4.2.165 Plate_strake_functional_definition

A Plate_strake_functional_definition is a type of Structural_system_functional_definition (see 4.2.242) that defines the function of a Plate_strake (see 4.2.164).

The data associated with a Plate_strake_functional_definition are the following:

— defined_for.

4.2.165.1 defined_for

The defined_for specifies the Plate_strake objects that are defined by the Plate_strake_functional_-definition. There may be more than one defined_for for a Plate_strake_functional_definition. See 4.3.115 for the application assertion.

4.2.166 Position_feature

A Position_feature is a type of Structural_feature (see 4.2.226). Position_feature objects are used during the ship structure design process. They are intended to provide auxiliary geometry that is used to position structural elements with respect to each other or to define the geometry of not yet existing structural elements during the preliminary design phase. Each Position_feature is a Seam (see 4.2.198).

4.2.167 Position_feature_design_definition

A Position_feature_design_definition is a type of Feature_design_definition (see 4.2.86) that describes the definition of a Position_feature from the view point of design. Each Position_feature_-design_definition is a Seam_design_definition (see 4.2.200).

The data associated with a Position_feature_design_definition are the following:

— defined_for;

— representations.

4.2.167.1 defined_for

The defined_for specifies the Position_feature objects that are defined by the Position_feature_-design_definition. There may be more than one defined_for for a Position_feature_design_-definition. See 4.3.116 for the application assertion.

4.2.167.2 representations

The representations specifies the Position_feature_design_definition is to be represented by shape representations objects. The representations need not be specified for a particular Position_feature_-design_definition. There may be more than one representations for a Position_feature_design_-definition. See 4.3.117, 4.3.118, 4.3.119, 4.3.120 for the application assertion.

4.2.168 Position_feature_relationship

A Position_feature_relationship is a type of Structural_feature_relationship (see 4.2.227) that defines the association of a Position_feature (see 4.2.166) with another Item (see 4.2.115). Each Position_-feature_relationship is a Seam_curve_relationship (see 4.2.168).

The data associated with a Position_feature_relationship are the following:

— item_1.

4.2.168.1 item_1

The item_1 specifies the relating item of the relationship is a Position_feature. See 4.3.121 for the application assertion.

4.2.169 Precision

Precision is the geometric precision of the CAD system from which the product data originated. The data associated with a Precision are the following:

— minimum_point_spacing.

4.2.169.1 minimum_point_spacing

The minimum_point_spacing specifies the minimum distance between two points that are considered to be coincident in the originating CAD system.

4.2.170 Principal_characteristics

A Principal_characteristics is a type of General_characteristics_definition (see 4.2.97) that specify the main shape parameters of the hull moulded form. Principal_characteristics also includes data that is required in subsequent iterations of the hull development process when one is considering hydrostatics.

The data associated with a Principal_characteristics are the following:

— block_coefficient;

— design_deadweight;

— design_draught;

— length_between_perpendiculars;

— max_draught_at_ap;

— max_draught_at_fp;

— min_draught_at_ap;

— min_draught_at_fp;

— moulded_breadth;

— moulded_depth.

4.2.170.1 block_coefficient

The block_coefficient specifies the ratio of the moulded displacement volume to the volume of a block that has its length equal to the length_between_perpendiculars, its breadth equal to the maximum immersed moulded_breadth and its depth equal to the design_draught. The block_-coefficient should be defined only for mono hull ships. The block_coefficient need not be specified for a particular Principal_characteristics.

4.2.170.2 design_deadweight

The design_deadweight specifies the weight of the ship representing the weight of cargo, bunker fuel, water, passengers, crew and consumables that a ship can carry when loaded to the summer load line.

4.2.170.3 design_draught

The design_draught specifies the draught to which the ship has been designed to operate.

4.2.170.4 length_between_perpendiculars

The length_between_perpendiculars specifies the length measured from the after perpendicular to the forward perpendicular of the ship.

4.2.170.5 max_draught_at_ap

The max_draught_at_ap specifies the maximum possible draught at the after perpendicular during the operation of the ship. The max_draught_at_ap is used for hull cross section approval for ice class notation.

4.2.170.6 max_draught_at_fp

The max_draught_at_fp specifies the maximum possible draught at the forward perpendicular during the operation of the ship. The max_draught_at_fp is used for hull cross section approval for ice class notation.

4.2.170.7 min_draught_at_ap

The min_draught_at_AP specifies the minimum possible draught at the after perpendicular during the operation of the ship. The min_draught_at_AP is used for hull cross section approval for ice class notation.

4.2.170.8 min_draught_at_fp

The min_draught_at_fp specifies the minimum possible draught at the forward perpendicular during the operation of the ship. The min_draught_at_fp is used for hull cross section approval for ice class notation.

4.2.170.9 moulded_breadth

The moulded_breadth specifies the maximum breath of the ship amidships and at the design_draught.

4.2.170.10 moulded_depth

The moulded_depth specifies the vertical distance above the baseline to the uppermost deck where the deck joins the side of the ship measured amidships.

4.2.171 Profile

A Profile is a type of Structural_part (see 4.2.229) that is made from profile shaped stock material.

4.2.172 Profile_approval

A Profile_approval is a type of Structural_class_approval (see 4.2.219) that confirms the approval of a Profile_design_definition (see 4.2.176).

The data associated with a Profile_approval are the following:

— subject.

4.2.172.1 subject

The subject specifies the approved Profile_design_definition. See 4.3.122 for the application assertion.

4.2.173 Profile_boundary_relationship

A Profile_boundary_relationship is a type of Profile_relationship (see 4.2.180) that defines a boundary for the trace line of a Profile by referring to a structural item. Structural item can be one of the following:

— Profile - the boundary for the trace line is the intersection point with the trace line of the referred Profile;

— Panel_system - the boundary for the trace line is specified by the intersection with a boundary of the Panel_system the Profile belongs to or the intersection with the moulded surface, implicit or explicit given;

— Plate - if the Plate is a bracket and belongs to the same Panel_system as the Profile does the boundary for the trace line is specified by the intersection with the moulded surface, implicit or explicit given;

— Seam - the boundary for the trace line is the intersection point;

— Interior_cutout - if the Interior_cutout belongs to the same Panel_system as the Profile does, the boundary of the trace line is the intersection point with the contour.

The data associated with a Profile_boundary_relationship are the following:

— boundary_index.

— displacement.

4.2.173.1 boundary_index

The boundary_index specifies the index of the referred boundary in the boundary list of the Panel_-system_design_definition associated with item_2, in case item_2 is a Panel_system. The boundary_-index need not to be specified for a particular Profile_boundary_relationship.

NOTE If item_2 is a Panel_system and boundary_index is not given intersection with the moulded surface of the Panel_system is taken into account.

4.2.173.2 displacement

The displacement specifies an offset of the resulting intersection point along the trace line. A positive value means a change of length in the positive direction of the trace line and vice versa. The displacement need not be specified for a particular Profile_boundary_relationship.

NOTE Displacements for end-points or start-points of profiles may also be modelled using Corner_cutout objects.

4.2.174 Profile_cross_section

A Profile_cross_section is a generic indicator of the standard or organization-unique two-dimensional shape of an extruded manufactured or purchased structural part used to stiffen a plate or to provide other structural support. A Profile_cross_section may be an Explicit_profile_cross_section (see 4.2.81), a Library_profile_cross_section (see 4.2.122) or a Parametric_profile_cross_section (see 4.2.151).

EXAMPLE ISO 657, ISO 1035, and ISO 9034 specify dimensions of some standard profile cross sections available from major steel manufacturers.

The data associated with a Profile_cross_section are the following:

— designation;

— section_properties.

4.2.174.1 designation

The designation specifies the part or stock designation name.

4.2.174.2 section_properties

The section_properties specify derived analytical properties of the geometric shape of the profile cross section that determine its load-bearing capacity. The section_properties need not be specified for a particular Profile_cross_section. See 4.3.123 for the application assertion.

4.2.175 Profile_curve_trace_line

A Profile_curve_trace_line specifies the geometry of the trace line for a Profile (see 4.2.171). If displacement is specified, then the trace line geometry is represented parametrically by a curve_replica where the parent_curve is the specified curve and the transformation is the translation defined by the displacement vector. If displacement is not specified, the trace line geometry is defined by the curve.

The data associated with a Profile_curve_trace_line are the following:

— curve;

— displacement.

4.2.175.1 curve

The curve specifies a curve.

4.2.175.2 displacement

The displacement specifies a vector. The displacement need not be specified for a particular Profile_curve_trace_line.

4.2.176 Profile_design_definition

A Profile_design_definition is a type of Structural_part_design_definition (see 4.2.231) that describes the Profile (see 4.2.171) from the detailed design point of view.

The design of the profile is defined parametrically by sweeping the specified Profile_cross_section along the trace line using the orientation defined by the cross_section_placement. The representation of the resulting shape depends on the geometry of the trace line and the presence or absence of twist. The actual orientation of the Profile_cross_section when swept along the trace line is defined relative

to the surface normal if the trace_curve is specified as a curve_on_surface; otherwise, it is defined relative to a projection of the Y-axis of the cross_section_placement. Twist_location objects may also be specified to change the orientation of the Profile_cross_section along the trace line.

If the trace line geometry is defined as a curve_on_surface and the cross_section_placement is oriented with the Y-axis in the direction of the surface normal, i.e., with no declination, then the resulting shape without twist is represented as a surface_curve_swept_surface, where the directrix is the trace line and the swept_curve is the Profile_cross_section. If the trace line is specified other than by a curve_on_surface, the resulting shape without twist is represented by a fixed_reference_swept_surface, where the directrix is the trace line, the swept_curve is the Profile_cross_section, and the reference direction is the Y-axis of the cross_section_placement.

NOTE 1 Figure 33 illustrates a profile without twist, generated by a rectangular Profile_cross_section oriented perpendicular to a singly curved plate. In this case, the resulting profile shape is the same, regardless of the type of trace line geometry used. If the trace line is given as a curve_on_surface, the profile is a surface_curve_swept_surface in which the Y-axis is always normal to the surface. If the trace line is otherwise specified, the profile shape is represented by a fixed_reference_swept_surface. In this case, the Y-axis is given by the projection of reference direction on Z=0 plane. Since the Y-axis of the cross_section_placement was normal to the underlying surface, the local Y-axis is normal to the surface along the trace line.

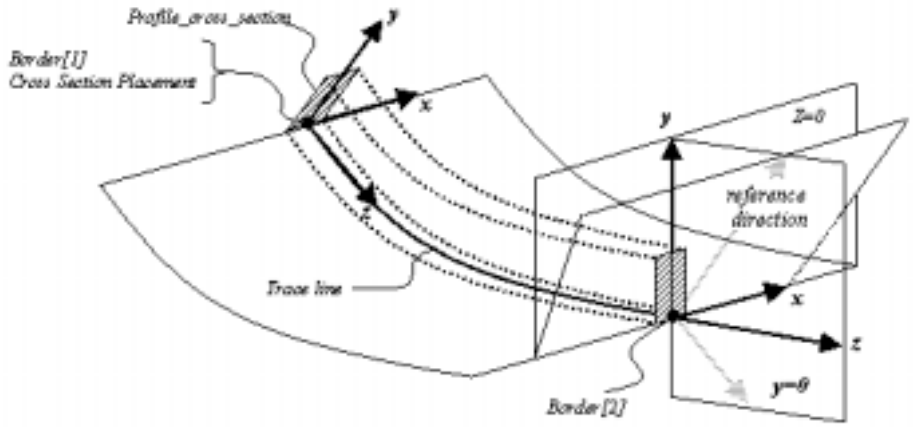


Figure 33 — Profile design of a simple profile

NOTE 2 Figure 34 illustrates a profile without twist, generated by a rectangular Profile_cross_section oriented perpendicular to a doubly curved surface where the trace line is given as a curve_on_surface. The resulting profile shape is a surface_curve_swept_surface as shown. Although the profile shape is defined without twist, physically the orientation of the web varies along the profile, and manufacturing such a stiffener would require physically twisting the web. In contrast, a profile without twist defined by a trace line other than a curve_on_surface results in a profile shape with a fixed web orientation.

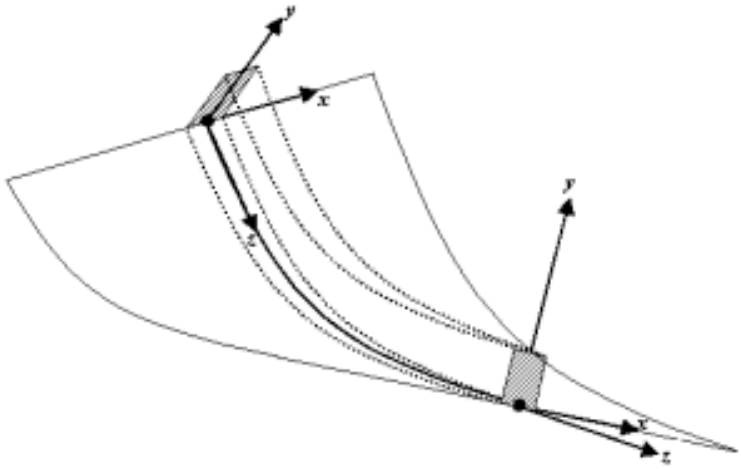


Figure 34 — Profile design using curve_on_surface geometry for trace line

If the trace line is defined as a curve_on_surface and the cross_section_placement is oriented with the Y-axis not in the direction of the surface normal, i.e., with declination, then the resulting shape without twist is represented as a surface_curve_swept_surface, where the directrix is the trace line and the swept_curve is the Profile_cross_section rotated by the declination angle.

NOTE 3 Figure 35 illustrates a profile with declination, but without twist, generated by a rectangular Profile_cross_section. In this case, the resulting profile shape is the same, regardless of the type of trace line geometry used. If the trace line is given as a curve_on_surface, the profile is a surface_curve_swept_surface in which the Profile_cross_section is always at the declination angle to the normal. If the trace line is otherwise specified, the profile shape is represented by a fixed_reference_swept_surface. In this case, the Y-axis is given by the projection of reference direction on Z=0 plane. Since the Y-axis of the cross_section_placement was at the declination angle to the normal of the underlying surface, the local Y-axis is also at the declination angle along the trace line.

Twist is used to specify profile shapes in which the direction of web with respect to either the surface normal or reference direction varies along the trace line. The twist, specified by Twist_location objects, is assumed to continuously change along the trace line between the Twist_location objects. The twist angle at a point on the trace line is the angle between the local Y-axis and the projection of the twist onto the local Z=0 plane. At each point along the trace line, the Profile_cross_section is additionally rotated by the twist angle.

If the trace line is defined as a curve_on_surface and the cross_section_placement is oriented with the Y-axis not in the direction of the surface normal, i.e., with declination, then the resulting shape with twist is represented as a surface_curve_swept_surface, where the directrix is the trace line and the swept_curve is the Profile_cross_section rotated by the declination angle plus the twist angle.

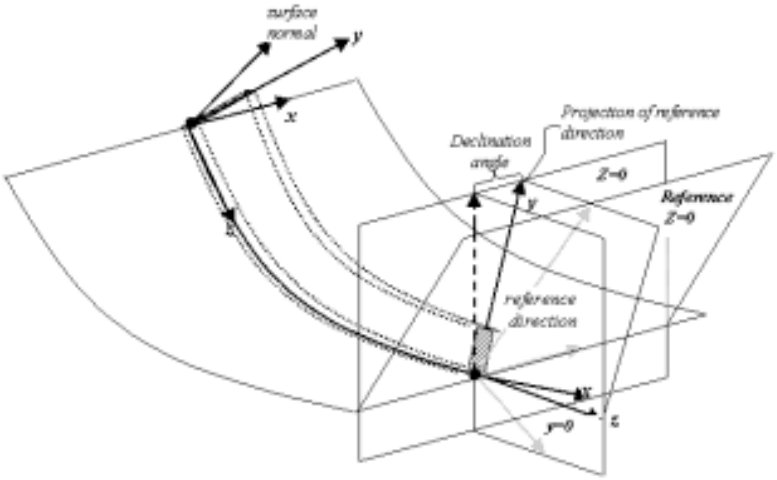


Figure 35 — Profile design with declination angle

NOTE 4 Figure 36 illustrates a profile with twist generated by a rectangular Profile_cross_section on a doubly curved surface where the trace line is not given as a curve_on_surface. Although the local Y-axis at border [2] is not normal to the surface, the Twist_location specifies the same direction as the surface normal, ensuring that the profile shape follows the contour of the underlying surface.

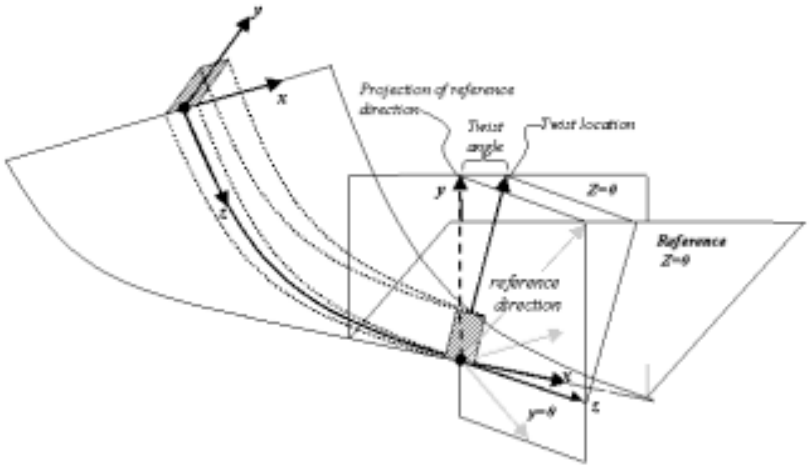


Figure 36 — Profile design using twist

The data associated with a Profile_design_definition are the following:

— border;

— cross_section;

— cross_section_placement;

— defined_for;

— mirrored;

— trace_line;

— twist.

4.2.176.1 border

The border specifies the boundary of the trace_line as a list of two geometric objects, Profile_boundary_relationship objects, or a combination of the two. If the trace_line curve is bounded or closed, the boundaries are implicitly defined by the end points of the curve.

The boundary can be one of: the following:

— an intersection point between the trace line and given line;

— a parametric point on the trace line;

— a cartesian point;

— a relationship to a Profile_boundary_relationship object.

The border need not to be specified for a particular Profile_design_definition. See 4.3.126 for the application assertion.

4.2.176.2 cross_section

The cross_section specifies the Profile_cross_section of the profile. See 4.3.127 for the application assertion.

4.2.176.3 cross_section_placement

The cross_section_placement specifies the orientation of the Profile_cross_section at border[1]. The Z-axis is tangent to the trace line at this point. The X-axis is the local u-axis of the cross-section, and the Y-axis is the local v-axis of the cross-section, tangent to the web of the profile. The local u-v plane of the cross-section is everywhere normal to the extrusion curve.

NOTE If the trace line is defined as a curve_on_surface, the angle between the surface normal and the Z-axis at border[1] is called the declination angle.

4.2.176.4 defined_for

The defined_for specifies a Pofile_design_definition is only to be valid for Profile objects. There may be more than one defined_for for a Pofile_design_definition. See 4.3.125 for the application assertion.

4.2.176.5 mirrored

The mirrored describes if the cross section is mirrored with respect to it's definition.

4.2.176.6 trace_line

The trace_line specifies the trace line of the Profile. If not specified, the local X-axis is implicitly assumed to be trace line. The trace line can be one of the following :

— a curve parallel to a given curve with a given displacement;

— a curve parallel to a boundary of a Panel_system with a given displacement;

— a line parallel to the tangent on a boundary of the same Panel_system in the intersection point with a given line;

— a curve parallel to a hole contour of the same panel with a given displacement.

The trace_line need not to be specified for a particular Profile_design_definition. See 4.3.124, 4.3.128, 4.3.129 for the application assertion.

NOTE The trace line is assumed to be tangent continuous.

4.2.176.7 twist

The twist specifies a set of Twist_location objects that are the angle of the local v-axis of the cross-section at locations along the extrusion trace for profiles that twist along their length. The location and orientation of twist is specified at explicit points along the length. The twist need not be specified for a particular Profile_design_definition. There may be more than one twist for a Profile_-design_definition. See 4.3.130 for the application assertion.

NOTE The change in orientation between Twist_location objects is assumed to be smooth, not knuckled.. The software system exporting the profile is responsible for generating a sufficient number of points to ensure that the twist is well defined.

4.2.177 Profile_functional_definition

A Profile_functional_definition is a type of Structural_part_functional_definition (see 4.2.232) that defines the functional role of a Profile (see 4.2.171); the role may be a pre-defined one or may be user-defined.

The data associated with a Profile_functional_definition are the following:

— defined_for;

— the_function.

4.2.177.1 defined_for

The defined_for specifies the redefinition of the defined_for attribute. A Profile_functional_-definition is only valid for one or multiple Profile objects. There may be more than one defined_for for a Profile_functional_definition. See 4.3.131 for the application assertion.

4.2.177.2 the_function

The the_function specifies the purpose for the Profile_functional_definition.

The value of the the_function is one of the following:

— stiffener;

— longitudinal_stiffener;

— transversal_stiffener;

— carling;

— beam;

— deck_beam;

— pillar;

— hold_pillar;

— flange;

— web;

— frame;

— user_defined.

NOTE See 4.2.177.2.1- 4.2.177.2.12 for the definition of each allowable value for the_function.

4.2.177.2.1 stiffener

a structural member that adds localized strength to a plate area.

NOTE A stiffener may be a tee beam, an angle beam, or a flat bar, any of which is welded to a plate.

4.2.177.2.2 longitudinal_stiffener

a structural member that stiffens the plating fore-and-aft in the ship.

4.2.177.2.3 transversal_stiffener

a structural member that stiffens the plating from port-to-starboard in the ship.

4.2.177.2.4 carling

a short beam forming a portion of the framing about deck opening. Also called header when it supports the ends of interrupted deck beams.

4.2.177.2.5 beam

a structural member that is designed to support or carry a specified load. It is an athwartship or longitudinal member of the ship frame supporting the deck.

EXAMPLE A beam may be a I- beam, a wide flange beam, or a girder of welded steel plate.

4.2.177.2.6 deck_beam

a structural member that is designed to support or carry a specified load on the deck and attached under the deck plating.

EXAMPLE A deck_beam may be a I- beam, a wide flange beam, or a girder of welded steel plate.

4.2.177.2.7 pillar

a vertical structural support column.

4.2.177.2.8 hold_pillar

a major structural support member in or under a large compartment area for cargo.

4.2.177.2.9 flange

the turned edge of a plate or girder, or in a rolled section, which acts to resist bending.

EXAMPLE The flange in a Wide Flange or a channel rolled section.

4.2.177.2.10 web

the portion of a beam that is located between two flanges.

EXAMPLE The web of an I-beam or that part of a web-frame between the shell plating and the internal flange.

4.2.177.2.11 frame

a transverse rib that is a part of the skeleton of the ship.

NOTE Frames act as stiffeners, holding the outside plating in shape and maintaining the transverse form of the ship.

4.2.177.2.12 user_defined

the Profile_functional_definition is defined within the user_def_function attribute, by the user.

NOTE The user_defined option may only be used when the Structural_part_functional_definition cannot be defined by the other function codes.

4.2.178 Profile_manufacturing_definition

A Profile_manufacturing_definition is a type of Structural_part_manufacturing_definition (see 4.2.235) that is a production lifecycle stage view of a profile part that represents characteristics and attributes directly related to the manufacturing of the part.

NOTE An earlier detailed design part stage view may or may not exist for the part. At this stage, structural part features are identified that affect aspects of the part that have to do with processes that will be used to cut and form the part. Some examples of the kinds of features identified during this stage are edge grinds, layout marks, and part stock. Information relating the structural part to the raw material stock it will be manufactured from is also added during this stage. A 2D flat pattern representation may be defined for the part, enabling it to be nested and cut from profile stock.

The data associated with a Profile_manufacturing_definition are the following:

— defined_for;

— inverse_bend_trace;

— outer_flange_contour;

— outer_web_contour.

4.2.178.1 defined_for

The defined_for specifies the redeclaration of the defined_for attribute to be allowed to point to Profile only. There may be more than one defined_for for a Profile_manufacturing_definition. See 4.3.132 for the application assertion.

4.2.178.2 inverse_bend_trace

The inverse_bend_trace specifies the curve in the representation of a curved profile that indicatesthe amount that the profile is to be shaped during manufacturing. As the profile part is bent into its correct final shape, the inverse bend trace will become a straight line. A inverse_bend_trace may have no value contained in it or it may have one or more values, without any duplications.

4.2.178.3 outer_flange_contour

The outer_flange_contour specifies the curve in the representation that specifies the outer boundary of the part's flange geometry that will be burned during the manufacturing process. This is the resulting curve after the resolution of manufacturing features such as added material feature and

weld_shrinkage allowance, and edge and corner features. An outer_flange_contour may have no value contained in it or it may have one or more values, without any duplications.

4.2.178.4 outer_web_contour

The outer_web_contour specifies the curve in the representation that specifies the outer boundary of the part's web geometry that will be burned during the manufacturing process. This is the resulting curve after the resolution of manufacturing features such as added material feature and weld_-shrinkage allowance, and edge and corner features. An outer_web_contour may have no value contained in it or it may have one or more values, without any duplications.

4.2.179 Profile_rejection

The Profile_rejection is a type of Structural_class_rejection (see 4.2.222) that confirms that a Profile_design_definition (see 4.2.176) is not approved. The reasons for rejection and existing requirements for the profile are specified.

NOTE Profile_rejection specifies additional information clarifying the rule requirements for the Profile. A yard has several options for action; increase the profile dimensions, improve the material quality, apply another profile etc. Although it is not mandatory, a class society may suggest improvements in these areas. An improvement might be applying additional stiffening. This may then be done using Structural_class_-rejection.proposed_alternative.

The data associated with a Profile_rejection are the following:

— subject.

4.2.179.1 subject

The subject specifies the rejected Profile_design_definition. See 4.3.133 for the application assertion.

4.2.180 Profile_relationship

A Profile_relationship is a type of Structural_part_relationship (see 4.2.237) that describes the relationship between a Profile (see 4.2.171) and another structural item. Structural item can be one of the following:

— Structural_system;

— Structural_part;

— Structural_feature.

Each Profile_relationship is a Profile_boundary_relationship (see 4.2.173).

The data associated with a Profile_relationship are the following:

— item_1.

4.2.180.1 item_1

The item_1 specifies the Profile that will be related to something else. See 4.3.134 for the application assertion.

4.2.181 Profile_renewal_definition

A Profile_renewal_definition is a type of Definition (see 4.2.56) that specifies the rule and the renewal section modulus of a Profile and its rule inertia.

NOTE Rule values are values that are required by organizations for approval, such as classification societies. These properties are specified during the new building of the ship and are used during its operation, for example during classification society surveys.

The data associated with a Profile_renewal_definition are the following:

— defined_for;

— renewal_modulus;

— rule_inertia;

— rule_section_modulus.

4.2.181.1 defined_for

The defined_for specifies the Profiles that this Definition is valid for. There may be more than one defined_for for a Profile_renewal_definition. See 4.3.135 for the application assertion.

4.2.181.2 renewal_modulus

The renewal_modulus specifies the minimum section modulus of a Profile, below which it must be replaced or repaired.

4.2.181.3 rule_inertia

The rule_inertia specifies the minimum allowable moment of inertia of a Profile during the life-time. Rule_inertia is specified in the classification rules and regulations.

4.2.181.4 rule_section_modulus

The rule_section_modulus specifies the minimum allowable section modulus of a Profile during its life-time. Rule_section_modulus is specified in the classification rules and regulations.

4.2.182 Profile_trace_line_relationship

A Profile_trace_line_relationship is a type of Profile_relationship (see 4.2.180) that defines the trace line of a Profile by referring to a structural item. Structural item can be one of the following:

— Panel_system - the trace line is specified by a boundary or the tangent on a boundary at the intersection point with an auxiliary line and an orthogonal displacement off that boundary or tangent;

— Interior_cutout - the trace line is specified by the contour and an orthogonal displacement off that contour;

— Seam - the trace line is specified by the contour and an orthogonal displacement off that contour.

The data associated with a Profile_trace_line_relationship are the following:

— auxiliary_line;

— boundary_index;

— displacement.

4.2.182.1 auxiliary_line

The auxiliary_line specifies the point as the intersection point of this line with the referenced boundary contour at which the tangent on the boundary referenced by boundary_index is to be taken as traceline. The auxiliary_line need not to be specified for a particular Profile_trace_line_-relationship.

4.2.182.2 boundary_index

The boundary_index specifies the index of the referred boundary in the boundary list of the Panel_-system_design_definition associated with item_2, in case item_2 is a Panel_system. The boundary_-index need not to be specified for a particular Profile_trace_line_relationship.

4.2.182.3 displacement

The displacement specifies the offset orthogonal off the boundary refered to by boundary_index.

4.2.183 Property_bsu

A Property_bsu is a type of Bsu (see 4.2.18) that identifies a property basic semantical unit of a class in a parts library.

The data associated with a Property_bsu are the following:

— dic_identifier;

— name_scope;

— version.

4.2.183.1 dic_identifier

The dic_identifier specifies the identifier of the information piece constructed from the code and the version data.

4.2.183.2 name_scope

The name_scope specifies the class this property belongs to. See 4.3.140 for the application assertion.

4.2.183.3 version

The version specifies the designation of the version of the information piece.

4.2.184 Property_value

A Property_value provides a value for a property as specified in the property basic semantical unit. The value type is specified in subtypes. A Property_value may be a Real_property_value (see 4.2.186), a Integer_property_value (see 4.2.109), a Number_property_value (see 4.2.141), a String_property_value (see 4.2.215), a Boolean_property_value (see 4.2.17),or a Logical_property_value (see 4.2.128).

The data associated with a Property_value are the following:

— property_identifier.

4.2.184.1 property_identifier

The property_identifier specifies the property this value is for. See 4.3.141 for the application assertion.

4.2.185 Proprietary_profile_cross_section

A Proprietary_profile_cross_section is a type of Parametric_profile_cross_section (see 4.2.151) that is not specified by national or international standard. Each Proprietary_profile_cross_section is a Flanged_plate_cross_section (see 4.2.89).

4.2.186 Real_property_value

A Real_property_value is a type of Property_value (see 4.2.184) that provides a value for a property that is a numeric value.

The data associated with a Real_property_value are the following:

— val.

4.2.186.1 val

The val specifies a numeric value.

4.2.187 Reason_for_decision

A Reason_for_decision is the background for a decision. This background is to be given as a reference to a document and as an informal textual description.

The data associated with a Reason_for_decision are the following:

— annotations;

— rule_reference.

4.2.187.1 annotations

The annotations specifies the informative textual description of the reason for a decision.

4.2.187.2 rule_reference

The rule_reference specifies the reason for a decision in terms of a reference to a document. See 4.3.142 for the application assertion.

4.2.188 Rectangular_cutback_corner_design_definition

A Rectangular_cutback_corner_design_definition is a type of Corner_cutout_design_definition (see 4.2.44) that specifies the removal of material from a corner such that the resulting cutoff edges are parallel to the local coordinate system axis. The local origin is placed at the corner that is to be removed.

NOTE Figure 37 illustrates a Rectangular_cutback_corner_cutout.

The data associated with a Rectangular_cutback_corner_design_definition are the following:

— radius;

— x_depth;

— y_depth.

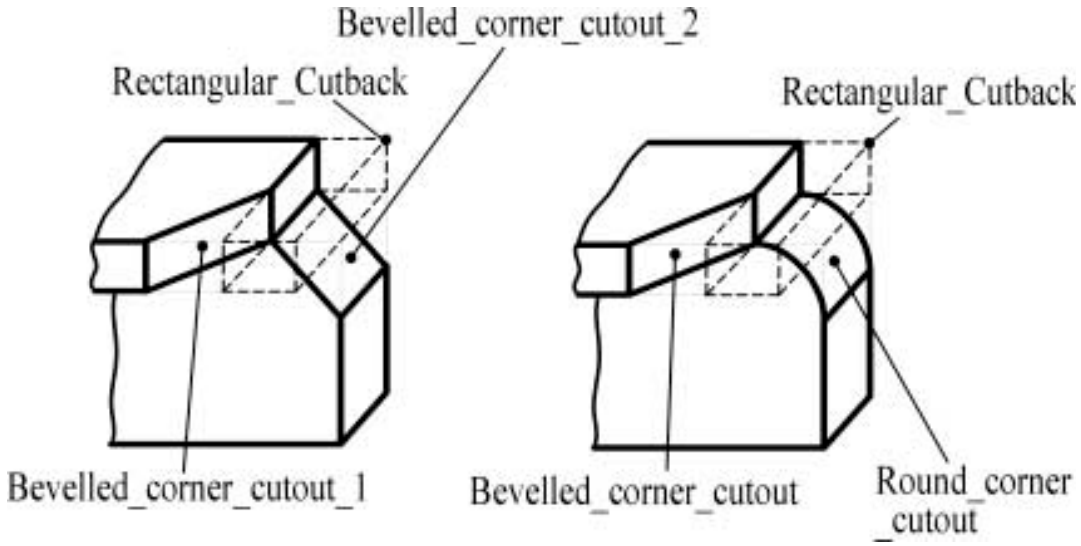


Figure 37 — Rectangular_cutback_corner_cutout

4.2.188.1 radius

The radius specifies the rounding of the new corner inside the cutback. A radius of 0.0 indicates a square corner. The circular arc defined by the radius is tangent to the edges of the cutout.

4.2.188.2 x_depth

The x_depth specifies the distance from local origin along X-axis.

4.2.188.3 y_depth

The y_depth specifies the distance from local origin along Y-axis.

4.2.189 Rectangular_cutout_design_definition

A Rectangular_cutout_design_definition is a type of Interior_cutout_design_definition (see 4.2.111) that is an opening where the opposing sides are parallel and adjacent sides are perpendicular. Each Rectangular_cutout_design_definition may be one of the following: a Round_corner_rectangular_-cutout_design_definition (see 4.2.196), or a Round_edge_rectangular_cutout_design_definition (see 4.2.197).

NOTE Combination of Round_corner_rectangular_cutout_design_definition with Round_edge_-rectangular_cutout_design_definition is allowed.

The data associated with a Rectangular_cutout_design_definition are the following:

— length_of;

— width.

4.2.189.1 length_of

The length_of specifies the opening along X-axis.

4.2.189.2 width

The width specifies the opening along Y-axis.

4.2.190 Regulation

A Regulation specifies the set of all international and national regulations as well as other standards that apply to the Ship.

The data associated with a Regulation are the following:

— international_regulations;

— national_regulations;

— standards.

4.2.190.1 international_regulations

The international_regulations specifies all relevant international regulations which apply to the Ship. The international_regulations need not be specified for a particular Regulation. There may be more than one international_regulations for a Regulation. See 4.3.143 for the application assertion.

4.2.190.2 national_regulations

The national_regulations specifies all relevant national regulations which apply to the Ship. The national_regulations need not be specified for a particular Regulation. There may be more than one national_regulations for a Regulation. See 4.3.143 for the application assertion.

4.2.190.3 standards

The standards specifies all relevant standards which apply to the Ship. The standards need not be specified for a particular Regulation. There may be more than one standards for a Regulation. See 4.3.143 for the application assertion.

4.2.191 Research_ship

A Research_ship is a type of Shiptype (see 4.2.207) that is a ship which is specially designed for research.

The data associated with a Research_ship are the following:

— has_type.

4.2.191.1 has_type

The has_type specifies the type of research ships.

The value of the has_type is one of the following:

— user defined

NOTE See 4.2.191.1.1 for the definition of each allowable value for has_type.

4.2.191.1.1 user_defined

user_defined: can be any shiptype, details should be found in the ship_type_description attribute for Ship_designation.

4.2.192 Revision

A Revision is a type of Versionable_object (see 4.2.263) that serves the link between the object of interest and the Definitions of its constituents and associated members. A Revision is not created automatically, but has to be created explicitly each time it is needed.

EXAMPLE The object of interest can be a hull cross section whose members are plate definitions, but only those plate definitions that belong to the same version.

The data associated with a Revision are the following:

— description;

— members;

— name.

4.2.192.1 description

The description specifies the description of what caused the creation of a new Revision.

4.2.192.2 members

The members specifies the Versionable_object objects of the Revision. There may be more than one members for a Revision. See 4.3.144 for the application assertion.

4.2.192.3 name

The name specifies the label which identifies a particular Revision.

4.2.193 Revision_with_context

A Revision_with_context is a type of Revision (see 4.2.192) that serves as a link between the object of interest, the context, and the definitions of its constituents and associated members.

The data associated with a Revision_with_context are the following:

— context_of_revision.

4.2.193.1 context_of_revision

The context_of_revision specifies the link to a higher level object of interest. This may be not be known explicitly. See 4.3.145 for the application assertion.

4.2.194 Round_bar_cross_section

A Round_bar_cross_section is a type of a Bar_profile_cross_section (see 4.2.13) identified by a solid circular shape.

NOTE ISO 1035-1 specifies dimensions for some round bars.

The data associated with a Round_bar_cross_section are the following:

— diameter.

4.2.194.1 diameter

The diameter specifies the the outside diameter of the bar.

4.2.195 Round_corner_design_definition

A Round_corner_design_definition is a type of Corner_cutout_design_definition (see 4.2.44) that specifies the circular arc of the cylindrical surface between two adjacent edges on surfaces. Each Round_corner_design_definition may be one of the following: an Inward_round_corner_design_-definition (see 4.2.114), or an Outward_round_corner_design_definition (see 4.2.142).

The data associated with a Round_corner_design_definition are the following:

— radius.

4.2.195.1 radius

The radius specifies the radius of the circular arc, cylindrical surface, between two adjacent edges, surfaces, of the corner. A radius of 0.0 indicates a square corner. The circular arc defined by the radius is tangent to the edges of the cutout.

4.2.196 Round_corner_rectangular_cutout_design_definition

A Round_corner_rectangular_cutout is a type of Rectangular_cutout_design_definition (see 4.2.189) that is an opening where the opposing sides are parallel and adjacent sides are perpendicular. The corners are rounded off with the same radius.

NOTE Figure 38 illustrates parameters of a Round_corner_rectangular_cutout.

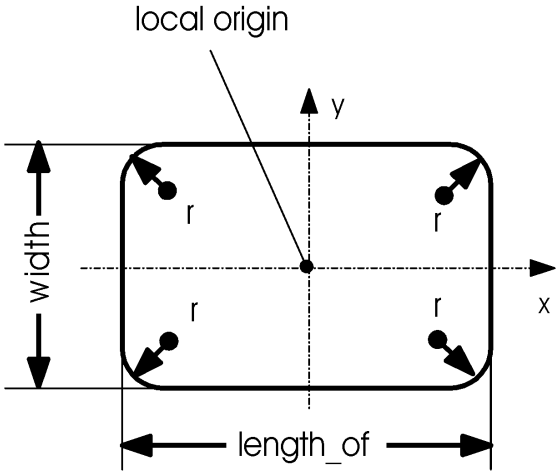


Figure 38 — Round_corner_rectangular_cutout

The data associated with a Round_corner_rectangular_cutout_design_definition are the following:

— corner_radius.

4.2.196.1 corner_radius

The corner_radius specifies the radius of the rounded corners; the same radius applies to all corners. A corner_radius of 0.0 indicates a square corner. The circular arc defined by the corner_radius is tangent to the edges of the cutout.

4.2.197 Round_edge_rectangular_cutout_design_definition

A Round_edge_rectangular_cutout_design_definition is a type of Rectangular_cutout_design_-definition (see 4.2.189) that has the same basic shape as a Rectangular_cutout but in addition two of the opposing sides are substituted by outward bended circle arcs.

The data associated with a Round_edge_rectangular_cutout_design_definition are the following:

— distance;

— edge_radius.

4.2.197.1 distance

The distance specifies the distance along X-axis of the circle center from the origin.

4.2.197.2 edge_radius

The edge_radius specifies the radius of the circular arc that describes the outward bended sides of the bended sides of the opening. An edge_radius of 0.0 indicates a square corner. The circular arc defined by the edge_radius is tangent to the edges of the cutout.

4.2.198 Seam

A Seam is a type of Position_feature (see 4.2.166) that provides a reference for an inner Plate (see 4.2.155) boundary on a Panel_system (see 4.2.144). Seam objects are usually created in the preliminary design stage in order to define where the Panel_system will be split into Plate objects in a later design stage.

The data associated with a Seam are the following:

— parent.

4.2.198.1 parent

The parent specifies the Seam is to be related to a Panel_system only. See 4.3.146 for the application assertion.

4.2.199 Seam_curve_relationship

A Seam_curve_relationship is a type of Position_feature_relationship (see 4.2.168) that defines the curve of a Seam by referring to a structural item. Structural item can be one of the following:

— Panel_system (see 4.2.144) - the trace line is specified by a boundary of the Panel_system and an orthogonal displacement off that boundary;

— Seam (see 4.2.198) - the trace line is specified by the curve of another Seam and an orthogonal displacement off that curve.

The data associated with a Seam_curve_relationship are the following:

— boundary_index;

— displacement.

4.2.199.1 boundary_index

The boundary_index specifies the index of the referred boundary in the boundary list of the Panel_-system_design_definition associated with item_2, in case item_2 is a Panel_system. The boundary_-index need not to be specified for a particular Seam_curve_relationship.

4.2.199.2 displacement

The displacement specifies the offset off the referred geometry. The displacement need not be specified for a particular Seam_curve_relationship.

4.2.200 Seam_design_definition

A Seam_design_definition is a type of Position_feature_design_definition (see 4.2.86) that describes the definition of a Seam (see 4.2.198) from the view point of design.

The data associated with a Seam_design_definition are the following:

— border;

— defined_for;

— seam_curve.

4.2.200.1 border

The border specifies a Seam may optionally be bounded on one or both ends by another Seam. There may be more than one border for a Seam_design_definition. See 4.3.147 for the application assertion.

4.2.200.2 defined_for

The defined_for specifies the Seam objects that are defined by the Seam_design_definition. There may be more than one defined_for for a Seam_design_definition. See 4.3.147 for the application assertion.

4.2.200.3 seam_curve

The seam_curve specifies the curve defining the Seam, if not specified the local X-axis is implicitly assumed as the seam defining curve. A seam curve can be one of the following:

— a given curve;

— a curve parallel to a boundary of a Panel_system with a given displacement;

— a curve parallel to a given Seam with a given displacement.

The seam_curve is on both ends required to intersect with the parent Panel_system's boundary, as provided by the related Panel_system_design_definition, or a border must explicitly be given for the Seam on that curve end. The seam_curve need not be specified for a particular Seam_design_-definition. See 4.3.148 for the application assertion.

4.2.201 Section_properties

The Section_properties hold some basic quantities characterizing the cross section of a profile which are often needed for the dimensioning and engineering calculations.

NOTE The following quantities can be derived from the attributes of Section_properties using the given formulas:

— section modulus in u (moi_u / na_u);

— section modulus in v (moi_v / na_v);

— radius of gyration in u (SQRT(moi_u / area));

— radius of gyration in v (SQRT(moi_v / area)).

The data associated with the Section_properties are the following:

— area;

— moi_u;

— moi_uv;

— moi_v;

— na_u;

— na_v;

— nominal_mass_per_len;

— tr.

4.2.201.1 area

The area specifies the area of solid material in the cross-section.

4.2.201.2 moi_u

The moi_u specifies the moment of inertia in U.

4.2.201.3 moi_uv

The moi_uv specifies the cross moment of interia. This value is zero if symmetric in U or V.

4.2.201.4 moi_v

The moi_v specifies the moment of inertia in V.

4.2.201.5 na_u

The na_u specifies the neutral axis in U - distance along u to centroid from local coordinate system.

4.2.201.6 na_v

The na_v specifies the neutral axis in V - distance along v to centroid from local coordinate system.

4.2.201.7 nominal_mass_per_len

The nominal_mass_per_len specifies the mass per unit length.

4.2.201.8 tr

The tr specifies the torsional rigidity.

4.2.202 Shear_bevel_design_definition

A Shear_bevel_design_definition is a type of Bevel_design_definition (see 4.2.16) that specifies the 3-dimensional surface obtained when intersecting a corner with a plane not being parallel to any of the corner surfaces.

NOTE Figure 39 illustrates the Shear_bevel_design_definition parameters.

The data associated with a Shear_bevel_design_definition are the following:

— x_z_angle.

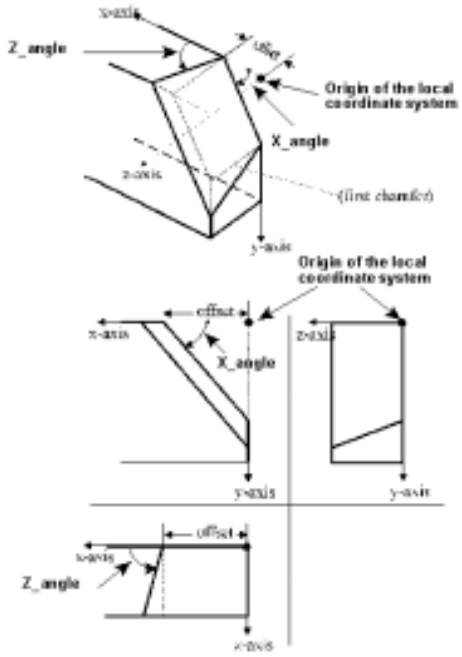


Figure 39 — Shear_bevel_design_definition

4.2.202.1 x_z_angle

The x_z_angle specifies the angle between the X-axis and the beveled side in the x-z plane.

4.2.203 Shear_force

A Shear_force is a type of Design_load (see 4.2.59) that is one way to represent Design_load objects, that is by the physical quantity force. Each Shear_force may be one of the following: a Design_still_water_shear_force (see 4.2.61), a Design_vertical_wave_shear_force (see 4.2.64), or a Maximum_permissible_still_water_shear_force (see 4.2.132).

4.2.204 Ship

A Ship is a type of Item (see 4.2.115) that is the primary product supported by the suite of ISO 10303 shipbuilding application protocols. All data defining the product is related to a Ship. Product model definition data related to the Ship object is supported for many stages of the life cycle of a ship,

including new project, early and detailed design, production engineering, manufacturing, operations, and scrapping.

The data associated with a Ship are the following:

— single_hull_or_class;

— ship_items;

— units.

4.2.204.1 single_hull_or_class

The single_hull_or_class specifies whether the exchange of data is applicable to a single hull or to multiple hulls in a class of ships.

The value of the single_hull_or_class is one of the following:

— design_for_single_hull;

— design_for_multiple_hulls.

NOTE See 4.2.204.1.1 - 4.2.204.1.2 for the definition of each allowable value for has_type.

4.2.204.1.1 design_for_single_hull

the product data is applicable to a single hull.

4.2.204.1.2 design_for_multiple_hulls

the product data is applicable to multiple hulls in a class of ships.

4.2.204.2 ship_items

The ship_items specifies the constituent objects that are applicable to a particular Ship. See 4.3.69 for the application assertion.

4.2.204.3 units

The units specifies the reference to a set of pre-defined units for all types of measures that may appear in the ship model. Units may be either a Derived_unit or a Named_unit. Units may have no value contained in it or it may have one or more values, without any duplications.

NOTE The name of the ship is specified as a Definition in Ship_designation, where it may be versioned.

4.2.205 Ship_designation

A Ship_designation is a type of General_characteristics_definition (see 4.2.97) that specifies the identification given to the Ship in order that it can be categorized by any shipping related organization.

The data associated with a Ship_designation are the following:

— call_sign;

— description;

— flag_state;

— local_units;

— port_of_registration;

— ship_identification;

— ship_name.

4.2.205.1 call_sign

The call_sign specifies the unique lifecycle identifier assigned to the Ship by the flag_state for radio communication..

4.2.205.2 description

The description specifies details about the function of the Ship and additional information about the cargo carried. If the ship type is user defined, then the description delivers the information for the type of the Ship.

4.2.205.3 flag_state

The flag_state specifies the national authority with which the ship is registered.

4.2.205.4 local_units

The local_units specifies that a Ship_designation is not to define local units.

4.2.205.5 port_of_registration

The port_of_registration specifies the national home port of the ship. The port_of_registration lies within the jurisdiction of the flag_state.

4.2.205.6 ship_identification

The ship_identification specifies a general identifier unique to the Ship assigned during the classification process.

4.2.205.7 ship_name

The ship_name specifies the name of the Ship assigned by the owner.

4.2.206 Ship_material_property

A Ship_material_property is a type of Definition (see 4.2.56) that is the supertype for all materials used to specify the raw materials by physical properties as a definition for Item objects being made of a material.

The data associated with a Ship_material_property are the following:

— density;

— description;

— material_reference.

4.2.206.1 density

The density specifies the mass of the material per unit volume.

4.2.206.2 description

The description specifies the name used for that material.The description need not to be specified for a particular Ship_material_property.

4.2.206.3 material_reference

The material_reference specifies the External_reference to a source where more specific information about a material is available. The material_reference need not be specified for a particular Ship_-material_property. See 4.3.149 for the application assertion.

4.2.207 Shiptype

A Shiptype is a type of Functional_definition (see 4.2.96) that is the description of the function, purpose or mission a Ship is designed for. A Shiptype is either a Carrier (see 4.2.23), a Navy_ship (see 4.2.138), a Research_ship (see 4.2.191), or a Working_ship (see 4.2.278).

NOTE A Ship may have multiple functions. In this case several Entities of this type is to be defined and assigned to an instance of Ship. It is almost impossible to provide a complete list of ship types. Therefore only the major types are named explicitly, other Shiptype objects may be assigned via the user_defined type with clarification of the function specified in the description attribute.

The data associated with a Shiptype are the following:

— defined_for;

— description.

4.2.207.1 defined_for

The defined_for specifies the Ship for which the Shiptype is defined. There may be more than one defined_for for a Shiptype. See 4.3.150 for the application assertion.

4.2.207.2 description

The description attribute specifies a function, purpose, or mission that a Ship is designed for.

4.2.208 Shipyard_designation

A Shipyard_designation is a type of General_characteristics_definition (see 4.2.97) that specifies the identification given to the Ship by the shipbuilder.

The data associated with a Shipyard_designation are the following:

— local_units;

— role;

— shipyard;

— shipyard_new_building_id;

— shipyard_project_name.

4.2.208.1 local_units

The local_units specifies that a Shipyard_designation is not to define local units.

4.2.208.2 role

The role specifies the contractual obligation the shipyard has in relation to the Ship.

The value of the role is one of the following:

— prime;

— prime_build;

— prime_design;

— prime_repair;

— subcontractor.

NOTE See 4.2.208.2.1 - 4.2.208.2.5 for the definition of each allowable value for role.

4.2.208.2.1 prime

prime: the prime contractor for the Ship.

4.2.208.2.2 prime_build

prime_build: the prime contractor with contract responsibilities for manufacture of the Ship.

4.2.208.2.3 prime design

prime_design: the prime contractor with contract responsibility for the design of the Ship.

4.2.208.2.4 prime repair

prime_repair: the prime contractor with contract responsibilities for repair of the Ship.

4.2.208.2.5 subcontractor

subcontractor: a subcontractor for the Ship.

4.2.208.3 shipyard

The shipyard specifies the name and organizational details of the shipyard.

4.2.208.4 shipyard_new_building_id

The shipyard_new_building_id specifies the identifier for the Ship that is assigned by the shipyard after an order has been confirmed.The shipyard_new_building_id need not be specified for a particular Shipyard_designation.

4.2.208.5 shipyard_project_name

The shipyard_project_name specifies an identifier, for the Ship, that is assigned by the shipyard on receipt of an order for a new Ship or for bid proposal documentation on a new Ship.

4.2.209 Spacing_position

A Spacing_position is a location on one of the global coordinate axes of the Ship that is used as a reference point for any geometrical or structural item during the design and manufacture of the ship. Each Spacing_position is either a Longitudinal_position (see 4.2.129), a Transversal_position (see

4.2.253), a Vertical_position (see 4.2.265), a Spacing_position_with_offset (see 4.2.210), or a combination of Spacing_position_with_offset with one of the other three objects stated previously.

EXAMPLE Typically spacing positions are specified by LFR 123, TFR 10, 100, 100.1, A. In addition the distance to the global origin is defined, for instance, by 154.5 meters.

NOTE Figure 40 illustrates longitudunal, transversal, and vertical Spacing_position objects.

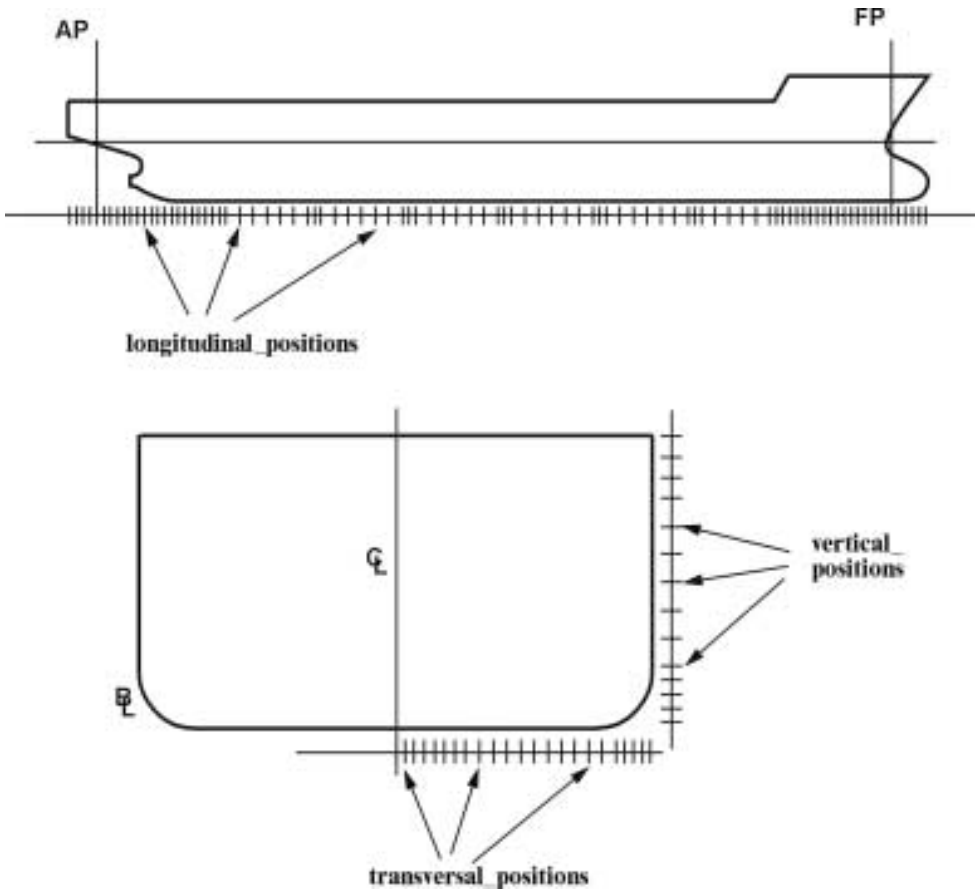


Figure 40 — Spacing positions

The data associated with a Spacing_position are the following:

— name;

— position;

— position_number.

4.2.209.1 name

The name specifies a label that is used to name the reference pont. The name need not be speified for a particular Spacing_position.

4.2.209.2 position

The position specifies the distance to the origin of the global coordinate system of the Ship. The axis on which the distance is measured depends on the type of Spacing_position.

4.2.209.3 position_number

The position_number specifies the numerical identification that is given to the Spacing_position.

4.2.210 Spacing_position_with_offset

A Spacing_position_with_offset is a type of Spacing_position (see 4.2.209) that is a position defined by an offset to an existing Spacing_position on one of the global coordinate axes of the Ship. It is used as a reference point for any geometrical or structural item during the design and manufacture of the Ship.

The data associated with a Spacing_position_with_offset are the following:

— offset;

— position;

— relating_spacing_position.

4.2.210.1 offset

The offset specifies the distance to the relating spacing position. The axis on which the distance is measured depends on the type of the relating Spacing_position.

4.2.210.2 position

The position specifies the distance to the origin of the global coordinate system of the Ship. The axis on which the distance is measured depends on the type of the relating Spacing_position.

4.2.210.3 relating_spacing_position

The relating_spacing_position specifies the Spacing_position from where the offset is taken to identify the Spacing_position_with_offset. See 4.3.151 for the application assertion.

4.2.211 Spacing_table

A Spacing_table is a type of Definition (see 4.2.56) that is a collection of Spacing_position objects (see 4.2.209) that defines a list of reference points along one of the coordinate axes of the Ship. Each Spacing_table may be one of the following: a Longitudinal_table (see 4.2.130), a Transversal_table (see 4.2.254), or a Vertical_table (see 4.2.266).

EXAMPLE A frame table is a type of Spacing_table where the frame positions would be longitudinal positions.

The data associated with a Spacing_table are the following:

— name;

— spacing_table_representations.

4.2.211.1 name

The name specifies the context specific identification for the Spacing_table. The name need not be specified for a particular Spacing_table.

4.2.211.2 spacing_table_representations

The spacing_table_representations specifies the positions that make up the table on the coordinate axis that are of interest. The spacing_table_representations need not be specified for a particular Spacing_table. There may be more than one spacing_table_representations for a Spacing_table. See 4.3.152 for the application assertion.

4.2.212 Spot_seam_weld

A Spot_seam_weld is a type of Welded_joint_design_definition (see 4.2.276) where the weld is performed by locally heating and simultaneously pressurizing through the electrode, with current and welding force concentrated in comparatively small area, spot or seam, of assembly formed by holding the lapped base metal with a properly formed electrode trip.

The data associated with a Spot_seam_weld are the following:

— weld_context.

4.2.212.1 weld_context

The weld_context specifies the indicator as to whether a Spot_seam_weld represents a spot weld or a seam weld.

The value of weld_context is one of the following:

— seam_weld;

— spot_weld.

NOTE See 4.2.212.1.1 - 4.2.212.1.2 for the definition of each allowable value for weld_context.

4.2.212.1.1 seam_weld

seam_weld: a weld performed by locally applying heat with electrode to a small line like area.

4.2.212.1.2 spot_weld

spot_weld: a weld performed by locally applying heat with electrode to a small point like area.

4.2.213 Square_tube_cross_section

A Square_tube_cross_section is a type of Non_circular_hollow_profile_cross_section (see 4.2.139) that is square or rectangular in shape and has a non-solid core.

NOTE ISO 657-14 [5] specifies dimensions for some standard square and rectangular hollow sections.

The data associated with a Square_tube_cross_section are the following:

— depth;

— wall_thk;

— width.

4.2.213.1 depth

The depth specifies the depth of the tube.

4.2.213.2 wall_thk

The wall_thk specifies the wall thickness.

4.2.213.3 width

The width specifies the width of the tube.

4.2.214 Station_table

A Station_table is a type of Longitudinal_table (see 4.2.130) that has positions that reference the location of stations that are located on the global X-axis.

NOTE Stations are used in the design process of a Ship and the station curves are curves of transversal sections through the ship hull. There are usually 20 stations, but the number can differ from shipyard to shipyard.

4.2.215 String_property_value

A String_property_value is a type of Property_value (see 4.2.184) that provides a value for a property that is a string value.

The data associated with a String_property_value are the following:

— val.

4.2.215.1 val

The val specifies the value of the string.

4.2.216 Structural_added_material_boundary_relationship

A Structural_added_material_boundary_relationship is a type of Structural_feature_relationship (see 4.2.227) that specifies the topological relationship between a Structural_added_material_feature (see 4.2.217) and the boundary of a Plate, Profile, or Corrugated_part.

The data associated with a Structural_added_material_boundary_relationship are the following:

— boundary_index;

— item_1.

4.2.216.1 boundary_index

The boundary_index specifies the index of the boundary in the boundary list of the Profile_design_-definition, Plate_design_definition, or Corrugated_part_design_definition, depending on which one is referenced by item_2, that is to have material added to its edge for manufacturing.

4.2.216.2 item_1

The item_1 specifies the relating item of the relationship is a Structural_added_material_feature (see 4.2.217). See 4.3.153 for the application assertion.

4.2.217 Structural_added_material_feature

A Structural_added_material_feature is a type of Structural_manufacturing_feature (see 4.2.228) that represents characteristics that describe the addition of material to a structural part edge to provide margins in joining the parts at later manufacturing stages and maintaining critical dimensions. This added material is also referred to as green material. Some of this added material may be trimmed off when the parts are joined. An added material feature will modify the topological representation of the part. The edge it is applied to will be superceded by a new set of edges that represent the extended part's boundary.

4.2.218 Structural_added_material_feature_design_definition

A Structural_added_material_feature_design_definition is a type of Feature_design_definition (see 4.2.86) that is one type of part Feature that represents characteristics that describe the addition of material to a structural part edge to provide margins in joining the parts at later manufacturing stages and maintaining critical dimensions. This added material is also referred to as green material. Some of this added material may be trimmed off when the parts are later joined. An added material feature will modify the topological representation of the part. The edge it is applied to will be superceded by a new set of edges that represent the extended part's boundary.

The data associated with a Structural_added_material_feature_design_definition are the following:

— added_material_length;

— defined_for.

4.2.218.1 added_material_length

The added_material_length specifies the measure of the amount of material that is to be added to a part edge. This dimension is measured normal to the part's edge on the moulded surface.

4.2.218.2 defined_for

The defined_for specifies the Structural_added_material_feature (see 4.2.217) that the Structural_-added_material_feature_design_definition is defined for. There may be more than one defined_for for a Structural_added_material_feature_design_definition. See 4.3.154 for the application assertion.

4.2.219 Structural_class_approval

A Structural_class_approval is a type of Structural_class_approval_result (see 4.2.220) that specifies a positive result of the assessment of a Structural_part (see 4.2.229). Each Structural_class_approval is either a Corrugated_part_approval (see 4.2.46), a Hull_cross_section_approval (see 4.2.105), a Plate_approval (see 4.2.156), or a Profile_aprroval (see 4.2.172).
EXAMPLE The assissment was based on the rules of a classification society and the part did pass the verification.

4.2.220 Structural_class_approval_result

A Structural_class_approval_result is a type of Approval_history (see 4.2.7) that specifies the result of the assessment of a Structural_part (see 4.2.229) against a set of rules. These rules could come from a classification society. Subtypes of this object indicate whether the part passed the verification or not. Each Structural_class_approval_result is either a Structural_class_approval (see 4.2.219), a Structural_class_rejection (see 4.2.222), a Hull_cross_section_result (see 4.2.108) or a Structural_-class_conditional_approval (see 4.2.221).

The data associated with a Structural_class_approval_result are the following:

— explanations.

4.2.220.1 explanations

The explanations specifies the reasons that led to the decision for approval or disapproval. There may not always be a reason specified, especially not in the case of approval. There may be more than one explanations for a Structural_class_approval_result. See 4.3.155 for the application assertion.

4.2.221 Structural_class_conditional_approval

A Structural_class_conditional_approval is a type of Structural_class_approval_result (see 4.2.220) that specifies a mainly positive result of the assessment of a Structural_part (see 4.2.229). The part did pass the verification under the condition that clearly specified changes will be incorporated.

The data associated with a Structural_class_conditional_approval are the following:

— explanations.

4.2.221.1 explanations

The explanations specifies the condition under which the approval is valid. At least one condition shall be given. There may be more than one explanations for a Structural_class_conditional_-approval. See 4.3.156 for the application assertion.

4.2.222 Structural_class_rejection

A Structural_class_rejection is a type of Structural_class_approval_result (see 4.2.220) that specifies a negative result of the assessment of a Structural_part (see 4.2.229).

EXAMPLE The part did not pass verification based on the rules of a classification society.

NOTE The rejection may be accompanied by an alternative design for the investigated part.

The data associated with a Structural_class_rejection are the following:

— explanations;

— proposed_alternative.

4.2.222.1 explanations

The explanations specifies the reason for the disapproval. At least one reason shall be given. There may be more than one explanations for a Structural_class_rejection. See 4.3.158 for the application assertion.

4.2.222.2 proposed_alternative

The proposed_alternative specifies the alternative Definitions that would pass the assessment. The collection may include different alternatives, or it may contain one alternative only the properties of which are spread across several Definitions. There may be more than one proposed_alternative for a Structural_class_rejection. See 4.3.157 for the application assertion.

4.2.223 Structural_cutout

A Structural_cutout is a type of Structural_feature (see 4.2.226) that specifies the nature of the cutout, which is the removal of material from a Structural_part (see 4.2.229) or a Structural_system (see 4.2.239). The global position of a Structural_cutout is given by the parent item of the Structural_feature. If a Structural_cutout is not to be at the origin of its parent, it shall be located relatively to its parent by using Local_co_ordinate_system. Each Structural_cutout is either a Corner_-cutout (see 4.2.42), an Edge_cutout (see 4.2.71), an Edge_feature (see 4.2.73), or an Interior_cutout (see 4.2.110).

NOTE 1 Complex cutouts may be constructed using the concept of composite features.

EXAMPLE 1 A plate is an example of a parent item.

NOTE 2 Structural_cutouts are distinguished by the number of part boundaries that they affect:

— Interior_cutout objects, no boundary crossed;

— Edge_cutout objects, 1 boundary crossed;

— Corner_cutout objects, at least 2 boundaries crossed.

Also Bevels, such as used for weld preparation, are Structural_cutouts; the Bevels modify a boundary of a Structural_part.

EXAMPLE 2 Cutouts may represent ratholes, the result of profile penetrations of plates, or others.

NOTE 3 Figure 41 illustrates plate cutouts. Figure 42 illustrates profile cutouts.

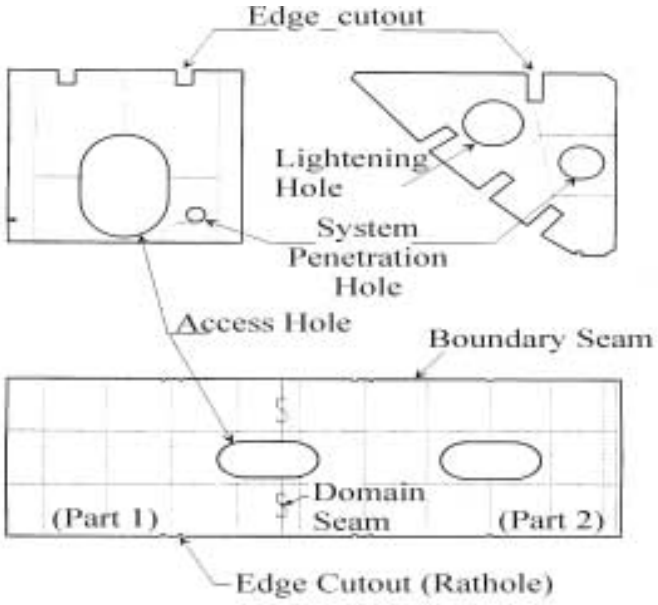


Figure 41 — Plate cutouts

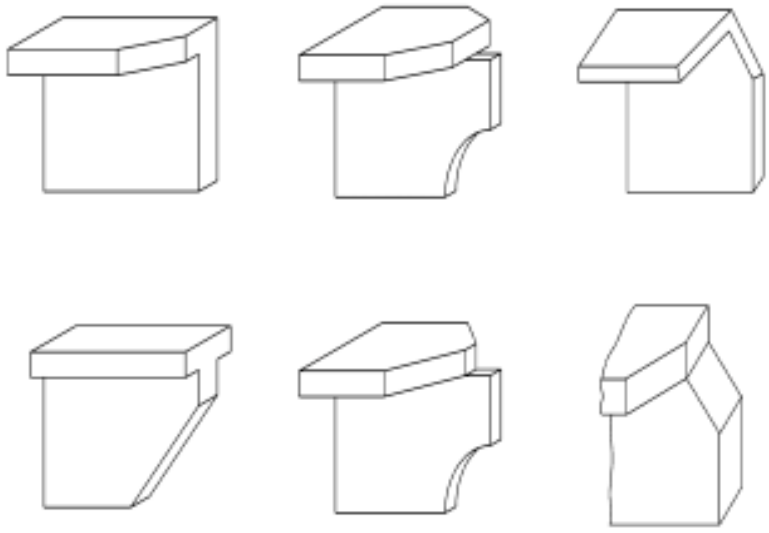


Figure 42 — Profile cutouts

4.2.224 Structural_cutout_boundary_relationship

A Structural_cutout_boundary_relationship is a type of Structural_feature_relationship (see 4.2.227) that specifies the topological relationship between a Structural_cutout (see 4.2.223) and the boundary of a Plate (see 4.2.155) or Panel_system (see 4.2.144). Each Structural_cutout_boundary_-relationship may be a Corner_cutout_boundary_relationship (see 4.2.43).

NOTE 1 This relationship may not be meaningful for Interior_cutout objects.

NOTE 2 For Corner_cutout objects the use of the subtype Corner_cutout_boundary_relationship is recommended as it allows references to both involved boundaries.

The data associated with a Structural_cutout_boundary_relationship are the following:

— boundary_index;

— item_1.

4.2.224.1 boundary_index

The boundary_index specifies the index of the referred boundary in the boundary list of the Panel_-system_design_definition or Plate_design_definition, depending on which one is referenced by item_2, that has the cutout.

4.2.224.2 item_1

The item_1 specifies the relating item of the relationship is a Structural_cutout. See 4.3.159 for the application assertion.

4.2.225 Structural_cutout_manufacturing_relationship

A Structural_cutout_manufacturing_relationship is a type of Structural_feature_relationship (see 4.2.227) that specifies the relationship between a Structural_cutout (see 4.2.223) and an edge in the outer_contour of a Plate_manufacturing_definition (see 4.2.159). This relationship is intended to provide Structural_cutout parameters to a manufacturing definition to allow production of a structural part. The feature is assumed to be applied to the entire referenced edge.

The data associated with a Structural_cutout_manufacturing_relationship are the following:

— item_1;

— outer_contour_index.

4.2.225.1 item_1

The item_1 specifies the relating item of a relationship is a Structural_cutout. See 4.3.160 for the application assertion.

4.2.225.2 outer_contour_index

The outer_contour_index specifies the index of the referred edge list of the Plate_manufacturing_-definition.

4.2.226 Structural_feature

A Structural_feature is a type of Feature (see 4.2.85) that specifies a Feature that can only be applied to structural elements. Each Structural_feature is either a Position_feature (see 4.2.166), a Structural_cutout (see 4.2.223), or a Structural_manufacturing_feature (see 4.2.228). If a Structural_feature represents the high level feature of a break-down into several low level features, it shall be instantiated as a complex entity instance of Structural_feature and Composite_feature.

If a Structural_feature is part of a set of features that make up a complex for example profile endcut, it shall not be instantiated as a complex subtype.

The data associated with a Structural_feature are the following:

— parent.

4.2.226.1 parent

The parent describes the element of the ship structure that the Structural_feature is applied to. See 4.3.161 for the application assertion.

NOTE Also System objects are Item objects and may, therefore, be parents to Structural_feature objects.

4.2.227 Structural_feature_relationship

A Structural_feature_relationship is a type of Feature_relationship (see 4.2.87) that defines the association of a Structural_feature (see 4.2.226) with another Item. Each Structural_feature_-relationship is either a Position_feature_relationship (see 4.2.168), a Structural_cutout_boundary_-relationship (see 4.2.225), or a Structural_added_material_boundary_relationship (see 4.2.216).

The data associated with a Structural_feature_relationship are the following:

— item_1.

4.2.227.1 item_1

The item_1 specifies the relating item of the relationship is a Structural_feature. See 4.3.162 for the application assertion.

4.2.228 Structural_manufacturing_feature

A Structural_manufacturing_feature is a type of Structural_feature (see 4.2.226) that can be applied to Structural_part objects to specify production engineering information or constraints that are to be applied to the design definition of the Structural_part for the purposes of manufacturing or assembly of the part. Each Structural_manufacturing_feature may be one of the following: a Structural_-added_material_feature (see 4.2.217), or a Structural_weld_shrinkage_allowance_feature (see 4.2.246).

The data associated with a Structural_manufacturing_feature are the following:

— parent.

4.2.228.1 parent

The parent specifies the element of the ship structure that the Structural_manufacturing_feature is applied to. See 4.3.163 for the application assertion.

4.2.229 Structural_part

A Structural_part is a type of Part (see 4.2.152) that forms a primary structural element in a ship. Each Structural_part is either a Corrugated_part (see 4.2.45), a Plate (see 4.2.155), or a Profile (see 4.2.171).

NOTE Structural parts are generally fabricated from purchased flat plate stock or rolled profile stock material which are cut, bent, formed, welded together, or drilled to produce the final physical part for installation in the ship.

4.2.230 Structural_part_connection_implementation

A Structural_part_connection_implementation is a type of Item (see 4.2.115) that contains the connectivity information between structural parts. When two or more parts are joined together, these requirements are fulfilled by a Structural_parts_connection_implementation that represents a physical realization of the connection. Each Structural_part_connection_implementation may be a Weld (see 4.2.270).

The data associated with a Structural_part_connection_implementation are the following:

— realization_of.

4.2.230.1 realization_of

The realization_of specifies the methods that represents a physical realization of the Structural_part_-connection_implementation. See 4.3.164 for the application assertion.

EXAMPLE The weld is a widely used joint method for shipbuilding.

4.2.231 Structural_part_design_definition

A Structural_part_design_definition is a type of Design_definition (see 4.2.58) that describes the abstract definition concept of any Structural_part (see 4.2.229) from the view of design. Each Structural_part_design_definition is either a Corrugated_part_design_definition (see 4.2.47), a Plate_desgin_definition (see 4.2.158), or a Profile_design_definition (see 4.2.176).

The data associated with a Structural_part_design_definition are the following:

— defined_for;

— mass;

— representations.

4.2.231.1 defined_for

The defined_for specifies Structural_part (see 4.2.229) for which the Structural_part_design_-definition is applicable. There may be more than one defined_for for a Structural_part_design_-definition. See 4.3.169 for the application assertion.

4.2.231.2 mass

The mass specifies the mass of the Structural_part. The mass need not be specified for a particular Structural_part_design_definition.

4.2.231.3 representations

The representations specifies that the Structural_part_design_definition is represented by Shape_-representation objects. The representations need not be specified for a particular Structural_part_-design_definition. There may be more than one representations for a Structural_part_design_-definition. See 4.3.165, 4.3.166, 4.3.167, 4.3.168 for the application assertions.

4.2.232 Structural_part_functional_definition

A Structural_part_functional_definition is a type of Functional_definition (see 4.2.96) that provides the capability to assign a function to a Structural_part (see 4.2.229). Each Structural_part_-functional_definition is either a Plate_functional_definition (see 4.2.159) or a Profile_functional_-definition (see 4.2.177).

EXAMPLE Plate or Profile.

NOTE 1 Figure 43 illustrates a deck composed from Plates and Profiles.

NOTE 2 Figure 44 illustrates a plain bulkhead composed from Plates and Profiles.

The data associated with a Structural_part_functional_definition are the following:

— defined_for;

— the_function.

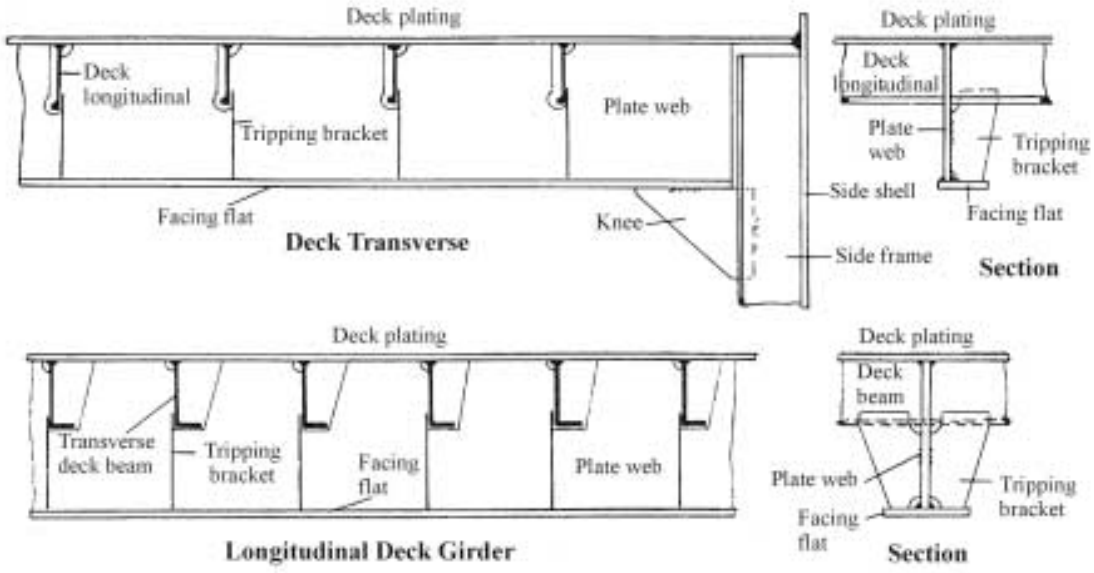


Figure 43 — Deck

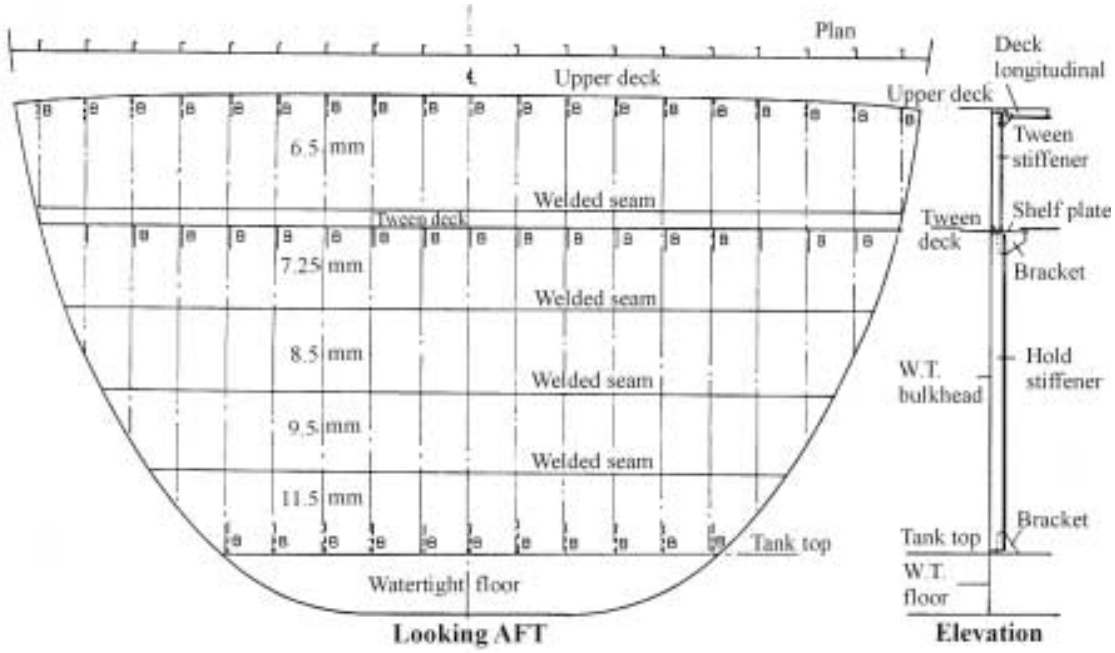


Figure 44 — Bulkhead

4.2.232.1 defined_for

The defined_for specifies one or multiple Structural_part objects that are defined by the Structural_-part_functional_definition. There may be more than one defined_for for a Structural_part_-functional_definition. See 4.3.170 for the application assertion.

4.2.232.2 the_function

The the_function specifies the purpose for the Structural_part_functional_definition.

The value of the the_function is one of the following:

— beam;

— bilge_plate;

— bracket;

— carling;

— clip;

— deck_beam;

— doubling_plate;

— face_plate;

— flange;

— frame;

— hold_pillar;

— longitudinal_stiffener;

— lug;

— pillar;

— shedding_plate;

— standard_plate;

— stiffener;

— transversal_stiffener;

— user_defined;

— watertight_clip;

— web.

NOTE See 4.2.232.2.1 - 4.2.232.2.21 for the definition of each allowable value for the_function.

4.2.232.2.1 beam

a structural member that is designed to support or carry a specified load. It is an athwartship or longitudinal member of the ship frame supporting the deck.

EXAMPLE A beam may be a I- beam, a wide flange beam, or a girder of welded steel plate.

4.2.232.2.2 bilge_plate

the curved shell plate that fits the bilge of the ship.

4.2.232.2.3 bracket

a steel plate connecting two structural members that strengthens the connection.

EXAMPLE A bracket may be a short piece of flanged plate, a tee bar, an angle bar, or a piece of flat plate that adds support at the joint.

4.2.232.2.4 carling

a short beam forming a portion of the framing about deck opening. Also called header when it supports the ends of interrupted deck beams.

4.2.232.2.5 clip

a short lengths of bar, generally angle, used to attach and connect the various members of the ship structure.

EXAMPLE A four to six-inch angle bar welded temporarily to floors, plates, webs, etc. It is used as a hold-fast which, with the aid of a bolt, pulls objects up close in fitting.

4.2.232.2.6 deck_beam

a structural member that is designed to support or carry a specified load on the deck and attached under the deck plating.

EXAMPLE A beam may be a I- beam, a wide flange beam, or a girder of welded steel plate.

4.2.232.2.7 doubling_plate

an extra plate secured to the original plating to add strength.

NOTE Often used to compensate for an opening in the structure.

4.2.232.2.8 face_plate

a reinforcing member attached to the free edge of a plate bracket to increase strength and resist buckling.

NOTE Also face plate is welded on to the flange of a girder or bracket to provide additional strength.

4.2.232.2.9 flange

the turned edge of a plate or girder, or in a rolled section, which acts to resist bending.

EXAMPLE The flange in a Wide Flange or a channel rolled section.

4.2.232.2.10 frame

a transverse rib that is a part of the skeleton of the ship.

NOTE Frames act as stiffeners, holding the outside plating in shape and maintaining the transverse form of the ship.

4.2.232.2.11 hold_pillar

a major structural support member in or under a large compartment area for cargo.

4.2.232.2.12 longitudinal_stiffener

a structural member that stiffens the plating fore-and-aft in the ship.

4.2.232.2.13 lug

a short plate with a hole attached to an item to facilitate lifting and handling.

4.2.232.2.14 pillar

a vertical structural support column.

4.2.232.2.15 shedding_plate

a plate added between corrugations of a corrugated bulkhead, angled down to avoid the collection of bulk cargoes.

4.2.232.2.16 standard_plate

a plate that is available, not a special order, from a supply vendor.

NOTE A standard plate is a plate of common size, shape, and material.

4.2.232.2.17 stiffener

a structural member that adds localized strength to a plate area.

NOTE A stiffener may be a tee beam, an angle beam, or a flat bar, any of which is welded to a plate.

4.2.232.2.18 transversal_stiffener

a structural member that stiffens the plating from port-to-starboard in the ship.

4.2.232.2.19 user_defined

the Structural_part_functional_definition is defined within the user_def_function attribute, by the user.

NOTE The user_defined option may only be used when the Structural_part_functional_definition cannot be defined by the other function codes.

4.2.232.2.20 watertight_clip

a flush or lapped plate used to close a penetration in a watertight boundary.

4.2.232.2.21 web

the portion of a beam that is located between two flanges.

EXAMPLE The web of an I-beam or that part of a web-frame between the shell plating and the internal flange.

4.2.233 Structural_part_joint

A Structural_part_joint is a type of Item_relationship (see 4.2.116) that represents a logical bridge between structural parts. The Structural_part_joint may decompose into lower level part joints. Each Structural_part_joint may be a Welded_joint (see 4.2.275).

The data associated with a Structural_part_joint are the following:

— joint_type;

— realization.

4.2.233.1 joint_type

The joint_type specifies an indicator which denotes the type of joint represented by a structural part connectivity.

The value of the joint_type is one of the following:

— fixed_joint;

— movable_joint.

NOTE See 4.2.233.1.1 - 4.2.233.1.2 for the definition of each allowable value for joint_type.

4.2.233.1.1 fixed_joint

a type of structural part joint that manifests itself as either a mechanical joint or a welded joint. The fixed joint results in a connection between structural parts that is rigid and allows no movement of the parts joined relative to one another.

4.2.233.1.2 movable_joint

a type of structural part joint where the parts to be joined are not constrained to fixed positions, but are allowed to move relative to one another, such as in the case of a hinge.

NOTE This type of joint is usually accomplished by placing a structural opening in both parts and holding them together with some sort of fastener set.

4.2.233.2 realization

The realization specifies the Structural_part_joint that is realized by Structural_part_connection_-implementation objects. See 4.3.164 for the application assertion.

4.2.234 Structural_part_joint_design_definition

A Structural_part_joint_design_definition is a type of Design_definition (see 4.2.58) that specifies the design aspects of a Structural_part_joint (see 4.2.233). Each Structural_part_joint_design_-definition may be a Welded_joint_design_definition (see 4.2.276).

NOTE Such as its tightness, and its joint_orientation.

The data associated with a Structural_part_joint_design_definition are the following:

— defined_for;

— joint_orientation;

— tightness.

4.2.234.1 defined_for

The defined_for specifies the Structural_part_joint (see 4.2.233) for which the Structural_part_-joint_design_definition is applicable. There may be more than one defined_for for a Structural_-part_joint_design_definition. See 4.3.171 for the application assertion.

4.2.234.2 joint_orientation

The joint_orientation specifies the identifier used to denote whether a structural part joint represents a face-face, a face-edge or an edge-edge joint. This identifier is based on the topological decomposition of a Structural_part_joint.

The value of the joint_orientation is one of the following:

— edge_edge;

— face_edge;

— face_face.

NOTE See 4.2.234.2.1 - 4.2.234.2.3 for the definition of each allowable value for joint_orientation.

4.2.234.2.1 edge_edge

the edge of one part is joined to the edge of the other.

EXAMPLE The side of one plate part joined with the side of another.

4.2.234.2.2 face_edge

the face of one part is joined to the edge of the other.

EXAMPLE The connection of a bracket to the web of a structural shape.

4.2.234.2.3 face_face

the face of one part is joined to the face of the other.

EXAMPLE In the case of that two plates lap over each other.

4.2.234.3 tightness

The tightness specifies the actual degree of tightness achieved by the structural part joint. The actual joint tightness may be greater than that required by the connection tightness.

The value of the tightness is one of the following:

— air_tight;

— fume_tight;

— non_tight;

— oil_tight;

— undefined;

— water_tight;

— weather_tight.

NOTE See 4.2.234.3.1 - 4.2.234.3.7 for the definition of each allowable value for tightness.

4.2.234.3.1 air_tight

an air tight boundary is one that will prevent the passage of air, gas, or fumes. Testing could be similar to that used for water tight test with no passage of liquid, a chalk test on doors and other closures to check contact or a pressure drop test.

NOTE The boundaries that are to be air, gas, and fume tight should in general have continuous weld at least one side.

4.2.234.3.2 fume_tight

a fume tight boundary is one that will prevent the passage of fumes, air, or gas. Testing could be similar to that used for water tight test with no passage of liquid, a chalk test on doors and other closures to check contact or a pressure drop test.

4.2.234.3.3 non_tight

there is no requirement to prevent the passage of gases or liquids in a non tight boundary.

4.2.234.3.4 oil_tight

an oil tight boundary is one that can withstand a head of liquid without deformation or damage. The boundary is subject to test to prove that it will not leak.

NOTE Oil tight is commonly used in Classification Society Rules and has both structure strength and tightness test requirements.

4.2.234.3.5 undefined

the tightness is not defined.

4.2.234.3.6 water_tight

a water tight boundary is one that will not pass water when subjected to a hose test. Generally, weld is continuous on one side. Door, bulkhead, deck, flat, and hatch are generally structures that are water tight.

4.2.234.3.7 weather_tight

the boundary test requirements would be the same as that for water test but some light leakage is expected. This is in general used for openings on the decks exposed to weather where water tightness is not required. It may be used for doors and some hatch covers.

4.2.235 Structural_part_manufacturing_definition

A Structural_part_manufacturing_definition is a type of Manufacturing_definition (see 4.2.131) that specifies the production lifecycle stage view of a structural part that represents characteristics and attributes directly related to the manufacturing of the part. The positive Z-direction shall indicate the top surface of the part.

NOTE 1 An earlier detailed design part stage view may or may not exist for the part. At this manufacturing stage, structural part features are identified that affect aspects of the part that have to do with processes that will be used to cut and form the part.

EXAMPLE Some examples of the kinds of features identified during this stage are edge grinds, layout marks, and part stock.

NOTE 2 Information relating the structural part to the raw material stock it will be manufactured from is also added during this stage. A 2D flat pattern representation may be defined for the part, enabling it to be nested and cut from flat raw material stock.

Each Structural_part_manufacturing_definition may be one of the following: a Plate_-manufacturing_definition (see 4.2.160) or a Profile_manufacturing_definition (see 4.2.178).

The data associated with a Structural_part_manufacturing_definition are the following:

— bottom_annotation;

— bottom_layout_marks;

— bottom_punch_points;

— defined_for;

— inner_contours;

— top_annotation;

— top_layout_marks;

— top_punch_points.

4.2.235.1 bottom_annotation

The bottom_annotation is the text that will be marked on the bottom surface of the part during manufacturing to provide part identification or indications of correct part orientation within the ship. A bottom_annotation may have no value contained in it or it may have one or more values, without any duplications. There may be more than one bottom_annotation for a Structural_part_-manufacturing_definition.

4.2.235.2 bottom_layout_marks

The bottom_layout_marks specifies the list of items in the representation that will be marked on the bottom_surface of the part during manufacturing to provide part indications of reference locations or attachments to other structural parts. Bottom_layout_marks also includes any symbols, such as crosses for hole drilling locations, arrows, and bevel sketches that will be marked on the part. A bottom_layout_marks may have no value contained in it or it may have one or more values, without any duplications. There may be more than one bottom_layout_marks for a Structural_part_-manufacturing_definition.

4.2.235.3 bottom_punch_points

The bottom_punch_points specifies the list of points that will be punched on the bottom surface of the part to provide marking to aid in manufacturing. A bottom_punch_points may have no value contained in it or it may have one or more values, without any duplications. There may be more than one bottom_punch_points for a Structural_part_manufacturing_definition.

4.2.235.4 defined_for

The defined_for specifies the Structural_part_manufacturing_definition points to Structural_part objects only. There may be more than one defined_for for a Structural_part_manufacturing_-definition. See 4.3.172 for the application assertion.

4.2.235.5 inner_contour

The inner_contour specifies items in the representation that represent holecuts in interior_features that will be burned from the part during manufacturing. An inner_contour may have no value contained in it or it may have one or more values, without any duplications. There may be more than one inner_contour for a Structural_part_manufacturing_definition.

4.2.235.6 top_annotation

The top_annotation is the text that will be marked on the top surface of the part during manufacturing to provide part identification or indications of correct part orientation within the ship. A top_annotation may have no value contained in it or it may have one or more values, without any duplications. There may be more than one top_annotation for a Structural_part_manufacturing_-definition.

4.2.235.7 top_layout_marks

The top_layout_marks specifies the list of items in the representation that will be marked on the top surface of the part during manufacturing to provide part indications of reference locations or attachments to other structural parts. Top_layout_marks also includes any symbols, such as crosses for hole drilling locations, arrows, and bevel sketches that will be marked on the part. A top_layout_marks may have no value contained in it or it may have one or more values, without any duplications. There may be more than one top_layout_marks for a Structural_part_manufacturing_-definition.

4.2.235.8 top_punch_points

The top_punch_points specifies the list of points that will be punched on the top surface of the part to provide marking to aid in manufacturing. A top_punch_points may have no value contained in it or it may have one or more values, without any duplications. There may be more than one top_punch_points for a Structural_part_manufacturing_definition.

4.2.236 Structural_part_penetration_relationship

A Structural_part_penetration_relationship is a type of Item_relationship (see 4.2.116) that specifies a Structural_cutout (see 4.2.223) that results from the fact that a Part or System penetrates a Structural_part (see 4.2.229). The Structural_part_penetration_relationship will normally result in one or several cutouts for the involved elements.

EXAMPLE This relationship may be used to describe that a pipe penetrates a profile or that a profile cuts through the edge of a plate.

The data associated with a Structural_part_penetration_relationship are the following:

— item_1;

— penetration_result.

4.2.236.1 item_1

The item_1 specifies the Structural_part that is the penetrated Item of the two Item objects involved in the Structural_part_penetration_relationship. See 4.3.174 for the application assertion.

NOTE Item_1 will become the parent of a cutout feature, as this relationship results in a cutout.

4.2.236.2 penetration_result

The penetration_result specifies the set of Structural_cutout objects that this penetration results in. There may be more than one penetration_result for a Structural_part_penetration_relationship. See 4.3.173 for the application assertion.

NOTE Normally the result would be just one cutout. However, it is possible that both involved parts that may get cutouts. The parent attribute in Structural_cutout tells which part gets which cutout. Item_2 is the Item that is considered the penetrating one, if it is possible to distinguish penetrated and penetrating.

4.2.237 Structural_part_relationship

A Structural_part_relationship is a type of Part_relationship (see 4.2.154) that describes the association of a Structural_part (see 4.2.229) with another structural item. The structural item can be one of a Structural_feature (see 4.2.226), a Structural_part (see 4.2.229), or a Structural_system (see 4.2.239). Each Structural_part_relationship is either a Profile_relationship (see 4.2.180), a Plate_-relationship (see 4.2.162), or a Structural_part_symmetry_relationship (see 4.2.238).

The data associated with a Structural_part_relationship are the following:

— item_1.

4.2.237.1 item_1

The item_1 specifies that a Structural_part_relationship relates a Structural_part with something else. See 4.3.175 for the application assertion.

4.2.238 Structural_part_symmetry_relationship

A Structural_part_symmetry_relationship is a type of Structural_part_relationship (see 4.2.237) that specifies that two Structural_part (see 4.2.229) objects are symmetric with respect to a given plane.

NOTE This relationship is only an informal one for those applications that can use this information. No implicit information is intended to be carried, especially this is no short-cut for the definition of one of the related Structural_part objects using the other one.

The data associated with a Structural_part_symmetry_relationship are the following:

— item_2;

— mirroring_plane.

4.2.238.1 item_2

The item_2 specifies that a Structural_part_symmetry_relationship is to be between two Structural_-part objects. See 4.3.176 for the application assertion.

4.2.238.2 mirroring_plane

The mirroring_plane specifies the plane the symmetry is with respect to.

4.2.239 Structural_system

A Structural_system is a type of System (see 4.2.249) that provides information and capabilities common to all types of Structural_system objects. The Item objects carried by a Structural_system are restricted to be of type Structural_system or Structural_part (see 4.2.229). Each Structural_-system may be one of the following: a Built_profile (see 4.2.19), a Corrugated_structure (see 4.2.49), a Panel_system (see 4.2.144), or a Plate_strake (see 4.2.164).

NOTE Figure 47 is a part of the Structural_system of a ship. Figure 45 and Figure 46 show a doublebottom section as a Structural_system.

The data associated with a Structural_system are the following:

— the_class;

— user_def_class.

4.2.239.1 the_class

The the_class specifies the type of structural system within the ship. The structure class allows for a subdivision into different levels of structural systems.

The value of the_class is one of the following:

— block_class;

— built_strake_class;

— corrugated_structure_class;

— general_structural_system_class;

— panel_system_class;

— plate_strake_class;

— section_class;

— user_defined_class.

NOTE See 4.2.239.1.1 - 4.2.239.1.8 for the definition of each allowable value for the_class.

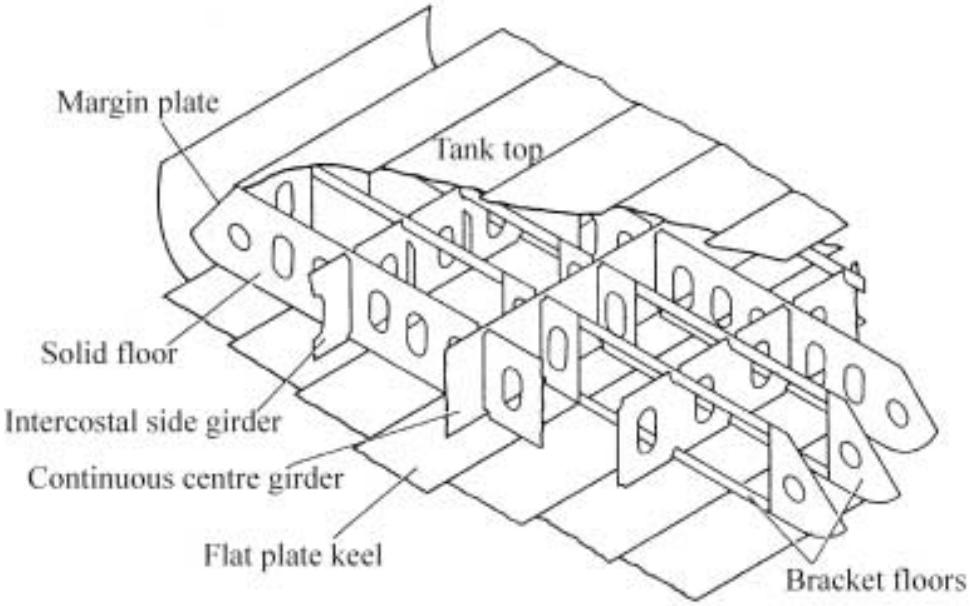


Figure 45 — Double bottom as section

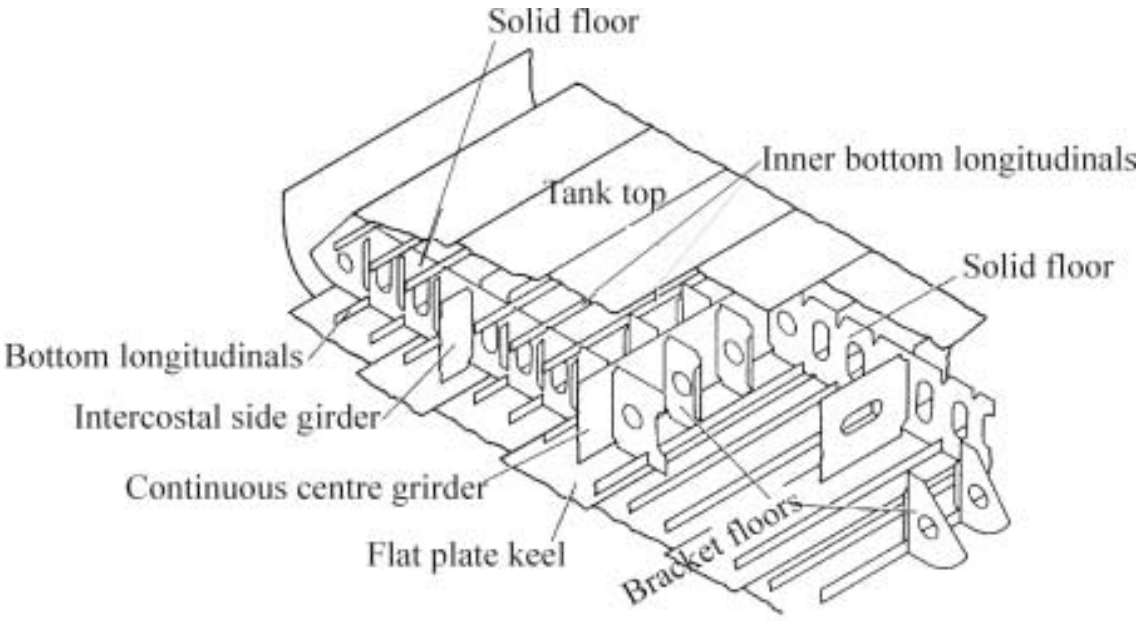


Figure 46 — Double bottom

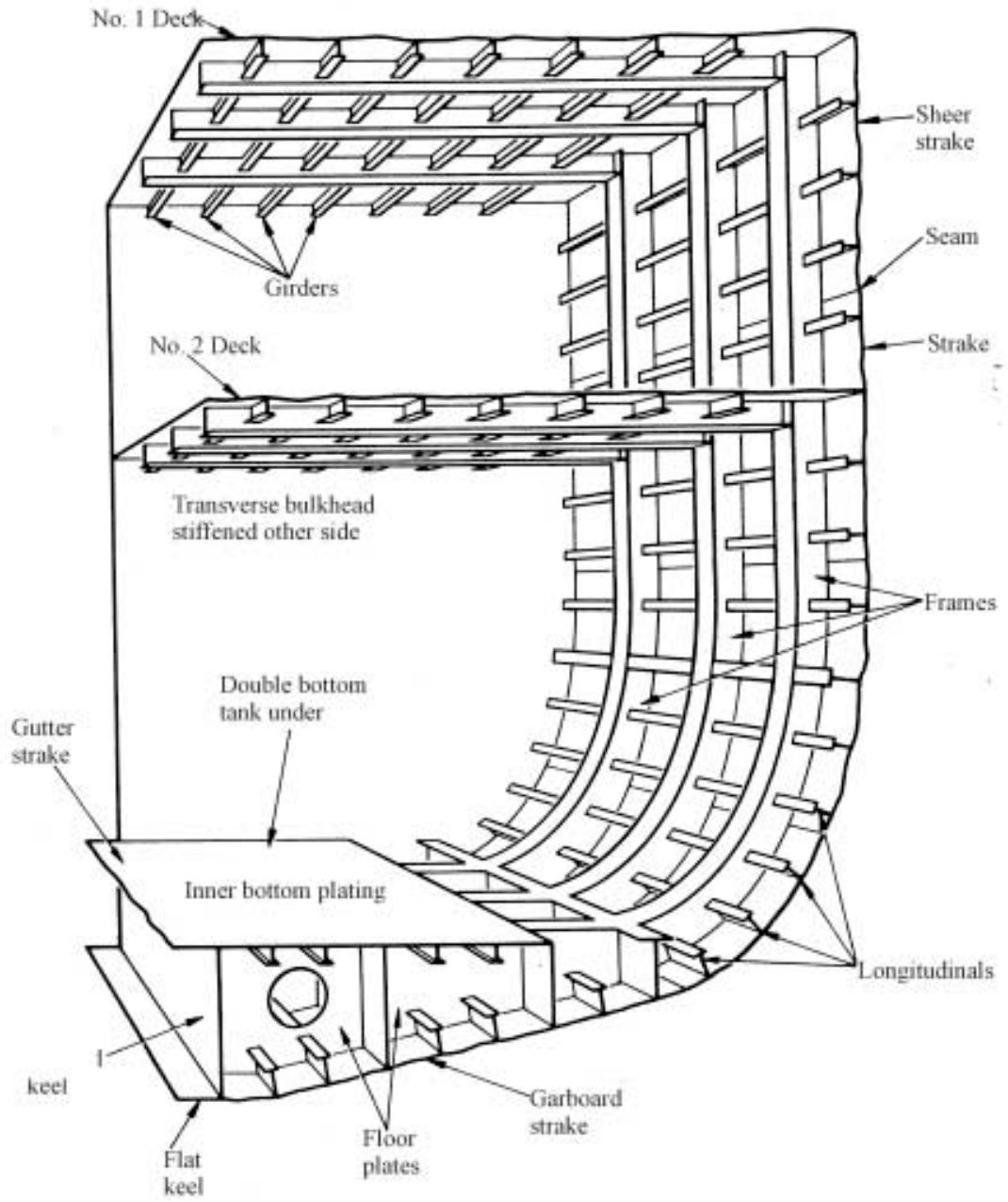


Figure 47 — Structural_system

4.2.239.1.1 block_class

an aggregate of other sub-blocks or Structural_part objects.

NOTE Also referred to as constructional group.

4.2.239.1.2 built_profile_class

a Structural_system that is made of at least two Structural_part objects, Plate objects or Profile objects, by an aggregation process; can nevertheless be used like a profile.

4.2.239.1.3 corrugated_structure_class

a special block that consists of only Corrugated_part objects and Plate objects.

4.2.239.1.4 general_structural_system_class

a class of general structural system.

4.2.239.1.5 panel_system_class

a special block that consists of zero to many Plate objects reinforced by Profile objects running in the same main direction; the Plate objects lie in the same planar or non-planar moulded surface.

4.2.239.1.6 plate_strake_class

a special block that consists of only Plate objects.

4.2.239.1.7 section_class

an aggregate of other sub-sections, blocks and or Structural_part objects.

4.2.239.1.8 user_defined_class

a class that is user defined.

4.2.239.2 user_def_class

The user_def_class specifies the user defined class of the structural system to be used in case the_-class value is user_defined_class. The user_def_class need not to be specified for a particular Structural_system.

4.2.240 Structural_system_adjacency_relationship

A Structural_system_adjacency_relationship is a type of Structural_system_relationship (see 4.2.244) that defines a boundary of a Structural_system (see 4.2.239) by specifying a boundary of another Structural_system that is adjacent to the initial Structural_system. The referred boundary is therefore shared by the related Structural_system objects.

The data associated with a Structural_system_adjacency_relationship are the following:

— boundary_index;

— displacement;

— item_2.

4.2.240.1 boundary_index

The boundary_index specifies the index of the referred boundary in the boundary list of the relevant Structural_system_design_definition (see 4.2.241) subtype associated with item_2.

4.2.240.2 displacement

The displacement specifies the offset orthogonal off the boundary referred to by boundary_index. The displacement need not be specified for a particular Structural_system_adjacency_relationship. There may be more than one displacement for a Structural_system_adjacency_relationship.

NOTE This is to cover the fact that the two panels are just virtually adjacent but with a gap in between. This allows two Structural_system objects not only to share boundaries, displacement = 0, but to have parallel boundaries where a change to the related boundary automatically changes the dependent one.

4.2.240.3 item_2

The item_2 specifies that the bound Item is to be of type Structural_system. See 4.3.177 for the application assertion.

4.2.241 Structural_system_design_definition

A Structural_system_design_definition is a type of System_design_definition (see 4.2.250) that describes the definition concept of any Structural_system (see 4.2.239) from the view of design.

The data associated with a Structural_system_design_definition are the following:

— defined_for;

— representations;

— tightness;

— user_defined_tightness.

4.2.241.1 defined_for

The defined_for specifies the Structural_system objects that are defined by the Structural_system_-design_definition. There may be more than one defined_for for a Structural_system_design_-definition. See 4.3.182 for the application assertion.

4.2.241.2 representations

The representations specifies that the generic shape of the Structural_system_design_definition. is represented by Shape_representation objects. The representations need not be specified for a particular Structural_system_design_definition. There may be more than one representations for a

Structural_system_design_definition. See 4.3.178, 4.3.179, 4.3.180, 4.3.181 for the application assertions. The valid shape representations are the following:

— Advanced_brep_shape (see 4.2.1);

— Edge_based_wireframe_shape (see 4.2.70);

— Geometrically_bounded_wireframe_shape (see 4.2.98);

— Non_manifold_surface_shape (see 4.2.140).

4.2.241.3 tightness

The tightness specifies the tightness requirements for a Structural_system.

The value of the tightness is one of the following:

— air_tight;

— fume_tight;

— gas_tight;

— non_tight;

— oil_tight;

— unspecified;

— user_defined;

— water_tight;

— weather_tight.

NOTE See 4.2.241.3.1 - 4.2.241.3.9 for the definition of each allowable value for tightness.

4.2.241.3.1 air_tight

an air tight boundary is one that will prevent the passage of air, gas, or fumes.

NOTE 1 Testing could be similar to that used for water tight test with no passage of liquid, a chalk test on doors and other closures to check contact or a pressure drop test.

NOTE 2 The boundaries that are to be air, gas, and fume tight should in general have continuous weld at least one side.

4.2.241.3.2 fume_tight

a fume tight boundary is one that will prevent the passage of fumes, air, or gas.

NOTE Testing could be similar to that used for water tight test with no passage of liquid, a chalk test on doors and other closures to check contact or a pressure drop test.

4.2.241.3.3 gas_tight

a gas tight boundary is one that will prevent the passage of gas, fumes, or air.

NOTE Testing could be similar to that used for water tight test with no passage of liquid, a chalk test on doors and other closures to check contact or a pressure drop test.

4.2.241.3.4 non_tight

there is no requirement to prevent the passage of gases or liquids in a non tight boundary.

4.2.241.3.5 oil_tight

an oil tight boundary is one that can withstand a head of liquid without deformation or damage. The boundary is subject to test to prove that it will not leak.

NOTE Oil tight is commonly used in Classification Society Rules and has both structure strength and tightness test requirements.

4.2.241.3.6 unspecified

the tightness is not defined.

4.2.241.3.7 user_defined

the tightness is defined by the user_defined_tightness attribute.

4.2.241.3.8 water_tight

a water tight boundary is one that will not pass water when subjected to a hose test.

NOTE Generally, weld is continuous on one side. Door, bulkhead, deck, flat, and hatch are generally structures that are water tight.

4.2.241.3.9 weather_tight

the boundary test requirements would be the same as that for water test but some light leakage is expected.

NOTE This is in general used for openings on the decks exposed to weather where water tightness is not required. It may be used for doors and some hatch covers.

4.2.241.4 user_defined_tightness

The user_defined_tightness specifies the user defined tightness. The user_defined_tightness need not be specified for a particular Structural_system_design_definition.

4.2.242 Structural_system_functional_definition

A Structural_system_functional_definition is a type of Functional_definition (see 4.2.96) that defines the functional role of a Structural_system (see 4.2.239); the role may be a pre-defined one or may be user-defined. Each Structural_system_functional_definition may be one of the following: a Corrugated_structure_functional_definition (see 4.2.52), a Panel_system_functional_definition (see 4.2.148), or a Plate_strake_functional_definition (see 4.2.165).

The data associated with a Structural_system_functional_definition are the following:

— defined_for;

— the_function.

4.2.242.1 defined_for

The defined_for specifies one or multiple Structural_system objects that are defined by the Structural_system_functional_definition. There may be more than one defined_for for a Structural_-system_functional_definition. See 4.3.183 for the application assertion.

4.2.242.2 the_function

The the_function specifies the name of a role that a Structural_system may play in a ship.

The value of the function is one of the following:

— accommodation_area;

— accomodation_deck;

— aft_ship;

— bottom;

— bracket;

— bulkhead;

— cargo_area;

— collision_bulkhead;

— cross_tie;

— deck;

— deck_beam;

— deck_house;

— deck_in_superstructure;

— double_bottom;

— double_shell;

— duct_keel;

— engine_area;

— engine_foundation;

— floor;

— fore_ship;

— frame;

— girder;

— hatch_cover;

— hatchway_coaming;

— hatchway_endcoaming;

— hatchway_sidecoaming;

— hold_bulkhead;

— hopper;

— inner_bottom;

— inner_shell;

— keel;

— longitudinal_bulkhead;

— longitudinal_girder;

— lower_boom;

— machinery_casing;

— main_deck;

— mid_ship;

— navigation_deck;

— outer_shell;

— platform_deck;

— plating;

— sheer_strake;

— ship_structure;

— stern_frame;

— stool;

— strength_bulkhead;

— strength_deck;

— stringer;

— superstructure;

— superstructure_aft_bulkhead;

— superstructure_front_bulkhead;

— superstructure_side_bulkhead;

— tank_bottom;

— tank_bulkhead;

— tank_side;

— tank_top;

— transom;

— transversal_bulkhead;

— transverse_floor;

— transverse_web_frame;

— upper_boom;

— user_defined;

— vertical_web_frame;

— wall;

— wash_bulkhead;

— weather_deck;

— web_frame;

— wing_bulkhead.

NOTE See 4.2.242.2.1 - 4.2.242.2.67 for the definition of each allowable value for the_function

4.2.242.2.1 accommodation_area

a deckhouse or a superstructure where berthing for passengers and crew is located.

4.2.242.2.2 accommodation_deck

a deck located in the accommodation area.

4.2.242.2.3 aft_ship

a portion of the ship which is aft of the midship section.

4.2.242.2.4 bottom

a bottom shell in general.

NOTE Figure 48 is an example of a single bottom.

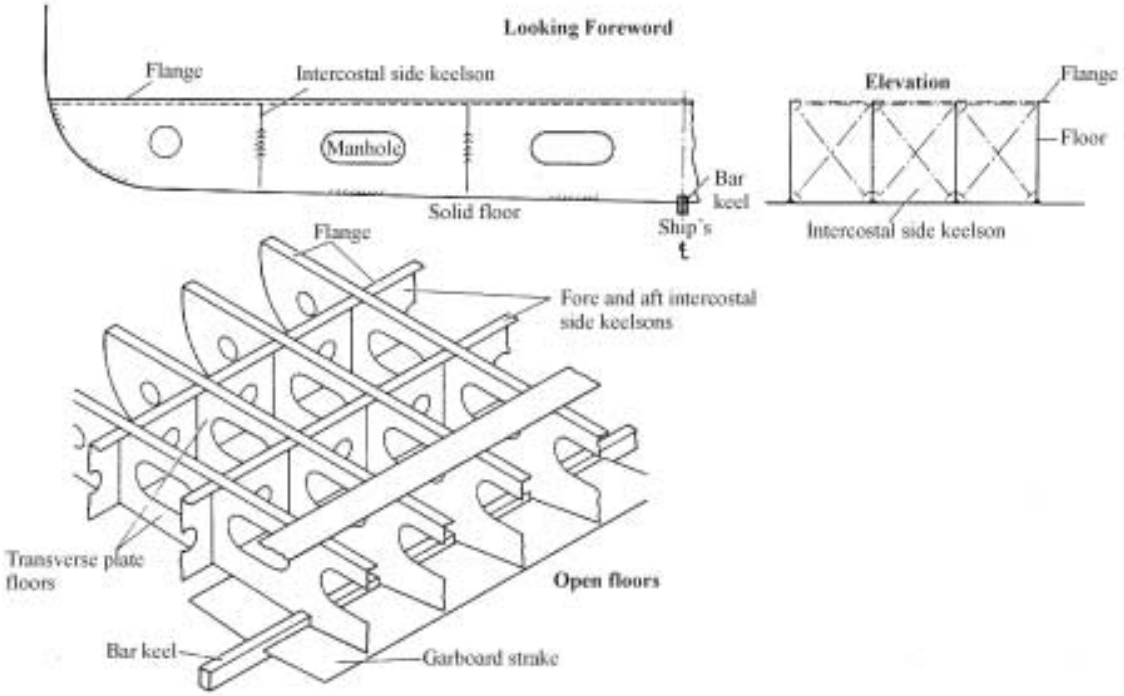


Figure 48 — Single bottom

4.2.242.2.5 bracket

a flanged plate or a regular plate used to provide rigid connection between two structural members. See also 4.2.230.2.3

4.2.242.2.6 bulkhead

a term applied to any of the partition wall used for subdividing the interior of a ship into various compartments.

NOTE Bulkhead may be watertight, structural which contributes to the strength of the ship, or non-structural which provides partition in the accommodation area. See also 3.8.6 and Figure 50.

4.2.242.2.7 cargo_area

a portion of the ship designated for the storage of the ship cargo.

4.2.242.2.8 collision_bulkhead

the bulkhead located in the bow area of a ship, which extends from the bottom of the ship to the weather deck, designed to provide watertight integrity in the event of bow damage, and also the foremost athwartship bulkhead in a ship.

NOTE See Figure 54.

4.2.242.2.9 cross_tie

an orthogonal structural member connecting two parallel beams or girders which provides extra strength to the deck.

4.2.242.2.10 deck

a platform in a ship corresponding to a floor in a building. It is the plating, planking, or covering of any tier of beams either in the hull or superstructure of a ship.

NOTE Figure 49 is an example of a DECK.

4.2.242.2.11 deck_beam

an athwartship or longitudinal horizontal structural member, usually a rolled shape, supporting a deck or flat.

4.2.242.2.12 deck_house

an enclosed structure on or above the weather deck that does not extend from side to side of the ship.

4.2.242.2.13 deck_in_superstructure

a deck in a superstructure.

4.2.242.2.14 double_bottom

a term applied to the space between the inner and outer bottom skins of a vessel.

EXAMPLE Double bottom is used for ballast water, fuel oil, etc.,

NOTE Also applied to indicate that a ship has a complete inner or extra envelop of watertight bottom plating. Figure 50 is an example of a double bottom.

4.2.242.2.15 double_shell

a term used to describe a double hull ship, designed so that damage in the outer hull will not affect the structural integrity of the ship or result in a loss of cargo.

NOTE Figure 51 is an example of a double_shell.

4.2.242.2.16 duct_keel

an area within the double bottom, bounded by watertight longitudinal plate girders, in which systems are typically run. Also it is called a pipe tunnel.

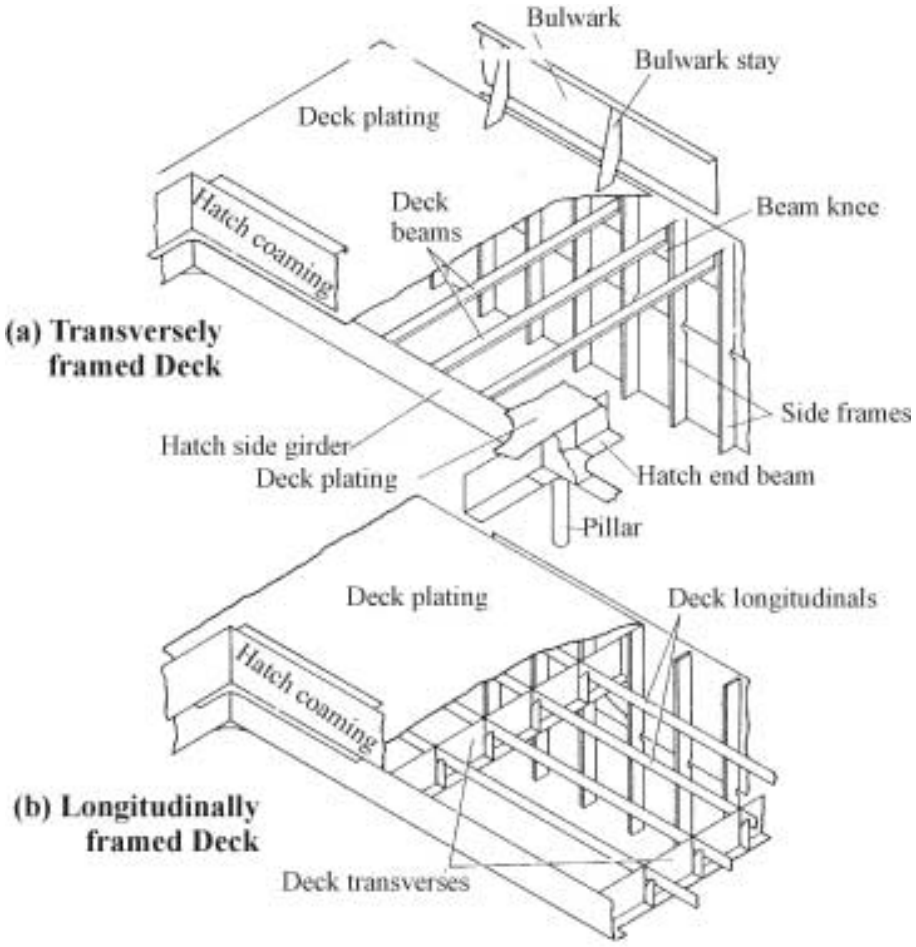


Figure 49 — Deck in three dimensions

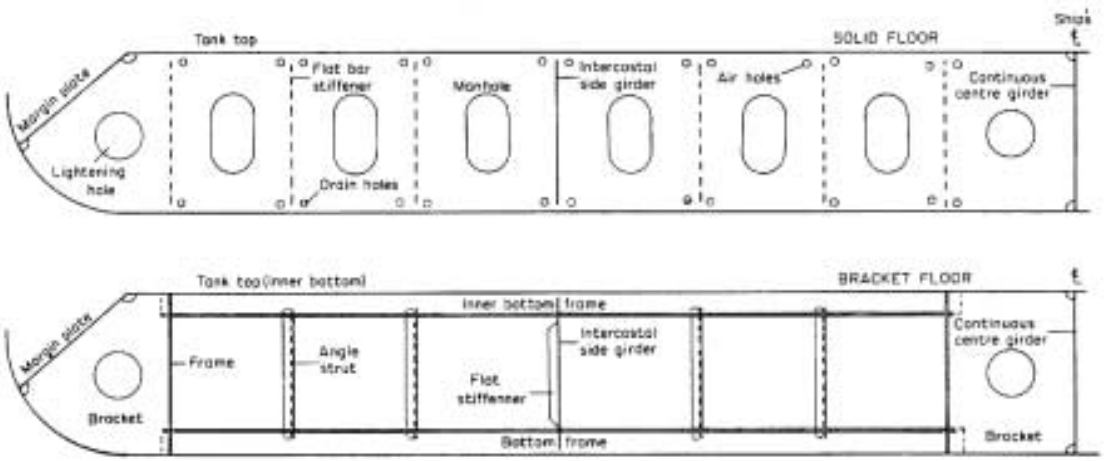


Figure 50 — Double bottom in two dimensions

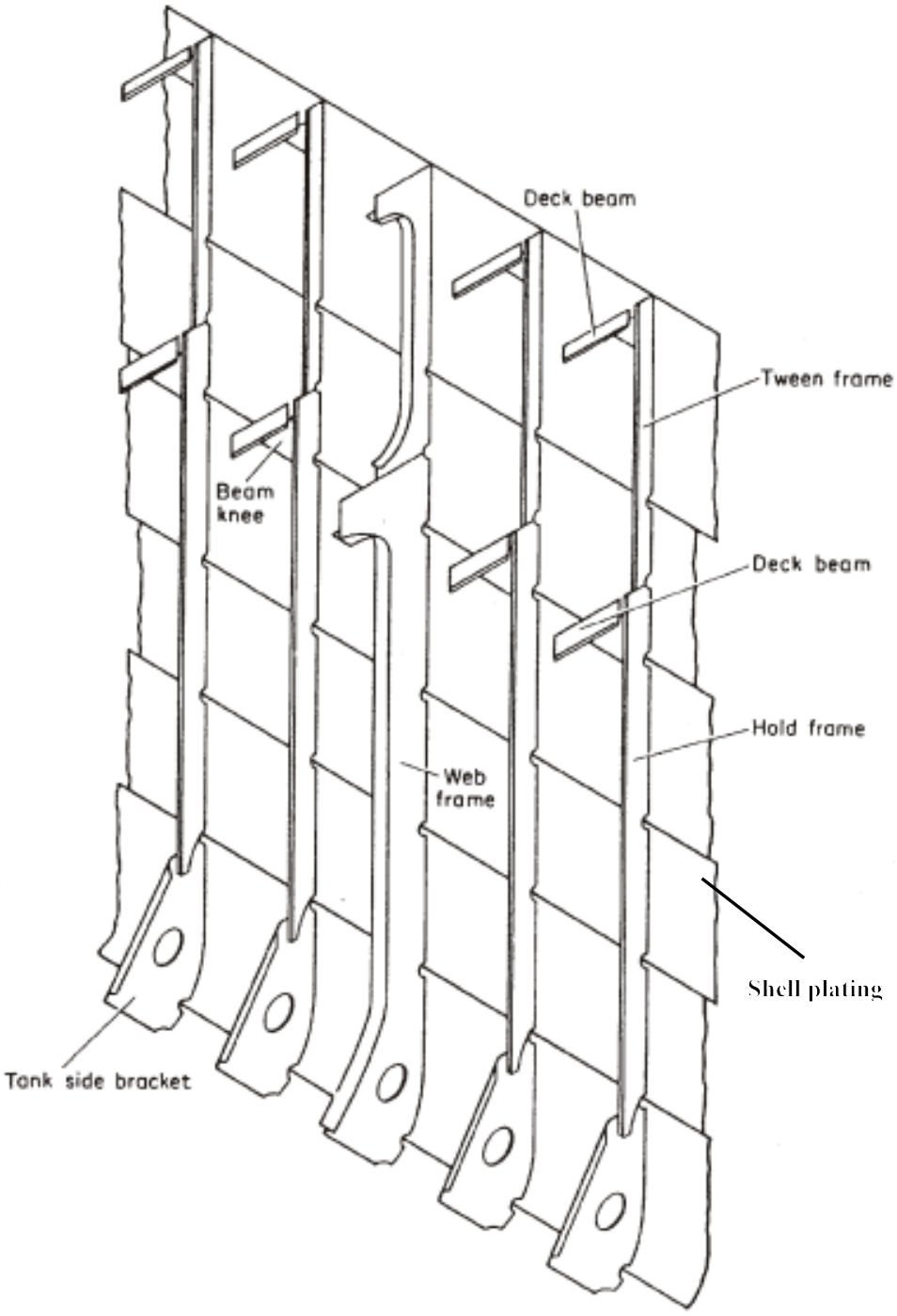


Figure 51 — Shell

4.2.242.2.17 engine_area

typically the ship engine room or machinery space, where the ship's main propulsion machinery is located.

4.2.242.2.18 engine_foundation

a heavy plate stiffened structure providing structural support for the main propulsion machinery, typically mounted to the top of the inner bottom structure.

4.2.242.2.19 floor

a plate used vertically in the bottom of a ship usually on every frame, to deepen it, and running athwartship from bilge to bilge.

NOTE Transverse plate structures typically located in the double bottom, used to provide subdivision of the double bottom and support bottom shell and inner bottom longitudinal members.

4.2.242.2.20 fore_ship

a term used indicating portions or part of a ship at or adjacent to the bow. Also applied to that portion and parts of the ship lying between the midship section and bow of the ship.

NOTE 1 Figure 52 is an example for a FORE SHIP in two dimensions.

NOTE 2 Figure 53 is an example for a FORE SHIP in three dimensions.

NOTE 3 Figure 54 is an example for a bow.

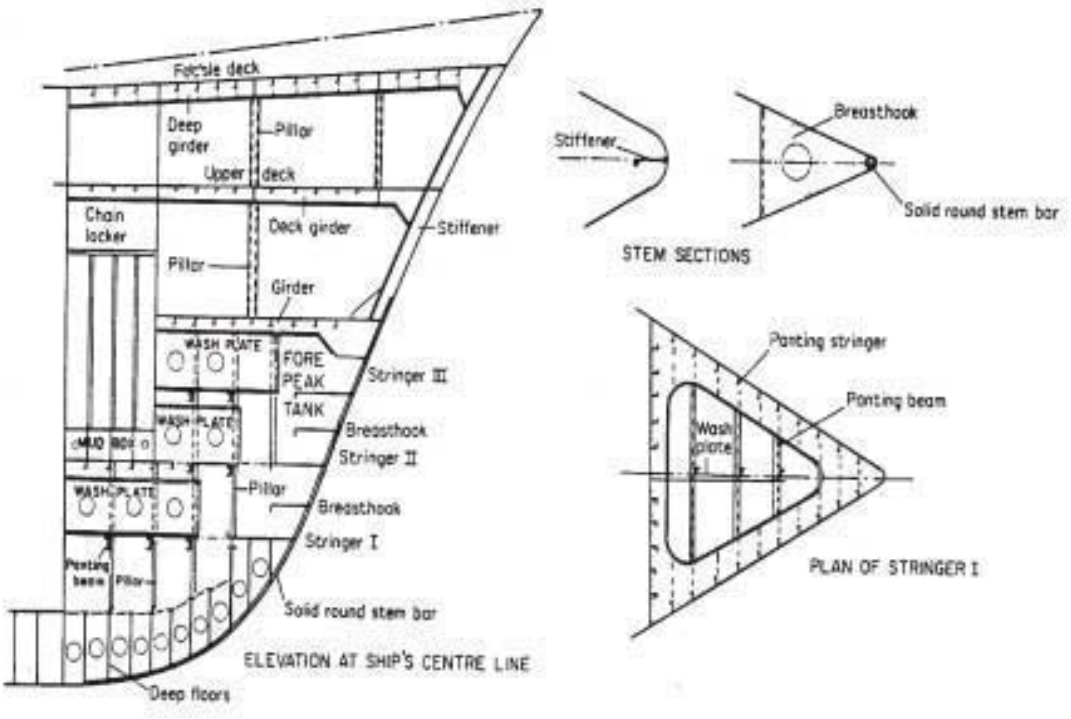


Figure 52 — Fore ship in two dimensions

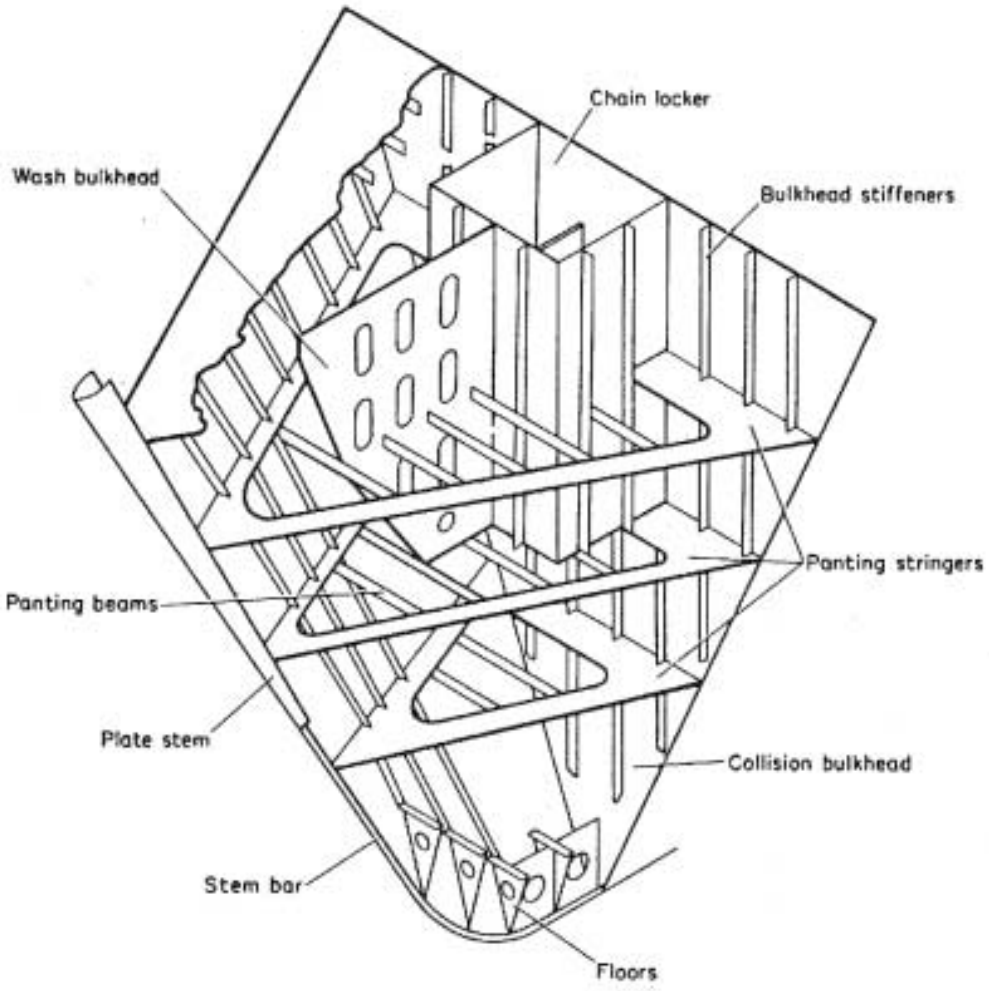


Figure 53 — Fore ship in three dimensions

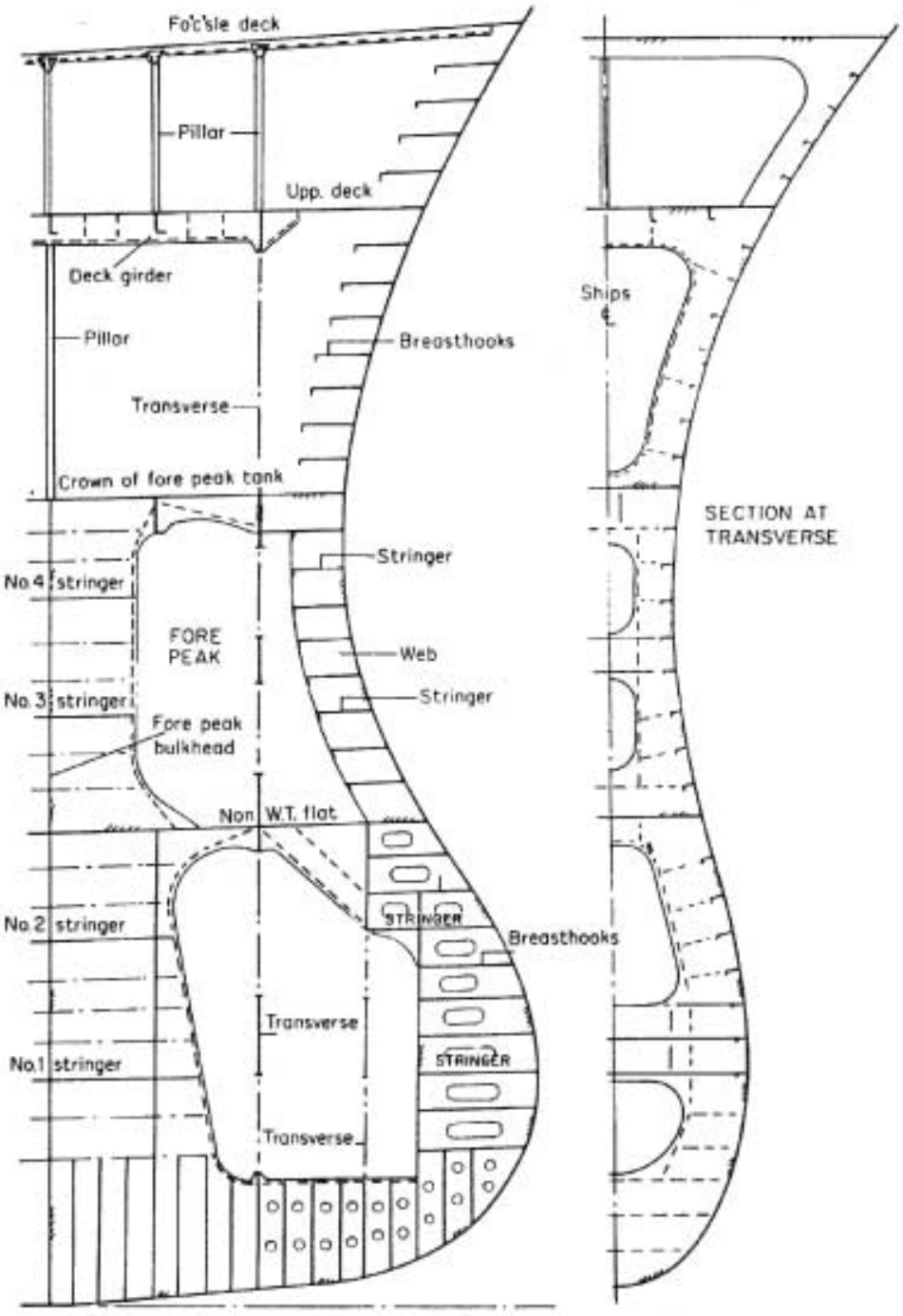


Figure 54 — Bow

4.2.242.2.21 frame

a term generally used to designate one of the transverse ribs that make up the skeleton of the ship. The frames act as stiffeners, holding the side shell plating in shape and maintaining transverse form of the ship.

4.2.242.2.22 girder

a primary structural support member which runs longitudinally in way of decks, shell, and bulkheads.

NOTE A girder provides support and rigidity to smaller and more closely spaced beams.

4.2.242.2.23 hatch_cover

a structural closure provided in way of large deck openings provided for hoisting freight or machinery in and out, and for access to the hold or machinery spaces.

4.2.242.2.24 hatchway_coaming

the four vertical web pieces in the frame of a hatchway are collectively called the hatchway coaming.

4.2.242.2.25 hatchway_endcoaming

athwartship vertical-web pieces at the forward and aft ends of the hatchway that land on the deck beams.

4.2.242.2.26 hatchway_sidecoaming

longitudinal vertical-web pieces at the sides of the hatchway that rest on the hatch carlings against which the deck beams abut in way of hatchway.

4.2.242.2.27 hold_bulkhead

a partition wall, either longitudinal or transverse, used to partition or subdivide the hold.

4.2.242.2.28 hopper

a dry cargo hold space in the bottom with sloping bulkhead plates on all four sides to facilitate directing the bulk cargoes towards the unloading system.

4.2.242.2.29 inner_bottom

an inner or extra envelope of watertight bottom plate in a ship with double bottom. Plate forming the top of the double bottom, also called tank top.

NOTE A double bottom is usually fitted in large ships extending from bilge to bilge and nearly the whole length fore and aft. Where there are two bottoms, the outer is called the “shell”, the inner the “inner bottom”.

4.2.242.2.30 inner_shell

in a double hull vessel, it is the inner shell plating of the hull.

4.2.242.2.31 keel

the principal fore-and-aft component of a ship framing, located along the centerline of the bottom and connected to the stem or stern frames. Floors or bottom transverses are attached to the keel.

NOTE Flat plate keel is horizontal, centerline, bottom shell strake constituting the lower flange of the keel. Center vertical keel is vertical centerline web of the keel. Bilge keel is a longitudinal fin fitted at the turn of the bilge to reduce rolling. Figure 55 is an example of a bilge keel.

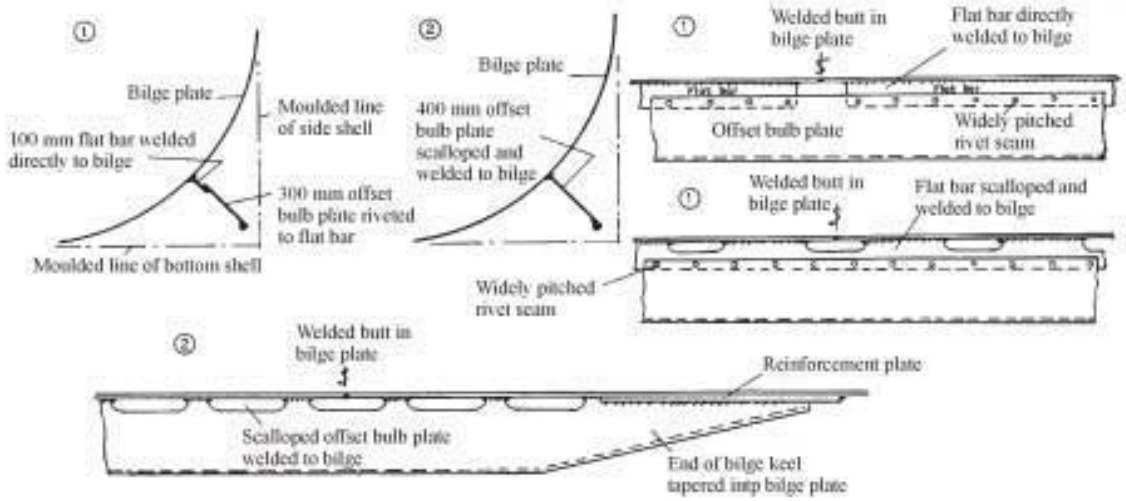


Figure 55 — Bilge keel

4.2.242.2.32 longitudinal_bulkhead

a vertical bulkhead running in fore and aft direction used to partition or subdivide the ship for strength, provide intact stability, and separate spaces.

4.2.242.2.33 longitudinal_girder

a term applied to the fore-and-aft girder in the bottom of a ship. These girders are usually made up from plates and shapes and are sometimes intercostal and sometimes continuous.

NOTE Figure 56 is an example for a longitudinal_girder.

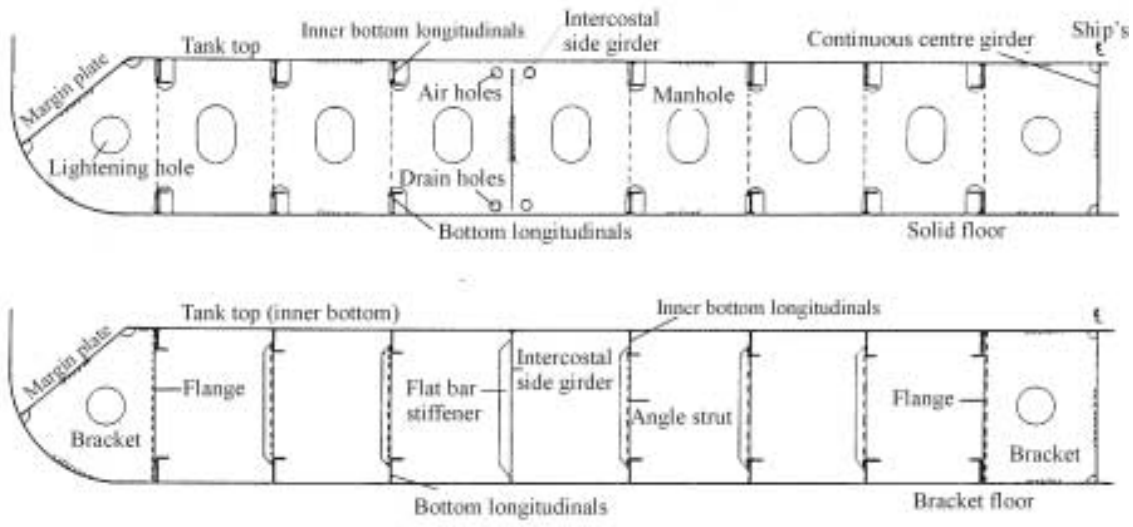


Figure 56 — Longitudinal girder

4.2.242.2.34 lower_boom

the lower boom is located at a lower level in the ship for cargo handling purpose and works in conjunction with upper_boom (see 4.2.242.2.61).

NOTE Boom is a long round spar hinged at its lower end, usually to a mast or king post in a ship, and supported by a wire rope or tackle from aloft to the upper end of the boom. Items such as cargo and stores are lifted by tackle leading from the upper end of the boom.

4.2.242.2.35 machinery_casing

the primary function of the machinery casings is to protect the openings which are fitted in the weather deck over the engines and boilers for access, for light and air, and for the uptakes from the boilers against the sea entering the ship through these openings in heavy weather. Below the weather deck, the machinery casing separates machinery spaces from accommodation and cargo spaces.

4.2.242.2.36 main_deck

the principal deck of the main hull, usually the highest extending from stem to stern and providing strength to the main hull, also called the strength deck.

4.2.242.2.37 mid_ship

a point located equidistant between the forward and after perpendiculars of the ship

4.2.242.2.38 navigation_deck

the navigation deck is typically located in the deckhouse and it is the deck where ship control/navigation is performed.

4.2.242.2.39 outer_shell

in a double hull vessel, the outer shell plating in contact with the sea.

4.2.242.2.40 platform_deck

a lower deck, usually in the cargo space, which does not contribute to the longitudinal strength of the ship.

4.2.242.2.41 plating

a continuous material used to provide support, subdivision, closure, etc. used for bulkheads, decks, deck houses, shell, and inner bottom.

4.2.242.2.42 sheer_strake

the course of side shell plating at strength deck level, usually strengthened and free of openings.

4.2.242.2.43 ship_structure

the structure of the whole ship.

4.2.242.2.44 stool

the stool is typically a structural support located at the base of a corrugated transverse bulkhead providing a transition between bulkhead and inner bottom.

4.2.242.2.45 strength_bulkhead

a bulkhead which contributes to the strength of a vessel.

4.2.242.2.46 strength_deck

the strength deck is designed as the uppermost part of the main hull longitudinal strength girder. The bottom shell plating forms the lower most part of this girder.

4.2.242.2.47 stern_frame

a large casting, forging, or weldment attached to after end of keel. Incorporates the rudder gudgeons and in single-screw ships includes the propeller post.

4.2.242.2.48 stringer

a term applied to a fore-and-aft girder running along the side of a ship at the shell and also to the outboard strake of plating on any deck.

NOTE Also the side pieces of a ladder or staircase to which the treads and risers are fastened.

4.2.242.2.49 superstructure

a decked-over structure above the upper deck, the outboard sides of which are formed by the shell plating as distinguished from a deckhouse that does not extend outboard to the ship sides.

4.2.242.2.50 superstructure_aft_bulkhead

the main transverse bulkhead forming the aft boundary of the superstructure house.

4.2.242.2.51 superstructure_front_bulkhead

the main transverse bulkhead forming the forward boundary of the superstructure house.

4.2.242.2.52 superstructure_side_bulkhead

the main longitudinal bulkheads on both port and starboard sides forming side boundary of the superstructure house.

4.2.242.2.53 tank_bottom

the lower boundary of the tank; bottom shell, top of inner bottom, or deck which forms the bottom boundary of a tank.

NOTE Figure 57 is an example of a tank.

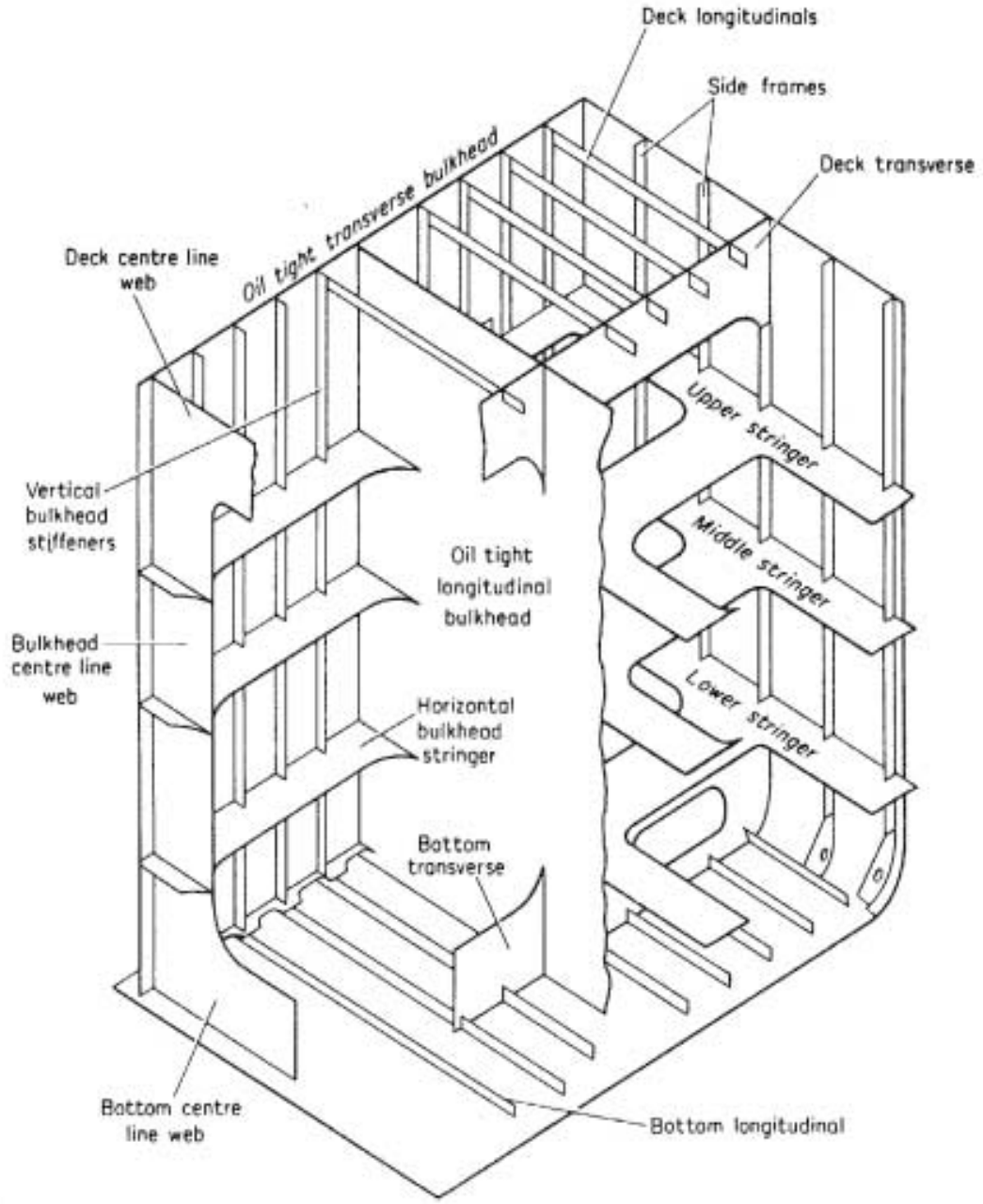


Figure 57 — Tank

4.2.242.2.54 tank_bulkhead

longitudinal or transverse vertical stiffened plate structures which form the side and end boundaries of a tank.

4.2.242.2.55 tank_side

See tank_bulkhead (4.2.242.2.54).

4.2.242.2.56 tank_top

a horizontal stiffened plate structure which forms the upper boundary of the tank.

NOTE See also inner bottom (4.2.242.2.29).

4.2.242.2.57 transom

a stern used to provide additional hull volume and deck space aft and to decrease resistance in some high speed ships.

4.2.242.2.58 transversal_bulkhead

a vertical stiffened plate with stiffeners oriented athwartship and used to partition or subdivide the ship for strength, provide intact stability, and separate spaces.

4.2.242.2.59 transverse_floor

a vertical transverse plate immediately above the bottom shell plating, often located at every frame, extending from bilge to bilge.

4.2.242.2.60 transverse_web_frame

a built-up deep frame to provide extra strength, usually consisting of a web plate flanged or otherwise stiffened on its edge, spaced several frame spaces apart, with smaller, regular frames in between.

NOTE Figure 58 is an example for a transverse_web_frame.

4.2.242.2.61 upper_boom

it is the boom located at a high level in a ship for cargo handling and works in conjunction with lower_boom (see 4.2.242.2.34).

4.2.242.2.62 user_defined

a project specific value to be determined by two or more exchanging organizations.

4.2.242.2.63 vertical_web_frame

a type of transverse web frame (see 4.2.241.2.66).

4.2.242.2.64 wash_bulkhead

longitudinal or transverse non-tight bulkheads fitted in a tank to decrease the swashing action of the liquid contents as a ship rolls and pitches at sea..

NOTE It is also called as swash bulkhead.

4.2.242.2.65 weather_deck

uppermost continuous deck with no overhead protection, and its purpose is to prevent rain water and seawater from entering the ship.

4.2.242.2.66 web_frame

a built-up deep frame to provide extra strength, usually consisting of a web plate flanged or otherwise stiffened on its edge, spaced several frame spaces apart, with the smaller, regular frames in between.

NOTE Figure 58 is an example for a web_frame.

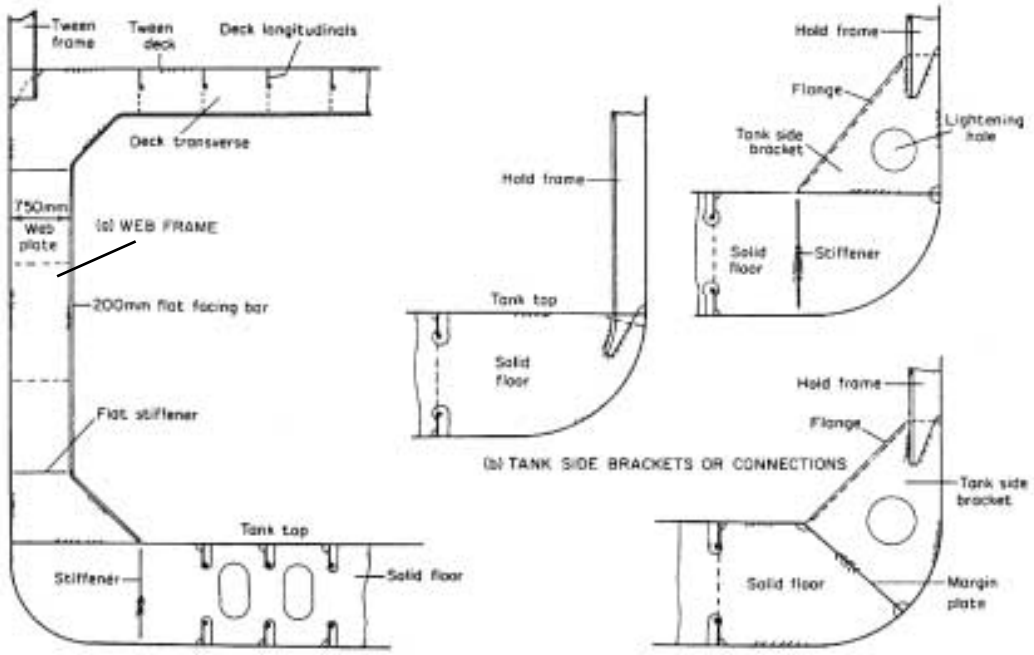


Figure 58 — Web frame

4.2.242.2.67 wing_bulkhead

an outermost longitudinal bulkhead which forms the boundary of wing tank on port side or starboard side.

4.2.243 Structural_system_penetration_relationship

A Structural_system_penetration_relationship is a type of Item_relationship (see 4.2.116) that specifies a Structural_cutout (see 4.2.223) that results from the fact that a Part (see 4.2.152) or System (see 4.2.249) penetrates a Structural_system (see 4.2.239). A Structural_system_-penetration_relationship will normally result in one or several cutouts for the involved elements.

EXAMPLE This relationship may be used to describe that a pipe penetrates a panel or that a profile cuts through the edge of a bulkhead. Also a plate group penetrating another plate group is a valid example.

The data associated with a Structural_system_penetration_relationship are the following:

— item_1;

— penetration_result.

4.2.243.1 item_1

The item_1 specifies the Structural_system that is the penetrated Item of the two Item objects involved in the Structural_system_penetration_relationship.

NOTE Item_1 will become the parent of a cutout feature, as this relationship results in a cutout. See 4.3.185 for the application assertion.

4.2.243.2 penetration_result

The penetration_result specifies the set of Structural_cutout objects that this penetration results in. There may be more than one penetration_result for a Structural_system_penetration_relationship. See 4.3.184 for the application assertion.

NOTE Normally the result would be just one cutout. However, it is possible that both involved parts may get cutouts. The parent attribute in Structural_cutout tells which part gets which cutout. Item_2 is the Item that is considered the penetrating one, if it is possible to distinguish penetrated and penetrating.

4.2.244 Structural_system_relationship

A Structural_system_relationship is a type of System_relationship (see 4.2.251). A Structural_-system_relationship describes the association of a Structural_system with another structural item. Structural item can be one of the following:

— Structural_feature (see 4.2.226),

— Structural_part (see 4.2.229),

— Structural_system (see 4.2.239).

Each Structural_system_relationship is either a Corrugated_structure_relationship (see 4.2.53), a Panel_system_relationship (see 4.2.150), a Structural_system_adjacency_relationship (see 4.2.240), or a Structural_system_symmetry_relationship (see 4.2.245).

The data associated with a Structural_system_relationship are the following:

— item_1.

4.2.244.1 item_1

The item_1 specifies that the Structural_system_relationship relates a Structural_system with something else. See 4.3.186 for the application assertion.

4.2.245 Structural_system_symmetry_relationship

A Structural_system_symmetry_relationship is a type of Structural_system_relationship (see 4.2.244) that specifies that two Structural_system (see 4.2.239) objects are symmetric with respect to a given plane.

NOTE This relationship is only an informal one for those applications that can use this information. No implicit information is intended to be carried, especially this is no short-cut for the definition of one of the related Structural_system objects using the other one.

The data associated with a Structural_system_symmetry_relationship are the following:

— item_2;

— mirroring_plane.

4.2.245.1 item_2

The item_2 specifies the Structural_system that the referencing one is said to be symmetric to. See 4.3.187 for the application assertion.

4.2.245.2 mirroring_plane

The mirroring_plane specifies the plane the symmetry is with respect to.

4.2.246 Structural_weld_shrinkage_allowance_feature

A Structural_weld_shrinkage_allowance_feature is a type of Structural_manufacturing_feature (see 4.2.228) that represents the amount by which the design definition of the part is to be scaled in the directions of the three primary axes of the ship to account for later shrinkage due to welding of the part into an assembly. Application of the Structural_weld_shrinkage_allowance_feature to the design definition results in a manufacturing definition representation whose geometry is scaled up in one or more of the primary ship axes directions.

4.2.247 Structural_weld_shrinkage_allowance_feature_design_definition

A Structural_weld_shrinkage_allowance_feature_design_definition is a type of Feature_design_-definition (see 4.2.86) that provides the values by which the manufactured part is to be scaled to account for later shrinkage due to welding of the part into an assembly. Values are provided for the longitudinal, transverse, and vertical dimensions relative to the ship global coordinate system. A Structural_weld_shrinkage_allowance_feature will scale the edge geometry and marking features of the part in either or both directions, resulting in a new set of edges that represent the extended part's boundary for burning during manufacturing of the part and new locations for marking lines and the centers of holecuts.

EXAMPLE No value means that the part size need not be increased to allowfor weld shrinkage. A value of 0.0003 means that the part and all of its features is scaled by .3 mm per meter of part length in the direction for which an allowance is specified.

The data associated with a Structural_weld_shrinkage_allowance_feature_design_definition are the following:

— defined_for:

— longitudinal_allowance;

— transverse_allowance;

— verical_allowance.

4.2.247.1 defined_for

The defined_for specifies the Structural_weld_shrinkage_allowance_feature that the Structural_-weld_shrinkage_allowance_feature_design_definition is defined for. There may be more than one defined_for for a Structural_weld_shrinkage_allowance_feature_design_definition. See 4.3.188 for the application assertion.

4.2.247.2 longitudinal_allowance

The longitudinal_allowance specifies the amount that the part is scaled in relation to the ship global x direction. The longitudinal_allowance need not be specified for a particular Structural_weld_-shrinkage_allowance_feature_design_definition.

4.2.247.3 transverse_allowance

The transverse_allowance specifies the amount that the part is scaled in relation to the ship global y direction. The transverse_allowance need not be specified for a particular Structural_weld_-shrinkage_allowance_feature_design_definition.

4.2.247.4 vertical_allowance

The vertical_allowance specifies the amount that the part is scaled in relation to the ship global z direction. The vertical_allowance need not be specified for a particular

Structural_weld_shrinkage_allowance_feature_design_definition.

4.2.248 Supplier_bsu

A Supplier_bsu is a type of Bsu (see 4.2.18) that identifies the supplier of a parts library.

The data associated with a Supplier_bsu are the following:

— code.

4.2.248.1 code

The code specifies the designation of the identification of a parts library supplier.

4.2.249 System

A System is a type of both Item (see 4.2.115) and Item_structure (see 4.2.117) that is a functional unit on board a ship. A System is a collection of Part objects and System objects. A System itself is independent of any discipline. Sub-types may, however, be created to represent for example structural systems, piping systems, or propulsion systems. A System shall have a tree structure; that is, the structure shall not be cyclic. This means that a System is not to be referenced by itself directly or indirectly, that is via the relationships that it consists of. Each System is a Structural_system (see 4.2.239).

4.2.250 System_design_definition

A System_design_definition is a type of Design_definition (see 4.2.58) that describes aspects of one or many System (see 4.2.249) objects. Such Definition objects are to collect properties that are specific to the life-cycle phase of design. Each System_design_definition is a Structural_system_-design_definition (see 4.2.241).

The data associated with a System_design_definition are the following:

— defined_for.

4.2.250.1 defined_for

The defined_for specifies the System objects that a System_design_definition is valid for. There may be more than one defined_for for a System_design_definition. See 4.3.189 for the application assertion.

4.2.251 System_relationship

A System_relationship is a type of Item_relationship (see 4.2.116) that describes the association of a System (see 4.2.249) and either another System or a Part (see 4.2.152). Each System_relationship is a Structural_system_relationship (see 4.2.244).

NOTE Subtypes of this entity may for example be used to specify the weld seams necessary to connect an Item to a System.

The data associated with a System_relationship are the following:

— item_1.

4.2.251.1 item_1

The item_1 specifies the System that is related to a Part or another System. See 4.3.190 for the application assertion.

4.2.252 T_bar_cross_section

A T_bar_cross_section is a type of Flanged_profile_cross_section (see 4.2.90) that consists of one flange and one web. The web is centered on the flange.

NOTE 1 ISO 657-21 specifies dimensions for some standard T-bar sections.

NOTE 2 Figure 59 illustrates the parameters of a T_bar_cross_section.

The data associated with a T_bar_cross_section are the following:

— depth;

— flange_thk;

— k;

— radius;

— web_thk;

— width.

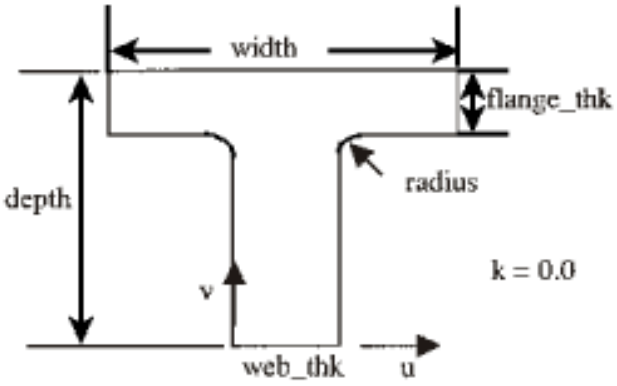


Figure 59 — T bar cross section

4.2.252.1 depth

The depth specifies the depth of the section.

4.2.252.2 flange_thk

The flange_thk specifies the flange thickness.

4.2.252.3 k

The k specifies the distance from outside flange to fillet on web.

4.2.252.4 radius

The radius specifies the radius between the web and the flange.

4.2.252.5 web_thk

The web_thk specifies the web thickness.

4.2.252.6 width

The width specifies the flange width.

4.2.253 Transversal_position

A Transversal_position is a type of Spacing_position (see 4.2.209) that is located on the global Y-axis.

4.2.254 Transversal_table

A Transversal_table is a type of Spacing_table (see 4.2.211) that has positions that are along the transverse axis of the global coordinate system that is the global Y-axis.

The data associated with a Transversal_table are the following:

— spacing_table_representations.

4.2.254.1 spacing_table_representations

The spacing_table_representations specifies the transversal positions that make up the transversal table. The spacing_table_representations need not be specified for a particular Transversal_table. There may be more than one spacing_table_representations for a Transversal_table. See 4.3.191 for the application assertion.

4.2.255 Triangular_cutout_design_definition

A Triangular_cutout_design_definition is a type of Interior_cutout_design_definition (see 4.2.111) that is characterised by the connection of three linear independent points in a plane. The origin is placed at corner point c1, and the X-axis is defined by the straight connection from corner point c1 to c2. The Y-axis is perpendicular to the X-axis and such that corner point c3 has a positive y-coordinate.

The data associated with a Triangular_cutout_design_definition are the following:

— c1_radius;

— c2_radius;

— c2_x;

— c3_radius;

— c3_x;

— c3_y.

4.2.255.1 c1_radius

The c1_radius specifies the radius of the circular arc that rounds corner c1.

4.2.255.2 c2_radius

The c2_radius specifies the radius of the circular arc that rounds corner c2.

4.2.255.3 c2_x

The c2_x specifies the x-coordinate of corner point c2.

4.2.255.4 c3_radius

The c3_radius specifies the radius of the circular arc that rounds corner c3.

4.2.255.5 c3_x

The c3_x specifies the x-coordinate of corner point c3.

4.2.255.6 c3_y

The c3_y specifies the y-coordinate of corner point c3.

4.2.256 Twist_location

A Twist_location is an explicit location and orientation along the trace line of a profile where an angle of twist is to be specified.

The data associated with a Twist_location are the following:

— location;

— orientation.

4.2.256.1 location

The location specifies the location on a Profile_design_definition trace line where an angle of twist is to be specified.

4.2.256.2 orientation

The orientation specifies the angle of the cross-section's local v-axis, tangent to the web of the profile, at the location, for a profile that twists along its length.

4.2.257 Universal_resource_locator

A Universal_resource_locator is the address of an electronic data source. The electronic data source is an internet address.

NOTE This is an alternative to the common mail address as provided by ISO 10303-41. Because there is a permanent development on this field it is impossible to cover all details here, such as, available protocol types. Instead, it is essentials that the structure of an Universal_resource_locator follow precisely the specification of the Uniform Resource Locator as described in RFC 1738 of the W3C Network Working Group.

EXAMPLE Http://www.w3.org.

The data associated with a Universal_resource_locator are the following:

— location.

4.2.257.1 location

The location specifies the storage place where a document or other information is located in the web.

NOTE Such a location is typically composed of several protocol type components such as ftp, or http, or https, machine address such as name or IP number, an optional port number, and a local path. Further components are possible.

EXAMPLE 1 http://www.w3.org/Addressing/rfc1738.txt

EXAMPLE 2 ftp://ftp.atlantec-es.com/pub/out/AP218/post-ws/mapping status.xls

4.2.258 Version_creation

A Version_creation is a type of Versionable_object_change_event (see 4.2.111) that identifies the event leading to a new Definition, Item_structure, or Item_relationship.

The data associated with a Version_creation are the following:

— base;

— subject.

4.2.258.1 base

The base specifies the Versionable_object objects the subject is derived from. The base need not be specified for a particular Version_creation. The base is populated based upon an existing Versionable_object. The base Versionable_object need not be the immediately preceding version of the subject Versionable_object, but may refer to any previous version in the version history of the same Item, or to any Versionable_object of another Item which contributes to the creation of the subject Versionable_object. There may be more than one base for a Version_creation. See 4.3.192 for the application assertion.

4.2.258.2 subject

The subject specifies the Versionable_object created by the Change Event. See 4.3.192 for the application assertion.

4.2.259 Version_deletion

A Version_deletion is a type of Versionable_object_change_event (see 4.2.111) that is the event leading to the deletion of a Definition, an Item_structure, or an Item_relationship.

The data associated with a Version_deletion are the following:

— subject.

4.2.259.1 subject

The subject specifies the Versionable_object deleted or to be deleted by the Versionable_object_change_event. See 4.3.193 for the application assertion.

4.2.260 Version_history

A Version_history is the identification of Versionable_object (see 4.2.263) objects and their Version_relationship (see 4.2.262) objects in terms of their role as predecessors and successors with respect to each other. The Version_history is a directed acyclic graph.

NOTE Consequently the Version_history may contain Versionable_object objects considered alternatives with respect to each other such as a Versionable_object having more than one successor, and merged Versionable_-object objects such as a Versionable_object having more than one predecessor.

The data associated with a Version_history are the following:

— current_version;

— relationships;

— versions.

4.2.260.1 current_version

The current_version specifies the Versionable_object that plays the role of the current version in this Version_history. See 4.3.195 for the application assertion.

4.2.260.2 relationships

The relationships specifies the Version_relationship objects the Version_history consists of. The relationships need not be specified for a particular Version_history. There may be more than one relationship for a Version_history. See 4.3.194 for the application assertion.

4.2.260.3 versions

The versions specifies the Versionable_object objects the Version_history consists of. There may be more than one version for a Version_history. See 4.3.195 for the application assertion.

4.2.261 Version_modification

The Version_modification is a type of Versionable_object_change_event (see 4.2.111) that is the Event leading to a Change of a Versionable_object such as the creation of a new version for an existing thing. The base Versionable_object need not be the immediately preceding version of the subject Versionable_object, but may refer to any previous version in the Version_history of the same Item.

The data associated with a Version_modification are the following:

— base;

— subject.

4.2.261.1 base

The base specifies the Versionable_object objects the subject is derived from. There may be more than one base for a Version_modification. See 4.3.196 for the application assertion.

4.2.261.2 subject

The subject specifies the Versionable_object objects modified or to be modified by the Versionable-_object_change_event. See 4.3.196 for the application assertion.

4.2.262 Version_relationship

A Version_relationship is the relationship of two Versionable_object (see 4.2.262) objects of same type in terms of a Version_history (see 4.2.260).

The data associated with a Version_relationship are the following:

— predecessor;

— reason;

— successor.

4.2.262.1 predecessor

The predecessor specifies the Versionable_object from which the successor is derived. See 4.3.197 for the application assertion.

4.2.262.2 reason

The reason specifies the purpose for a new version, created by a certain person at a certain time.

4.2.262.3 successor

The successor specifies the Versionable_object that the predecessor is the preceding version of. See 4.3.197 for the application assertion.

4.2.263 Versionable_object

A Versionable_object is any object that may be versioned. Each Versonable_object is either a Definition (see 4.2.56), an Item_relationship (see 4.2.116), an Item_structure (see 4.2.117), a Document (see 4.2.65), or a Revision (see 4.2.192)..

The data associated with a Versionable_object are the following:

— description;

— version_id.

4.2.263.1 description

The description specifies additional information that identifies or describes the object.

4.2.263.2 version_id

The version_id specifies the identification of the version.

4.2.264 Versionable_object_change_event

A Versionable_object_change_event is a type of Event (see 4.2.79) that is the generalization of the Event objects effectively changing a Definition, Item_structure, or Item_relationship. Each Versionable_object_change_event is either a Envisaged_version_creation (see 4.2.78), a Version_-creation (see 4.2.258), Version_deletion (see 4.2.259), or Version_modification (see 4.2.261).

4.2.265 Vertical_position

A Vertical_position is a type of Spacing_position (see 4.2.209) that is located on the global Z-axis.

4.2.266 Vertical_table

A Vertical_table is a type of Spacing_table (see 4.2.211) that has positions that are along the vertical axis of the global coordinate system that is the global Z-axis. Each Vertical_table may be a Waterline_table.

The data associated with a Vertical_table are the following:

— spacing_table_representations.

4.2.266.1 spacing_table_representations

The spacing_table_representations specifies the vertical positions that make up the vertical table. The spacing_table_representations need not be specified for a particular Vertical_table. There may be more than one spacing_table_representations for a Vertical_table. See 4.3.198 for the application assertion.

4.2.267 W_shape_cross_section

A W_shape_cross_section is a type of Flanged_profile_cross_section (see 4.2.90) that has non-sloping flanges.

NOTE A W_shape_cross_section is commonly known as a wide flange shape.

The data associated with a W_shape_cross_section are the following:

— depth;

— flange_thk;

— k;

— radius;

— web_thk;

— width.

4.2.267.1 depth

The depth specifies the depth at the outside of the flanges.

4.2.267.2 flange_thk

The flange_thk specifies the flange thickness.

4.2.267.3 k

The k specifies the distance from outside flange to fillet on web.

4.2.267.4 radius

The radius specifies the radius between the web and the flange.

4.2.267.5 web_thk

The web_thk specifies the web thickness.

4.2.267.6 width

The width specifies the width of the flanges.

4.2.268 Waterline_table

A Waterline_table is a type of Vertical_table (see 4.2.266) whose positions are a reference for the location of waterlines and are located on the global Z-axis.

4.2.269 Weight_and_centre_of_gravity

A Weight_and_centre_of_gravity specifies the weight and the centre of gravity of a Ship Part. The moment components will be derived if the moment origin exists. Each Weight_and_centre_of_-gravity may be a Lightship_weight_item (see 4.2.124).

The data associated with a Weight_and_centre_of_gravity are the following:

— centre_of_gravity;

— mass;

— moment.

4.2.269.1 centre_of_gravity

The centre_of_gravity specifies the centre of gravity of a Ship Part. See 4.3.199 for the application assertion.

4.2.269.2 mass

The mass specifies the weight of a ship part.

4.2.269.3 moment

The moment specifies the Moment_3d based on centre_of_gravity and weight of Ship Part. See 4.3.200 for the application assertion.

4.2.270 Weld

A Weld is a type of Structural_part_connection_implementation (see 4.2.230) that represents a connection between parts that is made by heating the part edges to a temperature sufficient to melt the metal while simultaneously depositing additional weld filler material. As the material cools the parts are permanently bonded together.

NOTE Figure 60 shows weld examples.

The data associated with a Weld are the following:

— realization_of.

4.2.270.1 realization_of

The realization_of specifies the weld which is applicable to a Welded_joint (see 4.2.275). See 4.3.201 for the application assertion.

4.2.271 Weld_design_definition

A Weld_design_definition is a type of Design_definition (see 4.2.58) that specifies the weld definition from the design lifecycle stage view. It represents characteristics and attributes directly related to the weld design of the parts. At this stage the welding features are identified that affect aspects of the parts that have to do with processes that will be used to cut and form the parts for the purpose of welding preparations.

The data associated with a Weld_design_definition are the following:

— connection_angle;

— defined_for;

— end_suspension;

— inclination_angle;

— rotation_angle;

— shape_of_weld_surface;

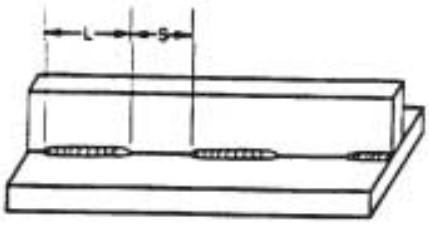
— shrinkage;

— start_suspension;

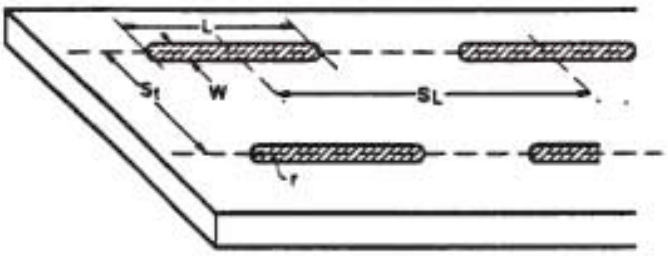
— weld_cross_section_area;

— weld_geometry;

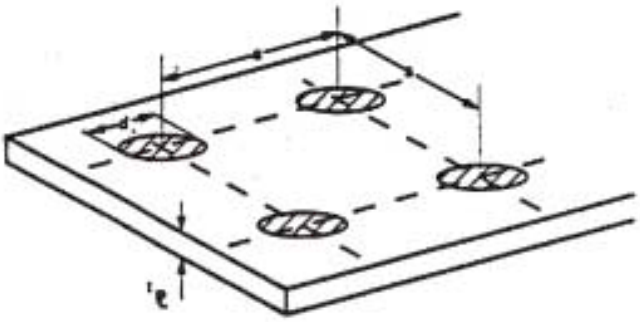
— weld_size.



EXAMPLE OF INTERMITTENT WELDING



SLOT WELD ATTRIBUTES



PLUG WELD ATTRIBUTES

Figure 60 — Weld examples

4.2.271.1 connection_angle

The connection_angle specifies the angle between two welded parts at the connecting point.

4.2.271.2 defined_for

The defined_for specifies the Weld (See 4.2.270) for which the Weld_design_definition is applicable. There may be more than one defined_for for a Weld_design_definition. See 4.3.202 for the application assertion.

4.2.271.3 end_suspension

The end_suspension specifies the suspension of welding at the end point of the Weld. The end_-supension need not be specified for a particular Weld_design_definition.

4.2.271.4 inclination_angle

The inclination_angle specifies the angle between the horizontal plane and the root line of welding. The value of the inclination_angle is variable from 0 to 90 degrees.

4.2.271.5 rotation_angle

The rotation_angle specifies the angle between the vertical plane including the root axis, and the plane including both the root axis and the line equally distant from the groove faces, or the line dividing equally the angle between two plates in case of fillet welding. The value of the rotation_-angle is 180 degree or under.

4.2.271.6 shape_of_weld_surface

The shape_of_weld_surface specifies an indicator as to the final shape of the top surface of a weld.

The value of the shape_of_weld_surface is one of the following:

— concave;

— convex;

— flush;

— none.

NOTE See 4.2.271.6.1 - 4.2.271.6.4 for the definition of each allowable value for shape_of_weld_surface.

4.2.271.6.1 concave

forming a concave top surface at a weld.

4.2.271.6.2 convex

forming a convex top surface at a weld.

4.2.271.6.3 flush

forming a continuous plane top surface at a weld.

4.2.271.6.4 none

forming no specific top surface at a weld.

4.2.271.7 shrinkage

The shrinkage specifies the length of the proposed shrinkage of the parts at the Weld.

4.2.271.8 start_suspension

The start_suspension specifies the suspension of welding at the start point of the Weld. The start_-suspension need not be specified for a particular Weld_design_definition.

4.2.271.9 weld_cross_section_area

The weld_cross_section_area specifies the area measure of the cross-section of the weld detail to be filled with weld filler material. This area can be used to estimate the amount of weld material required. The weld_cross_section_area need not to be specified for a particular Weld_design_-definition.

4.2.271.10 weld_geometry

The weld_geometry specifies the geometry of the connected contour of welded parts.

4.2.271.11 weld_size

The weld_size specifies the dimensions used in design to designate the result size of a weld. Cases of equal legs and unequal legs, concave and convex surface forms are considered. The weld_size need not be specified for a particular Weld_design_definition.

EXAMPLE Leg length of fillet weld, and width as well as reinforcement height of butt weld.

4.2.272 Weld_filler_material

A Weld_filler_material is a type of Ship_material_property (see 4.2.206) that represents the chemical composition and make-up of metal being used to fill the joint opening. The final strength of welded joint depends on the type of weld fillet material used.

The data associated with a Weld_filler_material are the following:

— chemical_composition;

— chemical_composition_id;

— defined_for;

— electrode_id;

— material_strength_id;

— notch_impact_work_id.

4.2.272.1 chemical_composition

The chemical_composition specifies the chemical composition of the wire electrode. It includes the percentage value of the identified alloy components. The chemical_composition need not be specified for a particular Weld_filler_material. See 4.3.203 for the application assertion.

4.2.272.2 chemical_composition_id

The chemical_composition_id specifies the identifier of the chemical composition of the wire electrode. The chemical_composition_id need not be specified for a particular Weld_filler_material.

4.2.272.3 defined_for

The defined_for specifies the Weld (See 4.2.270) for which the Weld_filler_material is applicable. There may be more than one defined_for for a Weld_filler_material. See 4.3.204 for the application assertion.

4.2.272.4 electrode_id

The electrode_id specifies the identifier for the used electrode wire which provides filler material to the welded joint. The electrode_id need not be specified for a particular Weld_filler_material.

EXAMPLE G0, G3Si1.

4.2.272.5 material_strength_id

The material_strength_id specifies the strength identifier for the weld filler material. The material_-strength_id specifies the minimum extensibility, the tensile strength and the minimum fracture-stretch of the weld filler material. The material_strength_id need not be specified for a particular Weld_-filler_material.

EXAMPLE A filler material has a material_strength_id "35"; it means that the material has minimum extensibility 355 N/mm2, tensile strength 440 to 570 N/mm2 and minimum fracture-stretch 22%.

4.2.272.6 notch_impact_work_id

The notch_impact_work_id specifies the identifier for the notch impact work of the filler material. It is defined by a temperature at which the notch impact work reaches a value of 47J. The notch_-impact_work_id need not be specified for a particular Weld_filler_material.

EXAMPLE A filler material has a notch_impact_work_id "A", it means that the temperature for the minimum notch_impact_work 47J is +20 degree C.

NOTE 47J = 47 Joules = 34.665 foot pounds

4.2.273 Weld_manufacturing_definition

A Weld_manufacturing_definition is a type of Manufacturing_definition (see 4.2.131) that specifies the weld definition from the production lifecycle stage view. It represents characteristics and attributes directly related to the weld manufacturing of the parts. At this stage the welding methods, environment, and procedures are specified that affect aspects of the weld manufacturing processes and the qualification of a weld.

The data associated with a Weld_manufacturing_definition are the following:

— defined_for;

— degree_of_automatizations;

— environment;

— number_of_weld_passes;

— position;

— process;

— sequences;

— tack_weld_used;

— torch_vector;

— weld_test;

— welding_deposition_sequences;

— welding_procedures.

4.2.273.1 defined_for

The defined_for specifies the Weld_manufacturing_definition points to Weld only. See 4.3.206 for the application assertion.

4.2.273.2 degree_of_automatizations

The degree_of_automatizations specifies the automatization status of the weld manufacturing process.

The value of the degree_of_automatizations is one of the following:

— automatic;

— manual;

— mechanized;

— robotic;

— self_run;

— user defined.

NOTE See 4.2.273.2.1 - 4.2.273.2.6 for the definition of each allowable value for degree_of_automatizations.

4.2.273.2.1 automatic

welding performed by using equipment so designed that welding progresses continuously without constant operating by a worker.

4.2.273.2.2 manual

welding in which welding operation is performed by hand.

NOTE Semi-automatic welding performed by using equipment that automatically feeds wire while operating the torch manually, may be identified as manual.

4.2.273.2.3 mechanized

welding performed by the motorization and automatization of operations.

EXAMPLE Part_supply, clamp, feed, rotation, reverse_rotation, clamp_release, and discharge are types of operations that can be motorized or automated.

4.2.273.2.4 robotic

a type of automatic welding performed by using an industrial robot.

EXAMPLE Robotic arc welding and robotic spot welding.

4.2.273.2.5 self_run

welding in which when a covered electrode is set to a simple equipment, the welding thereafter automatically progresses only by ignition of arc.

4.2.273.2.6 user defined

user defined welding automatization degree.

4.2.273.3 environment

The environment specifies the identifier used to denote whether the weld has been designated to be made in a controlled environment or in a non-controlled environment. Welds made in the field are usually more difficult to make and less efficiently produced.

The value of the environment is one of the following:

— field;

— shop.

NOTE See 4.2.273.3.1 - 4.2.273.3.2 for the definition of each allowable value for environment.

4.2.273.3.1 field

the non-controlled environment in the open.

4.2.273.3.2 shop

the controlled environment in a shop.

4.2.273.4 number_of_weld_passes

The number_of_weld_passes specifies a count measuring the number of incremental passes the person or machine making the weld needs to perform in order to achieve the final weld thickness.

NOTE Depending on the weld process used, only a fixed amount of weld filler material may be laid down by an electrode in a single layer. When welds thicker than this amount are required, multiple passes must be used to build up to the final thickness and to suit the required cross-section of weld.

EXAMPLE If a 12mm fillet weld is required and a 6mm rod is used, then 2 passes would be necessary.

4.2.273.5 position

The position specifies the orientation of the weld with respect of the person performing the welding. The position affects the rate at which weld material can be deposited and factors into weld estimating calculation. All welding positions are indicated by the angle of rotation and the angle of inclination.

NOTE Figure 61 illustrates weld positions.

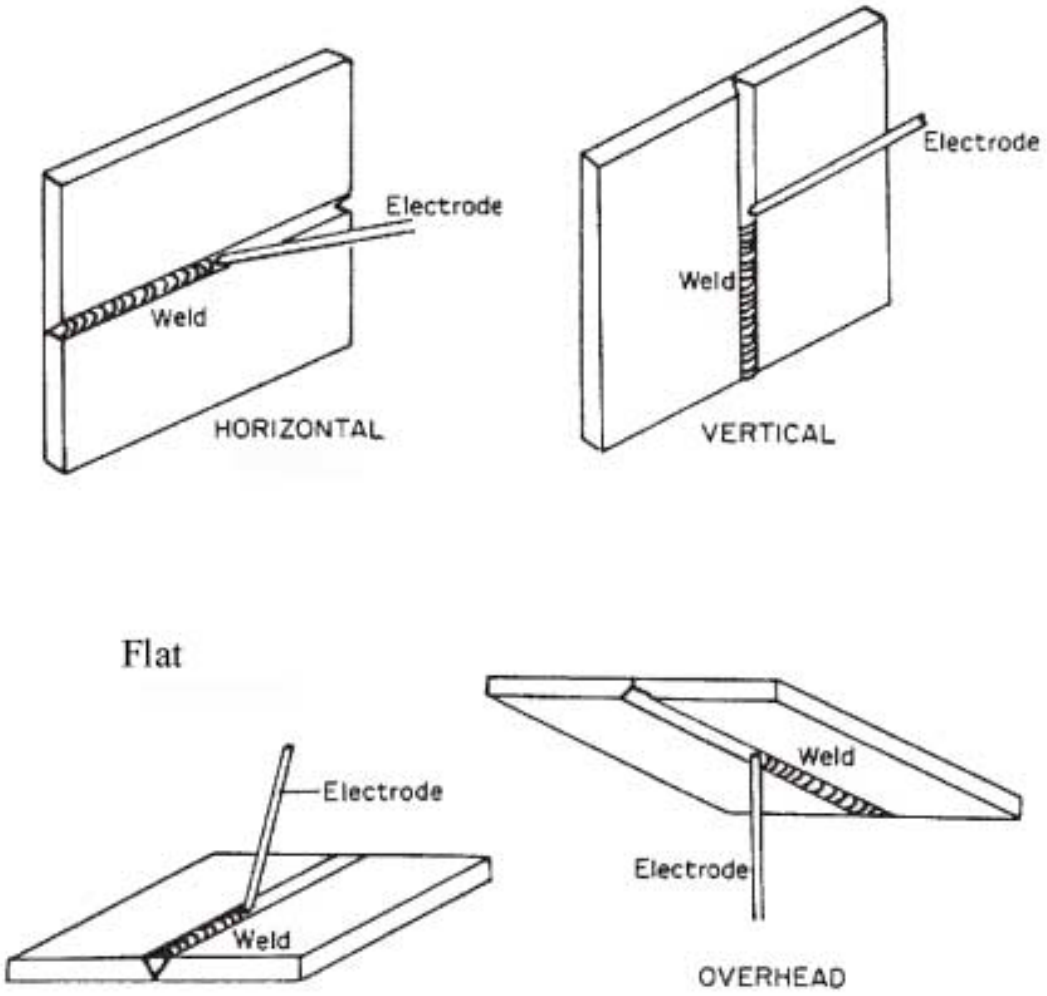


Figure 61 — Weld positions

The value of the position is one of the following:

— flat;

— horizontal;

— other;

— overhead;

— vertical.

NOTE See 4.2.273.5.1 - 4.2.273.5.5 for the definition of each allowable value for position.

4.2.273.5.1 flat

the welding position to weld while facing downward from the top against a joint whose welding axis is almost horizontal.

NOTE A flat welding position is used mostly in welding with manual electrodes.

EXAMPLE Welding at the deck from above.

4.2.273.5.2 horizontal

the welding position to put bead horizontally against a joint whose welding axis is almost horizontal while the parts being welded are in a vertical position.

EXAMPLE Welding across a bulkhead.

4.2.273.5.3 other

other possible position.

4.2.273.5.4 overhead

the welding position to weld while facing upward from the bottom against a joint whose welding axis is almost horizontal, the parts being welded are over the head of the welder.

EXAMPLE Welding the deck from below.

4.2.273.5.5 vertical

the welding position to put bead vertically from the top or bottom against a joint whose welding axis is almost vertical, both parts being welded are also in vertical position, for example up a bulkhead

EXAMPLE Welding vertically up a bulkhead.

4.2.273.6 process

The process specifies the process to be used to make the weld.

NOTE The process currently used in shipbuilding is the electric arc welding that includes slag shielded process such as smaw, saw and fcaw, as well as the inert gas shielded process such as gmaw.

The value of the process is one of the following:

— fcaw;

— gmaw;

— saw;

— smaw;

— user_defined.

NOTE See 4.2.273.6.1 - 4.2.273.6.5 for the definition of each allowable value for process.

4.2.273.6.1 fcaw

flux-cored arc welding.

4.2.273.6.2 gmaw

gas metal arc welding.

4.2.273.6.3 saw

submerged arc welding.

4.2.273.6.4 smaw

manually shielded metal arc welding.

4.2.273.6.5 user_defined

other user defined non-standard arc welding process.

4.2.273.7 sequences

The sequences specifies the welding sequences developed to specify the sequences involving welding of ship structures. There may be more than one defined_for for a Weld_manufacturing_definition. See 4.3.208 for the application assertion.

NOTE To reduce distortion and limit the residual stress in the structure it is important that a correct welding sequence should be utilized throughout the construction. This applies both during the fabrication of units and at erection and joining of units on the building berth.

4.2.273.8 tack_weld_used

The tack_weld_used specifies the indicator as to explain whether or not a tack weld is used. The tack weld holds plates and sections in place after alignment and prior to completion of the continuous fillet and butt weld.

4.2.273.9 torch_vector

The torch_vector specifies the vector of the welding torch, that indicates the direction of the torch that the welder holds.

4.2.273.10 weld_test

The weld_test specifies the test to be done to qualify the weld. The test methods, found weld faults, and test results should be given to judge the weld qualification. There may be more than one weld_-test for a Weld_manufacturing_definition See 4.3.207 for the application assertion.

4.2.273.11 welding_deposition_sequences

The welding_deposition_sequences specifies the methods of welding deposition used to minimize distortion of welding.

NOTE 1 Figure 62 illustrates the weld distortion.

NOTE 2 Figure 63 illustrates different weld methods: backstep and wandering methods of welding.

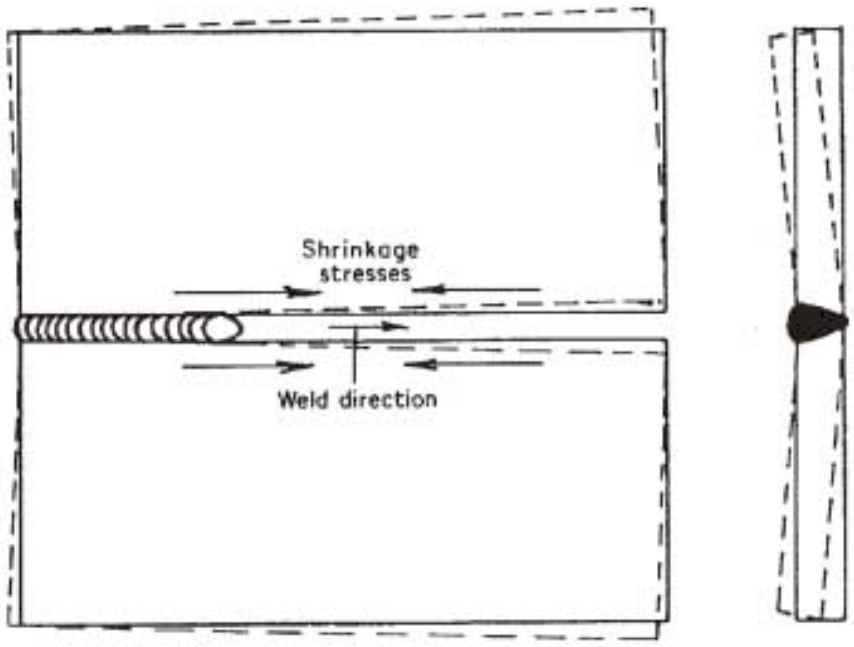


Figure 62 — Weld distortion

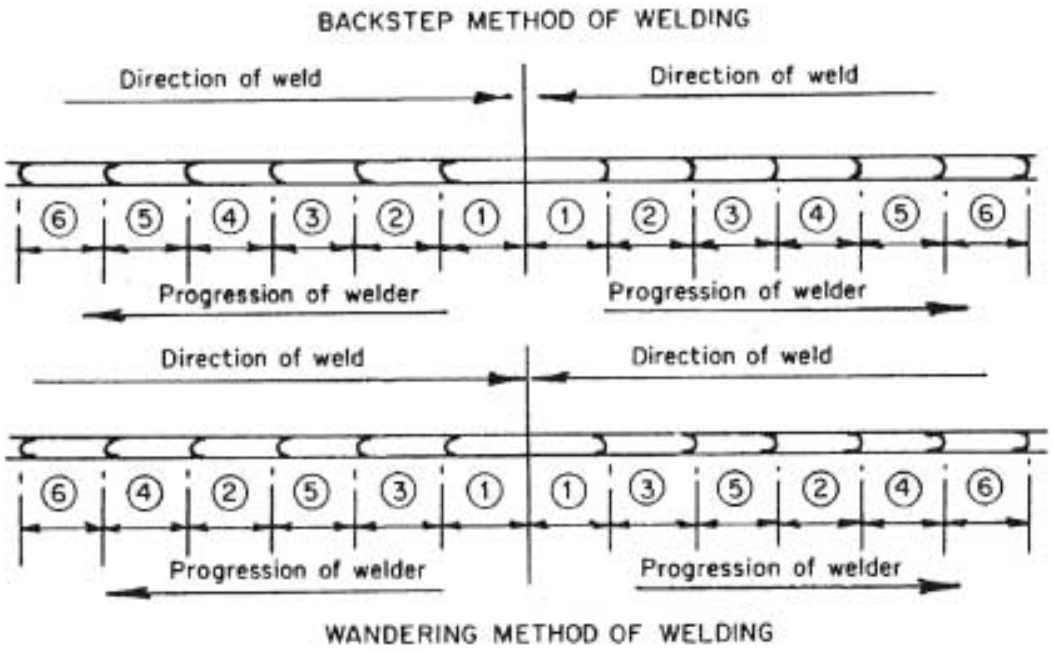


Figure 63 — Weld methods

The value of the welding_deposition_sequences is one of the following:

— backstep;

— progressive;

— user_defined;

— wandering.

NOTE See 4.2.273.11.1 - 4.2.273.11.4 for the definition of each allowable value for welding_deposition_-sequences.

4.2.273.11.1 backstep

a method in which deposition is carried out so that a welding direction of welder becomes reverse to a deposition direction.

4.2.273.11.2 progressive

a method in which deposition is carried out so that a welding direction of welder has the same direction as a deposition direction.

4.2.273.11.3 user_defined

a deposition sequence defined by the user.

4.2.273.11.4 wandering

a method in which both progressive and backstep methods are combined and used.

4.2.273.12 welding_procedures

The welding_procedures specifies the set of text instructions developed to ensure that the proper care, preparation and actions are taken or followed during production welding of the joint. There may be more than one welding_procedures for a Weld_manufacturing_definition. See 4.3.205 for the application assertion.

EXAMPLE Welding_procedures address such things as pre-head, post-head, edge preparation, cooling time between successive weld passes.

NOTE Figure 64 illustrates weld procedures.

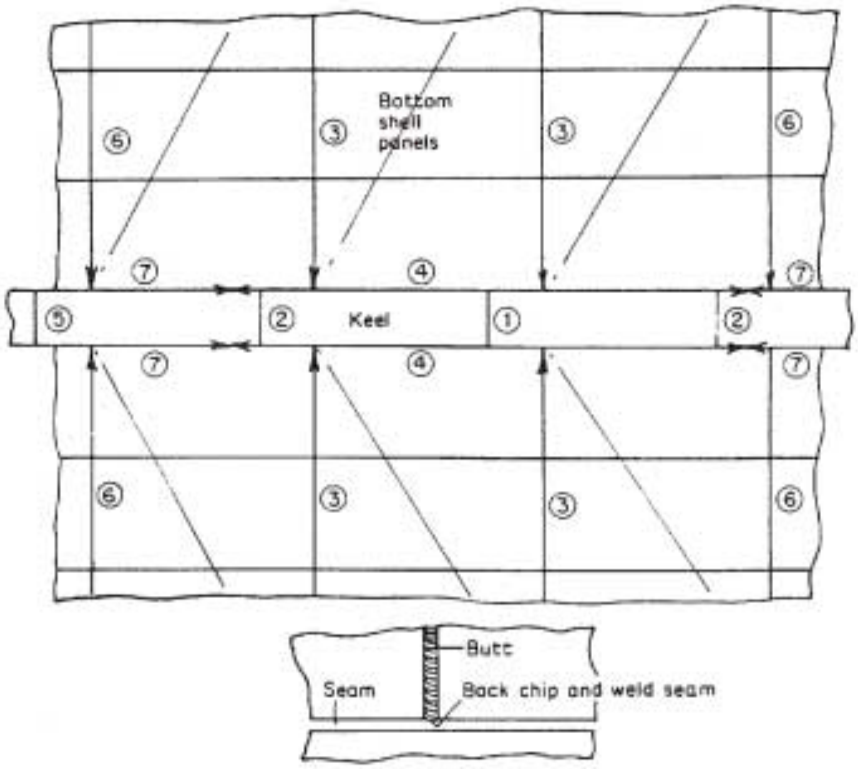


Figure 64 — Weld procedures

4.2.274 Weld_testing

A Weld_testing contains information that outlines the test methods and test steps that must be followed to ensure that the weld was properly made. Test pieces is evaluated according to the acceptance requirements specified for relevant types of imperfections.

The data associated with a Weld_testing are the following:

— test_methods;

— test_results;

— test_sequence.

4.2.274.1 test_methods

The test_methods specifies the methods for welding tests. The test_methods are required to enable the soundness of a ship weld to be assessed. Test_methods may have one or many values contained in it.

The value for the test_methods shall be one of the following:

— dye_penetrant;

— magnetic_particle;

— radiographic;

— ultrasonic;

— visual_examination.

NOTE See 4.2.274.1.1 - 4.2.274.1.5 for the definition of each allowable value for test_methods.

4.2.274.1.1 dye_penetrant

a method to show up a surface flaw if it remains after the casting part has been washed following the application of the dye.

NOTE The dye penetrant test has a small application in ship hull construction, being used for examining for surface cracks in stern frames and other castings. To aid the detection of a surface crack the dye penetrant used is often luminous and is revealed under an ultraviolet light.

4.2.274.1.2 magnetic_particle

a method carried out by magnetizing the casting parts, and spreading a fluid of magnetic particles on the surface such as iron filings suspended in paraffin.

NOTE Any discontinuity such as a surface crack will show up as the particles will concentrate at this point where there is an alteration in the magnetic field. The magnetic particle test has a small application in ship hull construction, being used for examining for surface cracks in stern frames and other castings.

4.2.274.1.3 radiographic

an inspection method the principle of which is simply to subject a material to radiation from one side, and record the radiation emitted from the opposite side.

NOTE Any obstacle in the path of the radiation will affect the radiation density emitted and may be recorded. As radiation will expose photographic plate, for all practice weld test purpose this is used to record the consistency of the weld metal. Either X-ray or gamma ray devices may be used to provide the source of radiation.

4.2.274.1.4 ultrasonic

a method used increasingly as a tool for locating defects in welds. The technique is particularly useful for locating fine cracks.

NOTE The principle of ultrasonic depends on the fact that pulses of ultrasonic energy are reflected from any surface which they encounter. Ultrasonic waves traveling through a plate may be reflected from the surface of the metal and also from the surface of any flaws which exist in the metal.

4.2.274.1.5 visual_examination

a routine procedure executed by sight.

NOTE Surface defects are soon noticed by the experienced inspector. Incorrect bead shape, high spatter, undercutting, bad stop and start points, incorrect alignment, and surface cracks are all faults which may be observed at the surface. This examination is commonly used in ship hull construction, is made before, during and after welding.

4.2.274.2 test_results

The test_results specifies the results of a weld test. The results assess a weld qualification status in accordance with approved standards. A weld is approved if the imperfections in the test piece are within the specified limits of those standards. Reference should be also made to the corresponding acceptance criteria for non-destructive examination.

The value of the test_results is one of the following:

— other;

— pending;

— pre_qualified;

— rejected.

NOTE See 4.2.274.2.1 - 4.2.274.2.4 for the definition of each allowable value for test_results.

4.2.274.2.1 other

other possible status.

4.2.274.2.2 pending

the weld status which is not finally judged. In this case an additional test is required in order to assess the quality and integrity of the weld.

4.2.274.2.3 pre_qualified

an approved status which indicates that the imperfections in the test piece are within the specified limits.

4.2.274.2.4 rejected

a not approved status which indicates that the imperfections in the test piece exceed the permitted maximum specified.

4.2.274.3 test_sequence

The test_sequence specifies the set of text instructions to give the necessary test sequence. The test_-sequence need not be specified for a particular Weld_testing. There may be more than one test_-sequence for a Weld_testing. See 4.3.209 for the application assertion.

EXAMPLE Which test should be done first, which method is necessary for a weld in hull structure.

4.2.275 Welded_joint

A Welded_joint is a type of Structural_part_joint (see 4.2.233) that represents a connection between parts that is made by heating the part edges in the vicinity of the joint to a temperature sufficient to melt the metal while simultaneously depositing additional weld filler material to the joint opening. As the materials cool, the parts are permanently bonded together.

4.2.276 Welded_joint_design_definition

A Welded_joint_design_definition is a type of Structural_part_joint_design_definition (see 4.2.234) that specifies the the design aspects of a connection. A Welded_joint_design_definition may be one of the following: a Fillet_weld (see 4.2.88), a Groove_weld (see 4.2.101), or a Spot_seam_weld (see 4.2.212).

The data associated with a Welded_joint_design_definition are the following:

— configuration;

— defined_for;

— penetration;

— representations.

4.2.276.1 configuration

The configuration specifies the actual arrangement of the structural parts being joined at the structural
part joint. The required connection configuration should agree with this physical implementation.

NOTE Figure 65 illustrates different weld types.

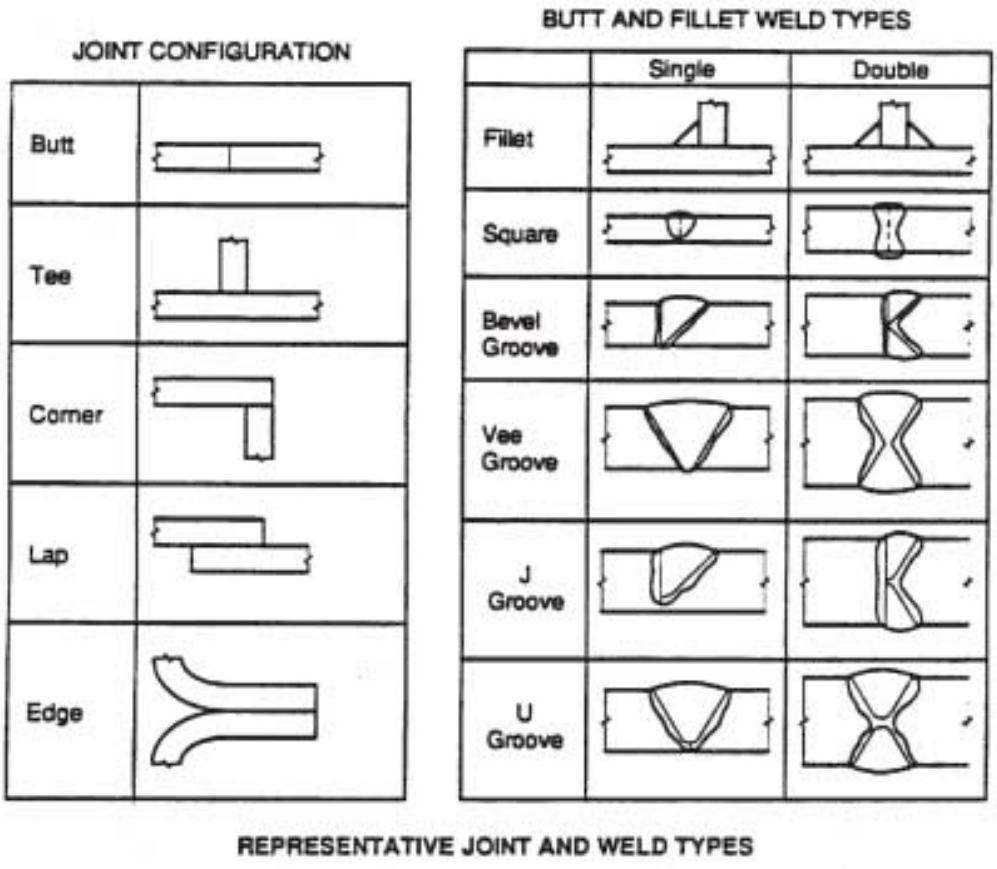


Figure 65 — Weld types

The value of the configuration is one of the following:

— butt;

— corner;

— cruciform;

— edge_joint;

— flare_groove;

— lap;

— stud;

— surfacing;

— tee;

— user_defined.

NOTE See 4.2.276.1.1 - 4.2.276.1.10 for the definition of each allowable value for configuration.

4.2.276.1.1 butt

weld joints in which the base metals lie nearly in the same plane.

4.2.276.1.2 corner

weld joints at a corner where two base metals are kept in an L shape making approximately a right angle.

4.2.276.1.3 cruciform

weld joints forming a cross.

4.2.276.1.4 edge_joint

weld joints at a common end of two base metals with or without partially equal surfaces overlapping.

EXAMPLE Single flange and double flange welding.

4.2.276.1.5 flare_groove

weld joints whose groove form is constructed from arc to arc or from arc to line.

4.2.276.1.6 lap

weld joints formed by partially overlapping base metal.

EXAMPLE Fillet, spot, seam weldings, brazing and soldering.

4.2.276.1.7 stud

weld joints whose welding is performed by generating arc between an attached metallic part such as a bolt or bar and the base metallic part, and pressing the attached part into the resulting molten pool.

4.2.276.1.8 surfacing

weld joints in which metal is deposited on a base metal surface.

EXAMPLE Cladding by welding, thermal spraying.

4.2.276.1.9 tee

weld joints making approximately a right angle in a T shape formed by putting the end of one plate on the surface of another plate.

4.2.276.1.10 user defined

other user defined possible weld joint configurations.

4.2.276.2 defined_for

The defined_for specifies the Welded_joint (see 4.2.275) for which the Welded_joint_design_definition is applicable. There may be more than one defined_for for a Welded_joint_design_definition. See 4.3.214 for the application assertion.

4.2.276.3 penetration

The penetration specifies the identifier used to determine the result of weld whether the weld metal in "full" or "partial" penetration with the base metal of the parts being joined.

NOTE A full penetration weld is stronger than a partial penetration weld and there are situations and areas of a ship where full penetration welds are required. The most significant factor affecting whether a weld detail is full or partial are weld parameters such as root opening, bevel angles and backing.

The value of the penetration is one of the following:

— full;

— partial.

NOTE See 4.2.276.3.1 - 4.2.276.3.2 for the definition of each allowable value for penetration.

4.2.276.3.1 full

the weld metal penetrates fully through the base metal of the parts being joined.

4.2.276.3.2 partial

the weld metal penetrates only partially through the base metal of the parts being joined.

4.2.276.4 representations

The representations specifies the geometric shape of the Welded_joint_design_definition. The representations need not be specified for a particular Welded_joint_design_definition. There may be more than one representations for a Welded_joint_design_definition. See 4.3.210, 4.3.211, 4.3.212, 4.3.213 for the application assertions. The valid shape representations are the following:

— Advanced_brep_shape (see 4.2.1);

— Edge_based_wireframe_shape (see 4.2.70);

— Geometrically_bounded_wireframe_shape (see 4.2.98);

— Non_manifold_surface_shape (see 4.2.140).

Representations may have one or many values contained in it, without any duplications.

4.2.277 Welding_sequence

A Welding_sequence is the production sequence of the Weld (see 4.2.270).

NOTE Figure 66 illustrates weld sequence information.

The data associated with a Welding_sequence are the following:

— direction;

— end_offset;

— name;

— start_offset.

4.2.277.1 direction

The direction specifies the flag indicating the welding direction. This is a Boolean identifier that indicates TRUE when going from the start point to the end point of the weld, or False when going from the end point to the start point of the weld.

4.2.277.2 end_offset

The end_offset specifies the offset of the weld location from the end point of a Welding_sequence.

4.2.277.3 name

The name specifies the name of Welding_sequence.

4.2.277.4 start_offset

The start_offset specifies the offset of the weld location from the start point of a Welding_sequence.

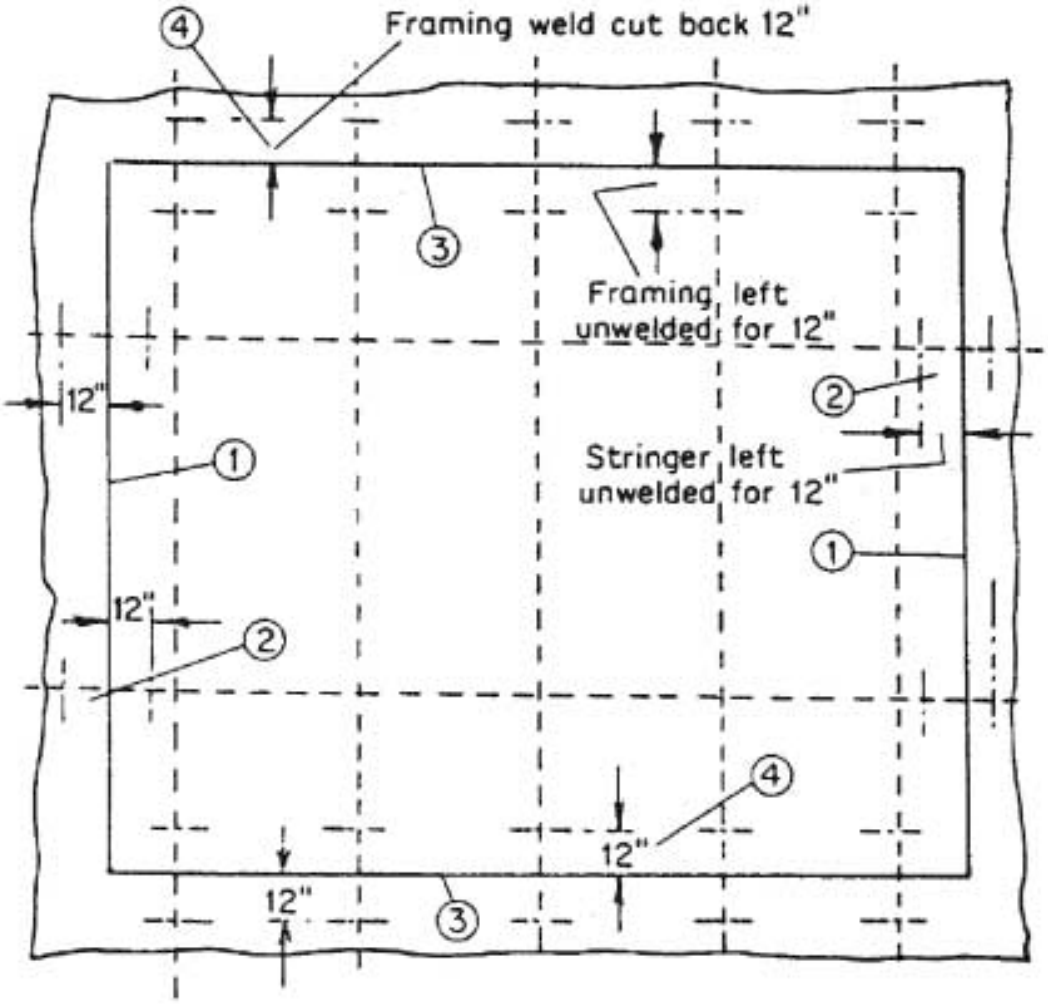


Figure 66 — Weld sequence

4.2.278 Working_ship

A Working_ship is a type of Shiptype (see 4.2.207) that is a Ship that is constructed to perform specific tasks.

The data associated with a Working_ship are the following:

— has_type.

4.2.278.1 has_type

The has_type specifies the type of a Working_ship.

The value of the has_type is one of the following:

— crane_vessel;

— dredger;

— drilling_vessel;

— fire_fighter;

— fishing_vessel;

— floating_dock;

— floating_hotel;

— fpgo;

— fpso;

— ice_breaker;

— offshore_supply_vessel;

— oil_production_and_storage_vessel;

— oil_production_vessel;

— oil_storage_vessel;

— pilot_boat;

— pipe_laying_vessel;

— pusher;

— reefer;

— sealer;

— shuttle_tanker;

— stern_trawler;

— supply_vessel;

— tug;

— user_defined.

— well_smulation_vessel;

NOTE See 4.2.278.1.1 - 4.2.278.1.25 for the definition of each allowable value for has_type.

4.2.278.1.1 crane_vessel

a Working_ship constructed for lifting purposes.

4.2.278.1.2 dredger

a Working_ship constructed for dredging channels or harbour entrances.

4.2.278.1.3 drilling_vessel

a Working_ship constructed for drilling purposes.

4.2.278.1.4 fire_fighter

a Working_ship constructed for fire fighting purposes.

4.2.278.1.5 fishing_vessel

a Working_ship constructed for fishing.

4.2.278.1.6 floating_dock

a Working_ship constructed for lifting ships for repair purposes.

4.2.278.1.7 floating_hotel

a Working_ship constructed for hotel purpose.

4.2.278.1.8 fpgo

a floating platform used for production and storage of gas.

4.2.278.1.9 fpso

a floating platform used for production and storage of oil.

4.2.278.1.10 ice_breaker

A Working_ship constructed for breaking ice.

4.2.278.1.11 offshore_supply_vessel

a Working_ship constructed for supplying offshore platforms.

4.2.278.1.12 oil_production_and_storage_vessel

a Working_ship used for production and storage of oil.

4.2.278.1.13 oil_production_vessel

a Working_ship constructed for production and storage of oil.

4.2.278.1.14 oil_storage_vessel

a Working_ship used for storage of oil.

4.2.278.1.15 pilot_boat

a Working_ship constructed for carrying the pilot to the Ship that the pilot conducts in and out of the harbour.

4.2.278.1.16 pipe_laying_vessel

a Working_ship constructed for pipe laying purposes.

4.2.278.1.17 pusher

a Working_ship constructed to push other unpropelled ships.

4.2.278.1.18 reefer

a Working_ship constructed for the transport of any refrigerated cargo.

4.2.278.1.19 sealer

a Working_ship constructed for the purpose of seal hunting.

4.2.278.1.20 shuttle_tanker

a ship equipped for offshore oil loading. It may be used for transporting oil from a platform to shore.

4.2.278.1.21 stern_trawler

a Working_ship constructed for trawling the sea from the stern of the Ship.

4.2.278.1.22 supply_vessel

a Working_ship constructed for supplying purposes.

EXAMPLE Offshore supply vessel.

4.2.278.1.23 tug

a Working_ship constructed for towing purposes.

4.2.278.1.24 user_defined

this can be any Shiptype that is not in the enumeration list. Details should be found in the description attribute for Shiptype.

4.2.278.1.25 well_stimulation_vessel

a Working_ship constructed for the purposes of stimulating a well.

4.3 Application assertions

This subclause species the application assertions for the Ship Structures application protocol. Application assertions specify the relationships between application objects, the cardinality of the relationships, and the rules required for the integrity and validity of the application objects and UoFs. The application assertions and their definitions are given below.

NOTE There are several assertions below which specify a cardinality of zero for the relationship between a subtype of Definition and Derived_unit (or Named_unit). This occurs because the local_units attribute of Definition is redeclared in the subtype to be an empty set.

4.3.1 Alternative_version_relationship to Versionable_object

Each Alternative_version_relationship has alternative_1 defined by exactly one Versionable_object objects. Each Versionable_object defines alternative_1 for zero, one, or many Alternative_version_-relationship objects.

Each Alternative_version_relationship has alternative_2 defined by exactly one Versionable_object objects. Each Versionable_object defines alternative_2 for zero, one, or many Alternative_version_-relationship objects.

4.3.2 Approval_history to Approval_event

Each Approval_history has approvals defined by one or many Approval_event objects. Each Approval_event defines approval_reference for exactly one Approval_history.

4.3.3 Approval_history to Definition

Each Approval_history has subject defined by exactly one Definition. Each Definition defines the subject for zero, one, or many Approval_history objects.

4.3.4 Assembly_bounding_box to Centre_location

Each Assembly_bounding_box has point_max defined by exactly one Centre_location object. Each Centre_location defines point_max for zero, one, or many Assembly_bounding_box objects.

Each Assembly_bounding_box has point_min defined by exactly one Centre_location object. Each Centre_location defines point_min for zero, one, or many Assembly_bounding_box objects.

4.3.5 Assembly_manufacturing_definition to Assembly

Each Assembly_manufacturing_definition has defined_for defined by one or many Assembly objects. Each Assembly defines defined_for for zero, one, or many Assembly_manufacturing_definition objects.

4.3.6 Assembly_manufacturing_definition to Assembly_manufacturing_-position

Each Assembly_manufacturing_definition has assembly_positions defined by zero, one, or many Assembly_manufacturing_position objects. Each Assembly_manufacturing_position defines assembly_positions for zero, one, or many Assembly_manufacturing_definition objects.

4.3.7 Assembly_manufacturing_definition to Document_reference

Each Assembly_manufacturing_definition has assembly_drawing defined by exactly one Document_-reference. Each Document_reference defines assembly_drawing for zero, one, or many Assembly_-manufacturing_definition objects.

4.3.8 Assembly_manufacturing_definition to Centre_location

Each Assembly_manufacturing_definition has centre_of_gravity defined by exactly one Centre_-location object. Each Centre_location defines centre_of_gravity for zero, one, or many Assembly_-manufacturing_definition objects.

4.3.9 Assembly_manufacturing_position to Assembly_bounding_box

Each Assembly_manufacturing_position has bounding_box defined by exactly one Assembly_-bounding_box. Each Assembly_bounding_box defines bounding_box for zero, one, or many Assembly_manufacturing_position objects.

4.3.10 Assembly_relationship to Assembly

Each Assembly_relationship has item_1 defined by exactly one Assembly. Each Assembly defines item_1 for zero, one, or many Assembly_relationship objects.

4.3.11 Change_definition to Change

Each Change_definition has defined_for defined by one or many Change objects. Each Change defines defined_for for zero, one, or many Change_definition objects.

4.3.12 Change_impact to Versionable_object_change_event

Each Change_impact has impact defined by one or many Versionable_object_change_event objects. Each Versionable_object_change_event defines impact for zero, one, or many Change_impact objects.

4.3.13 Change_plan to Change_impact

Each Change_plan has planned_impact defined by exactly one Change_impact. Each Change_impact defines planned_impact for zero, one, or many Change_plan objects.

4.3.14 Change_plan to Change_request

Each Change_plan has chosen_solution_for defined by exactly one Change_request. Each Change_-request defines chosen_solution_for for zero, one, or many Change_plan objects.

4.3.15 Change_plan to Check

Each Change_plan has checks defined by zero, one, or many Check objects. Each Check defines checks for zero, one, or many Change_plan objects.

4.3.16 Change_realization to Change_impact

Each Change_realization has impact defined by exactly one Change_impact. Each Change_impact defines impact for zero, one, or many Change_realization objects.

4.3.17 Change_realization to Change_plan

Each Change_realization has realization_of defined by exactly one Change_plan. Each Change_plan defines realization_of for zero, one, or many Change_realization objects.

4.3.18 Change_realization to Check

Each Change_realization has checks defined by zero, one, or many Check objects. Each Check defines checks for zero, one, or many Change_realization objects.

4.3.19 Change_request to Change_impact

Each Change_request has solution_alternatives defined by zero, one, or many Change_impact objects. Each Change_impact defines solution_alternatives for zero, one, or many Change_request objects.

4.3.20 Class_and_statutory_designation to Class_notation

Each Class_and_statutory_designation has the_class defined by exactly one Class_notation. Each Class_notation defines the_class for zero, one, or many Class_and_statutory_designation objects.

4.3.21 Class_and_statutory_designation to Regulation

Each Class_and_statutory_designation has the_statutory defined by exactly one Regulation. Each Regulation defines the_statutory for zero, one, or many Class_and_statutory_designation objects.

4.3.22 Class_bsu to Supplier_bsu

Each Class_bsu has delivered_by defined by exactly one Supplier_bsu. Each Supplier_bsu defines delivered_by for zero, one, or many Class_bsu objects.

4.3.23 Composite_feature to Feature

Each Composite_feature has composed_of defined by one or many Feature objects. Each Feature defines composed_of for zero, one, or many Composite_feature objects.

4.3.24 Corner_cutout_boundary_relationship to Corner_cutout

Each Corner_cutout_boundary_relationship has item_1 defined by exactly one Corner_cutout. Each Corner_cutout defines item_1 for zero, one, or many Corner_cutout_boundary_relationship objects.

4.3.25 Corner_cutout_design_definition to Corner_cutout

Each Corner_cutout_design_definition has defined_for defined by one or many Corner_cutout objects. Each Corner_cutout defines defined_for for zero, one, or many Corner_cutout_design_-definition objects.

4.3.26 Corner_cutout_design_definition to Edge_feature

Each Corner_cutout_design_definition has defined_for defined by one or many Edge_feature objects. Each Edge_feature defines defined_for for zero, one, or many Corner_cutout_design_-definition objects.

4.3.27 Corrugated_part_approval to Corrugated_part_design_definition

Each Corrugated_part_approval has subject defined by exactly one Corrugated_part_design_-definition object. Each Corrugated_part_design_definition defines subject for zero, one, or many Corrugated_part_approval objects.

4.3.28 Corrugated_part_design_definition to Corrugated_part

Each Corrugated_part_design_definition has defined_for defined by one or many Corrugated_part objects. Each Corrugated_part defines defined_for for zero, one, or many Corrugated_part_design_-definition objects.

4.3.29 Corrugated_part_design_definition to Corrugated_structure_-boundary_relationship

Each Corrugated_part_design_definition has border defined by zero, one, or two Corrugated_-structure_boundary_relationship objects. Each Corrugated_structure_boundary_relationship defines border for zero, one, or many Corrugated_part_design_definition objects.

4.3.30 Corrugated_part_design_definition to Corrugation

Each Corrugated_part_design_definition has shape_aspect defined by exactly one Corrugation. Each Corrugation defines shape_aspect for zero, one, or many Corrugated_part_design_definition objects.

4.3.31 Corrugated_part_design_definition to Structural_system_-adjacency_relationship

Each Corrugated_part_design_definition has border defined by zero, one, or two Structural_system_-adjacency_relationship objects. Each Structural_system_adjacency_relationship defines border for zero, one, or many Corrugated_part_design_definition objects.

4.3.32 Corrugated_part_rejection to Corrugated_part_design_definition

Each Corrugated_part_rejection has subject defined by exactly one Corrugated_part_design_-definition object. Each Corrugated_part_design_definition defines subject for zero, one, or many Corrugated_part_rejection objects.

4.3.33 Corrugated_structure_design_definition to Corrugated_structure

Each Corrugated_structure_design_definition has defined_for defined by one or many Corrugated_-structure objects. Each Corrugated_structure defines defined_for for zero, one, or many Corrugated_structure_design_definition objects.

4.3.34 Corrugated_structure_design_definition to External_instance_-reference

Each Corrugated_structure_design_definition has moulded_surface defined by zero or one External_-instance_reference object. Each External_instance_reference defines moulded_surface for zero, one, or many Corrugated_structure_design_definition objects.

4.3.35 Corrugated_structure_design_definition to Non_manifold_-surface_shape

Each Corrugated_structure_design_definition has moulded_surface defined by zero or one Non_manifold_surface_shape object. Each Non_manifold_surface_shape defines moulded_surface for zero, one, or many Corrugated_structure_design_definition objects.

4.3.36 Corrugated_structure_functional_definition to Corrugated_-structure

Each Corrugated_structure_functional_definition has defined_for defined by one or many Corrugated_structure objects. Each Corrugated_structure defines defined _for for zero, one, or many Corrugated_structure_functional_definition objects.

4.3.37 Corrugated_structure_relationship to Corrugated_structure

Each Corrugated_structure_relationship has item_1 defined by exactly one Corrugated_structure. Each Corrugated_structure defines item_1 for zero, one, or many Corrugated_structure_relationship objects.

4.3.38 Definable_object to Global_id

Each Definable_object has id defined by exactly one Global_id. Each Global_id defines id for zero, one, or many Definable_object objects.

4.3.39 Definition to Global_id

Each Definition has id defined by exactly one Global_id. Each Global_id defines id for zero, one, or many Definition objects.

4.3.40 Design_load to Ship

Each Design_load has defined_for defined by one or many Ship objects. Each Ship defines defined_-for for zero, one, or many Design_load objects.

4.3.41 Design_still_water_shear_force to Design_swsf_values

Each Design_still_water_shear_force has table_of_shear_force_values defined by one or many Design_swsf_values objects. Each Design_swsf_values defines table_of_shear_force_values for zero, one, or many Design_still_water_shear_force objects.

4.3.42 Design_swsf_values to Spacing_position

Each Design_swsf_values has position defined by exactly one Spacing_position. Each Spacing_-position defines position for zero, one, or many Design_swsf_values objects.

4.3.43 Document_portion to Document

Each Document_portion has source defined by exactly one Document. Each Document defines source for zero, one, or many Document_portion objects.

4.3.44 Document_reference to Document

Each Document_reference has assigned_document defined by exactly one Document. Each Document defines assigned_document for zero, one, or many Document_reference objects.

NOTE This assertion is established though Document_referent select.

4.3.45 Document_reference to Document_portion

Each Document_reference has assigned_document defined by exactly one Document_portion. Each Document_portion defines assigned_document for zero, one, or many Document_reference objects.

NOTE This assertion is established though Document_referent select.

4.3.46 Drain_hole_cutout_design_definition to Edge_cutout

Each Drain_hole_cutout_design_definition has defined_for defined by one or many Edge_cutout objects. Each Edge_cutout defines defined_for for zero, one, or many Drain_hole_cutout_design_-definition objects.

4.3.47 Edge_cutout_functional_definition to Edge_cutout

Each Edge_cutout_functional_definition has defined_for defined by one or many Edge_cutout objects. Each Edge_cutout defines defined_for zero, one, or many Edge_cutout_functional_-definition objects.

4.3.48 Edge_feature_functional_definition to Edge_feature

Each Edge_feature_functional_definition has defined_for defined by one or many Edge_feature objects. Each Edge_feature defines defined_for zero, one, or many Edge_feature_functional_-definition objects.

4.3.49 Envisaged_version_creation to Versionable_object

Each Envisaged_version_creation has base defined by zero, one, or many Versionable_object objects. Each Versionable_object defines base for zero, one, or many Envisaged_version_creation objects.

4.3.50 Explicit_feature_design_definition to Advanced_brep_shape

Each Explicit_feature_design_definition has representations defined by zero or one Advanced_brep_-shape object. Each Advanced_brep_shape defines representations for zero, one, or many Explicit_-feature_design_definition objects.

4.3.51 Explicit_feature_design_definition to Edge_based_wireframe_-shape

Each Explicit_feature_design_definition has representations defined by zero or one Edge_based_-wireframe_shape object. Each Edge_based_wireframe_shape defines representations for zero, one, or many Explicit_feature_design_definition objects.

4.3.52 Explicit_feature_design_definition to Geometrically_bounded_-wireframe_shape

Each Explicit_feature_design_definition has representations defined by zero or one Geometrically_-bounded_wireframe_shape object. Each Geometrically_bounded_wireframe_shape defines representations for zero, one, or many Explicit_feature_design_definition objects.

4.3.53 Explicit_feature_design_definition to Non_manifold_surface_shape

Each Explicit_feature_design_definition has representations defined by zero or one Non_manifold_-surface_shape object. Each Non_manifold_surface_shape defines representations for zero, one, or many Explicit_feature_design_definition objects.

4.3.54 External_instance_reference to Global_id

Each External_instance_reference has target_guid defined by exactly one Global_id. Each Global_id defines target_guid for zero, one, or many External_instance_reference objects.

4.3.55 External_reference to Universal_resource_locator

Each External_reference has location defined by zero or one Universal_resource_locator object. Each Universal_resource_locator defines location for zero, one, or many External_reference objects.

4.3.56 Feature_design_definition to Feature

Each Feature_design_definition has defined_for defined by one or many Feature objects. Each Feature defines defined_for for zero, one, or many Feature_design_definition objects.

4.3.57 Feature_relationship to Feature

Each Feature_relationship has item_1 defined by exactly one Feature. Each Feature defines item_1 for zero, one, or many Feature_relationship objects.

4.3.58 Freeboard_characteristics to Loadline

Each Freeboard_characteristics has applicable_loadline defined by exactly one Loadline. Each Loadline defines applicable_loadline for zero, one, or many Freeboard_characteristics objects.

4.3.59 General_characteristics_definition to Ship

Each General_characteristics_definition has defined_for defined by one or many Ship objects. Each Ship defines defined_for for zero, one, or many General_characteristics_definition objects.

4.3.60 Homogeneous_ship_material_property to Part

Each Homogeneous_ship_material_property has defined_for defined by one or many Part objects. Each Part defines defined_for for zero, one, or many Homogeneous_ship_material_property objects.

4.3.61 Hull_applicability to Definition

Each Hull_applicability may apply to zero, one, or many Definition objects. Each Definition may be applicable to zero, one, or many Hull_applicability objects.

4.3.62 Hull_applicability to Item

Each Hull_applicability may apply to zero, one, or many Item objects. Each Item may be applicable to zero, one, or many Hull_applicability objects.

4.3.63 Hull_cross_section_design_definition to Flare_area

Each Hull_cross_section_design_definition has flare_area_buckling defined by exactly one Flare_-area. Each Flare_area defines flare_area_buckling for zero, one, or many Hull_cross_section_-design_definition objects.

4.3.64 Hull_cross_section_design_definition to Hull_cross_section

Each Hull_cross_section_design_definition has defined_for defined by exactly one Hull_cross_-section object. Each Hull_cross_section defines defined_for for zero, one, or many Hull_cross_-section_design_definition objects.

4.3.65 Hull_cross_section_result to Hull_cross_section_design_definition

Each Hull_cross_section_result has subject defined by exactly one Hull_cross_section_design_-definition. Each Hull_cross_section_design_definition defines subject for zero, one, or many Hull_-cross_section_result objects.

4.3.66 Interior_cutout_design_definition to Interior_cutout

Each Interior_cutout_design_definition has defined_for defined by one or many Interior_cutout objects. Each Interior_cutout defines defined_for for zero, one, or many Interior_cutout_design_-definition objects.

4.3.67 Interior_cutout_functional_definition to Interior_cutout

Each Interior_cutout_functional_definition has defined_for defined by one or many Interior_cutout objects. Each Interior_cutout defines defined_for for zero, one, or many Interior_cutout_functional_-definition objects.

4.3.68 Item to External_reference

Each Item has documentation defined by zero, one, or many External_reference objects. Each External_reference defines documentation for zero, one, or many Item objects.

4.3.69 Item to Ship

Each Item has ship_context defined by zero or one Ship object. Each Ship defines ship_items for one or many Item objects.

4.3.70 Item_relationship to External_instance_reference

Each Item_relationship has external_item_1 defined by zero or one External_instance_reference object. Each External_instance_reference defines external_item_1 for zero, one, or many Item_-relationship objects.

Each Item_relationship has external_item_2 defined by zero or one External_instance_reference object. Each External_instance_reference defines external_item_2 for zero, one, or many Item_-relationship objects.

4.3.71 Item_relationship to Item

Each Item_relationship has item_1 defined by zero or one Item object. Each Item defines item_1 for zero, one, or many Item_relationship objects.

Each Item_relationship has item_2 defined by zero or one Item object. Each Item defines item_2 for zero, one, or many Item_relationship objects.

4.3.72 Item_structure to External_instance_reference

Each Item_structure has external_items defined by zero, one, or many External_instance_reference objects. Each External_instance_reference defines external_items for zero, one, or many Item_-structure objects.

Each Item_structure has external_relationships defined by zero, one, or many External_instance_-reference objects. Each External_instance_reference defines external_relationships for zero, one, or many Item_structure objects.

4.3.73 Item_structure to Item

Each Item_structure has items defined by zero, one, or many Item objects. Each Item defines items for zero, one, or many Item_structure objects.

4.3.74 Item_structure to Item_relationship

Each Item_structure has relationships defined by zero, one, or many Item_relationship objects. Each Item_relationship defines relationships for zero, one, or many Item_structure objects.

4.3.75 Library_definition to Library_element_reference

Each Library_definition has source defined by exactly one Library_element_reference. Each Library_element_reference defines source for zero, one, or many Library_definition objects.

4.3.76 Library_element_reference to Class_bsu

Each Library_element_reference has library_identifier defined by exactly one Class_bsu. Each Class_bsu defines library_identifier for zero, one, or many Library_element_reference objects.

4.3.77 Library_element_reference to Property_value

Each Library_element_reference has property_value_pairs defined by zero, one, or many Property_-value objects. Each Property_value defines property_value_pairs for zero, one, or many Library_-element_reference objects.

4.3.78 Lightship_definition to Centre_location

Each Lightship_definition has centre_of_gravity defined by exactly one Centre_location object. Each Centre_location defines centre_of_gravity for zero, one, or many Lightship_definition objects.

4.3.79 Lightship_definition to Lightship_weight_item

Each Lightship_definition has lightship_items defined by zero, one, or many Lightship_weight_item objects. Each Lightship_weight_item defines lightship_items for zero, one, or many Lightship_-definition objects.

4.3.80 Lightship_definition to Ship

Each Lightship_definition has defined_for defined by exactly one Ship object. Each Ship defines defined_for for zero, one, or many Lightship_definition objects.

4.3.81 Local_co_ordinate_system to Global_axis_placement

Each Local_co_ordinate_system has parent defined by zero or one Global_axis_placement object. Each Global_axis_placement defines parent for zero, one, or many Local_co_ordinate_system objects.

4.3.82 Local_co_ordinate_system to Local_co_ordinate_system

Each Local_co_ordinate_system has parent defined by zero or one Local_co_ordinate_system object. Each Local_co_ordinate_system defines parent for zero, one, or many Local_co_ordinate_system objects.

4.3.83 Local_co_ordinate_system_with_position_reference to Spacing_-position

Each Local_co_ordinate_system_with_position_reference has longitudinal_ref defined by exactly one Spacing_position. Each Spacing_position defines longitudinal_ref for zero, one, or many Local_co_ordinate_system_with_position_reference objects.

Each Local_co_ordinate_system_with_position_reference has transversal_ref defined by exactly one Spacing_position. Each Spacing_position defines transversal_ref for zero, one, or many Local_-co_ordinate_system_with_position_reference objects.

Each Local_co_ordinate_system_with_position_reference has vertical_ref_defined by exactly one Spacing_position. Each Spacing_position defines vertical_ref for zero, one, or many Local_-co_ordinate_system_with_position_reference objects.

4.3.84 Longitudinal_table to Longitudinal_position

Each Longitudinal_table has spacing_table_representations defined by zero, one, or many Longitudinal_position objects. Each Longitudinal_position defines spacing_table_representations for zero, one, or many Longitudinal_table objects.

4.3.85 Maximum_permissible_still_water_bending_moment to Maximum_permissible_swbm_values

Each Maximum_permissible_still_water_bending_moment has table_of_max_values defined by one or many Maximum_permissible_swbm_values objects. Each Maximum_permissible_swbm_values defines table_of_max_values for zero, one, or many Maximum_permissible_still_water_bending_-moment objects.

4.3.86 Maximum_permissible_still_water_shear_force to Maximum_-permissible_swsf_values

Each Maximum_permissible_still_water_shear_force has table_of_max_shear_force_values defined by one or many Maximum_permissible_swsf_values objects. Each Maximum_permissible_swsf_-values defines table_of_max_shear_force_values for zero, one, or many Maximum_permissible_-still_water_shear_force objects.

4.3.87 Maximum_permissible_swbm_values to Spacing_position

Each Maximum_permissible_swbm_values has position defined by exactly one Spacing_position. Each Spacing_position defines position for zero, one, or many Maximum_permissible_swbm_values objects.

4.3.88 Maximum_permissible_swsf_values to Spacing_position

Each Maximum_permissible_swsf_values has position defined by exactly one Spacing_position. Each Spacing_position defines position for zero, one, or many Maximum_permissible_swsf_values objects.

4.3.89 Moment_3d to Centre_location

Each Moment_3d has origin defined by exactly one Centre_location object. Each Centre_location defines origin for zero, one, or many Moment_3d objects.

4.3.90 Panel_system_design_definition to External_instance_reference

Each Panel_system_design_definition has moulded_surface defined by zero or one External_-instance_reference object. Each External_instance_reference defines moulded_surface for zero, one, or many Panel_system_design_definition objects.

4.3.91 Panel_system_design_definition to Non_manifold_surface_shape

Each Panel_system_design_definition has moulded_surface defined by zero or one Non_manifold_surface_shape object. Each Non_manifold_surface_shape defines moulded_surface for zero, one, or many Panel_system_design_definition objects.

4.3.92 Panel_system_design_definition to Panel_system

Each Panel_system_design_definition has defined_for defined by one or many Panel_system objects. Each Panel_system defines defined_for for zero, one, or many Panel_system_design_definition objects.

4.3.93 Panel_system_design_definition to Panel_system_boundary_-relationship

Each Panel_system_design_definition has border defined by zero, one, or many Panel_system_-boundary_relationship objects. Each Panel_system_boundary_relationship defines border for zero, one, or many Panel_system_design_definition objects.

4.3.94 Panel_system_design_definition to Panel_system_curve_boundary

Each Panel_system_design_definition has border defined by zero, one, or many Panel_system_-curve_boundary objects. Each Panel_system_curve_boundary defines border for zero, one, or many Panel_system_design_definition objects.

4.3.95 Panel_system_design_definition to Panel_system_plane_boundary

Each Panel_system_design_definition has border defined by zero, one, or many Panel_system_-plane_boundary objects. Each Panel_system_plane_boundary defines border for zero, one, or many Panel_system_design_definition objects.

4.3.96 Panel_system_design_definition to Structural_system_adjacency_-relationship

Each Panel_system_design_definition has border defined by zero, one, or many Structural_system_-adjacency_relationship objects. Each Structural_system_adjacency_relationship defines border for zero, one, or many Panel_system_design_definition objects.

4.3.97 Panel_system_functional_definition to Panel_system

Each Panel_system_functional_definition has defined_for defined by one or many Panel_system objects. Each Panel_system defines defined_for for zero, one, or many Panel_system_functional_-definition objects.

4.3.98 Panel_system_relationship to Panel_system

Each Panel_system_relationship has item_1 defined by exactly one Panel_system. Each Panel_-system defines the item_1 for zero, one, or many Panel_system_relationship objects.

4.3.99 Part_edge_cutout_design_definition to Edge_cutout

Each Part_edge_cutout_design_definition has defined_for defined by one or many Edge_cutout objects. Each Edge_cutout defines defined_for for zero, one, or many Part_edge_cutout_design_-definition objects.

4.3.100 Part_relationship to Part

Each Part_relationship has item_1 defined by exactly one Part. Each Part defines item_1 for zero, one, or many Part_relationship objects.

4.3.101 Plate_approval to Plate_design_definition

Each Plate_approval has subject defined by exactly one Plate_design_definition. Each Plate_-design_definition defines subject for zero, one, or many Plate_approval objects.

4.3.102 Plate_boundary_relationship to Seam

Each Plate_boundary_relationship has item_2 defined by exactly one Seam. Each Seam defines item_2 for zero, one, or many Plate_boundary_relationship objects.

4.3.103 Plate_design_definition to External_instance_reference

Each Plate_design_definition has moulded_surface defined by zero or one External_instance_-reference object. Each External_instance_reference object defines moulded_surface for zero, one, or many Plate_design_definition objects.

4.3.104 Plate_design_definition to Non_manifold_surface_shape

Each Plate_design_definition has moulded_surface defined by zero or one Non_manifold_surface_shape object. Each Non_manifold_surface_shape object defines moulded_-surface for zero, one, or many Plate_design_definition objects.

4.3.105 Plate_design_definition to Panel_system_boundary_relationship

Each Plate_design_definition has border defined by zero, one, or many Panel_system_boundary_-relationship objects. Each Panel_system_boundary_relationship defines border for zero, one, or many Plate_design_definition objects.

4.3.106 Plate_design_definition to Panel_system_curve_boundary

Each Plate_design_definition has border defined by zero, one, or many Panel_system_curve_-boundary objects. Each Panel_system_curve_boundary defines border for zero, one, or many Plate_-design_definition objects.

4.3.107 Plate_design_definition to Panel_system_plane_boundary

Each Plate_design_definition has border defined by zero, one, or many Panel_system_plane_-boundary objects. Each Panel_system_plane_boundary defines border for zero, one, or many Plate_-design_definition objects.

4.3.108 Plate_design_definition to Structural_system_adjacency_-relationship

Each Plate_design_definition has border defined by zero, one, or many Structural_system_-adjacency_relationship objects. Each Structural_system_adjacency_relationship defines border for zero, one, or many Plate_design_definition objects.

4.3.109 Plate_design_definition to Plate

Each Plate_design_definition has defined_for defined by one or many Plate objects. Each Plate defines defined_for for zero, one, or many Plate_design_definition objects.

4.3.110 Plate_design_definition to Plate_boundary_relationship

Each Plate_design_definition has border defined by zero, one, or many Plate_boundary_relationship objects. Each Plate_boundary_relationship defines border for zero, one, or many Plate_design_-definition objects.

4.3.111 Plate_functional_definition to Plate

Each Plate_functional_definition has defined_for defined by one or many Plate objects. Each Plate defines defined_for for zero, one, or many Plate_functional_definition objects.

4.3.112 Plate_rejection to Plate_design_definition

Each Plate_rejection has subject defined by exactly one Plate_design_definition. Each Plate_-design_definition defines subject for zero, one, or many Plate_rejection objects.

4.3.113 Plate_relationship to Plate

Each Plate_relationship has item_1 defined by exactly one Plate. Each Plate defines item_1 for zero, one, or many Plate_relationship objects.

4.3.114 Plate_renewal_definition to Plate

Each Plate_renewal_definition has defined_for defined by one or many Plate objects. Each Plate defines defined_for for zero, one, or many Plate_renewal_definition objects.

4.3.115 Plate_strake_functional_definition to Plate_strake

Each Plate_strake_functional_definition has defined_for defined by one or many Plate_strake objects. Each Plate_strake defines defined_for for zero, one, or many Plate_strake_functional_-definition objects.

4.3.116 Position_feature_design_definition to Position_feature

Each Position_feature_design_definition has defined_for defined by one or many Position_feature objects. Each Position_feature defines defined_for for zero, one, or many Position_feature_design_-definition objects.

4.3.117 Position_feature_design_definition to Advanced_brep_shape

Each Position_feature_design_definition has representations defined by zero, one, or many Advanced_brep_shape objects. Each Advanced_brep_shape defines representations for zero, one, or many Position_feature_design_definition objects.

4.3.118 Position_feature_design_definition to Edge_based_wireframe_-shape

Each Position_feature_design_definition has representations defined by zero, one, or many Edge_-based_wireframe_shape objects. Each Edge_based_wireframe_shape defines representations for zero, one, or many Position_feature_design_definition objects.

4.3.119 Position_feature_design_definition to Geometrically_bounded_-wireframe_shape

Each Position_feature_design_definition has representations defined by zero, one, or many Geometrically_bounded_wireframe_shape objects. Each Geometrically_bounded_wireframe_shape defines representations for zero, one, or many Position_feature_design_definition objects.

4.3.120 Position_feature_design_definition to Non_manifold_surface_-shape

Each Position_feature_design_definition has representations defined by zero, one, or many Non_-manifold_surface_shape objects. Each Non_manifold_surface_shape defines representations for zero, one, or many Position_feature_design_definition objects.

4.3.121 Position_feature_relationship to Position_feature

Each Position_feature_relationship has item_1 defined by exactly one Position_feature. Each Position_feature defines item_1 for zero, one, or many Position_feature_relationship objects.

4.3.122 Profile_approval to Profile_design_definition

Each Profile_approval has subject defined by exactly one Profile_design_definition. Each Profile_-design_definition defines subject for zero, one, or many Profile_approval objects.

4.3.123 Profile_cross_section to Section_properties

Each Profile_cross_section has section_properties defined by zero or one Section_properties object. Each Section_properties defines section_properties for zero, one, or many Profile_cross_section objects.

4.3.124 Profile_design_definition to External_instance_reference

Each Profile_design_definition has trace_line defined by zero or one External_instance_reference object. Each External_instance_reference object defines the trace_line for zero, one, or many Profile_design_definition objects.

4.3.125 Profile_design_definition to Profile

Each Profile_design_definition has defined_for defined by one or many Profile objects. Each Profile defines defined_for for zero, one, or many Profile_design_definition objects.

4.3.126 Profile_design_definition to Profile_boundary_relationship

Each Profile_design_definition has border defined by zero or one Profile_boundary_relationship objects. Each Profile_boundary_relationship defines border for zero, one, or many Profile_design_-definition objects.

4.3.127 Profile_design_definition to Profile_cross_section

Each Profile_design_definition has cross_section defined by exactly one Profile_cross_section. Each Profile_cross_section defines cross_section for zero, one, or many Profile_design_definition objects.

4.3.128 Profile_design_definition to Profile_curve_trace_line

Each Profile_design_definition has trace_line defined by zero or one Profile_curve_trace_line object. Each Profile_curve_trace_line defines the trace_line for zero, one, or many Profile_design_definition objects.

4.3.129 Profile_design_definition to Profile_trace_line_relationship

Each Profile_design_definition has trace_line defined by zero or one Profile_trace_line_relationship object. Each Profile_trace_line_relationship defines the trace_line for zero, one, or many Profile_-design_definition objects.

4.3.130 Profile_design_definition to Twist_location

Each Profile_design_definition has twist defined by zero, one, or many Twist_location objects. Each Twist_location defines twist for zero, one, or many Profile_design_definition objects.

4.3.131 Profile_functional_definition to Profile

Each Profile_functional_definition has defined_for defined by one or many Profile objects. Each Profile defines defined_for for zero, one, or many Profile_functional_definition objects.

4.3.132 Profile_manufacturing_definition to Profile

Each Profile_manufacturing_definition has defined_for defined by one or many Profile objects. Each Profile defines defined_for for zero, one, or many Profile_manufacturing_definition objects.

4.3.133 Profile_rejection to Profile_design_definition

Each Profile_rejection has subject defined by exactly one Profile_design_definition. Each Profile_-design_definition defines subject for zero, one, or many Profile_rejection objects.

4.3.134 Profile_relationship to Profile

Each Profile_relationship has item_1 defined by exactly one Profile. Each Profile defines item_1 for zero, one, or many Profile_relationship objects.

4.3.135 Profile_renewal_definition to Profile

Each Profile_renewal_definition has defined_for defined by one or many Profile objects. Each Profile defines defined_for for zero, one, or many Profile_renewal_definition objects.

4.3.136 Profile_trace_line_relationship to External_instance_reference

Each Profile_trace_line_relationship has external_item_2 defined by exactly one External_instance_-reference. Each External_instance_reference defines external_item_2 for zero, one, or many Profile_trace_line_relationship objects.

4.3.137 Profile_trace_line_relationship to Interior_cutout

Each Profile_trace_line_relationship has item_2 defined by exactly one Interior_cutout. Each Interior_cutout defines item_2 for zero, one, or many Profile_trace_line_relationship objects.

4.3.138 Profile_trace_line_relationship to Panel_system

Each Profile_trace_line_relationship has item_2 defined by exactly one Panel_system. Each Panel_-system defines item_2 for zero, one, or many Profile_trace_line_relationship objects.

4.3.139 Profile_trace_line_relationship to Seam

Each Profile_trace_line_relationship has item_2 defined by exactly one Seam. Each Seam defines item_2 for zero, one, or many Profile_trace_line_relationship objects.

4.3.140 Property_bsu to Class_bsu

Each Property_bsu has name_scope defined by exactly one Class_bsu. Each Class_bsu defines name_scope for zero, one, or many Property_bsu objects.

4.3.141 Property_value to Property_bsu

Each Property_value has property_identier defined by exactly one Property_bsu. Each Property_bsu defines property_identier for zero, one, or many Property_value objects.

4.3.142 Reason_for_decision to Document_reference

Each Reason_for_decision has rule_reference defined by exactly one Document_reference. Each Document_reference defines rule_reference for zero, one, or many Reason_for_decision objects.

4.3.143 Regulation to External_reference

Each Regulation has international_regulations defined by zero, one, or many External_reference objects. Each External_reference defines international_regulations for zero, one, or many Regulation objects.

Each Regulation has national_regulations defined by zero, one, or many External_reference objects. Each External_reference defines national_regulations for zero, one, or many Regulation objects.

Each Regulation has standards defined by zero, one, or many External_reference objects. Each External_reference defines standards for zero, one, or many Regulation objects.

4.3.144 Revision to Versionable_object

Each Revision has members defined by one or many Versionable_object objects. Each Versionable_-object defines members for zero, one, or many Revision objects.

4.3.145 Revision_with_context to Definable_object

Each Revision_with_context has context_of_revision defined by exactly one Definable_object. Each Definable_object defines context_of_revision for zero, one, or many Revision_with_context objects.

4.3.146 Seam to Panel_system

Each Seam has parent defined by exactly one Panel_system. Each Panel_system defines parent for zero, one, or many Seam objects.

4.3.147 Seam_design_definition to Seam

Each Seam_design_definition has border defined by zero, one, or two Seam objects. Each Seam defines border for zero, one, or many Seam_design_definition objects.

Each Seam_design_definition has defined_for defined by one or many Seam objects. Each Seam defines defined_for for zero, one, or many Seam_design_definition objects.

4.3.148 Seam_design_definition to Seam_curve_relationship

Each Seam_design_definition has seam_curve defined by zero or one Seam_curve_relationship object. Each Seam_curve_relationship defines seam_curve for zero, one, or many Seam_design_-definition objects.

4.3.149 Ship_material_property to External_reference

Each Ship_material_property has material_reference defined by zero or one External_reference object. Each External_reference defines material_reference for zero, one, or many Ship_material_-property objects.

4.3.150 Shiptype to Ship

Each Shiptype has defined_for defined by one or many Ship objects. Each Ship defines defined_for for zero, one, or many Shiptype objects.

4.3.151 Spacing_position_with_offset to Spacing_position

Each Spacing_position_with_offset has relating_spacing_position defined by exactly one Spacing_-position. Each Spacing_position defines relating_spacing_position for zero, one, or many Spacing_-position_with_offset objects.

4.3.152 Spacing_table to Spacing_position

Each Spacing_table has spacing_table_representations defined by zero, one, or many Spacing_-position objects. Each Spacing_position defines spacing_table_representations for zero, one, or many Spacing_table objects.

4.3.153 Structural_added_material_boundary_relationship to Structural_added_material_feature

Each Structural_added_material_boundary_relationship has item_1 defined by exactly one Structural_added_material_feature. Each Structural_added_material_feature defines item_1 for zero, one, or many Structural_added_material_boundary_relationship objects.

4.3.154 Structural_added_material_feature_design_definition to Structural_added_material_feature

Each Structural_added_material_feature_design_definition has defined_for defined by one or many Structural_added_material_feature objects. Each Structural_added_material_feature defines defined_for for zero, one, or many Structural_added_material_feature_design_definition objects.

4.3.155 Structural_class_approval_result to Reason_for_decision

Each Structural_class_approval_result has explanations defined by zero, one, or many Reason_for_-decision objects. Each Reason_for_decision defines explanations for zero, one, or many Structural_-class_approval_result objects.

4.3.156 Structural_class_conditional_approval to Reason_for_decision

Each Structural_class_conditional_approval has explanations defined by one or many Reason_for_-decision objects. Each Reason_for_decision defines explanations for zero, one, or many Structural_-class_conditional_approval objects.

4.3.157 Structural_class_rejection to Definition

Each Structural_class_rejection has proposed_alternative defined by zero, one, or many Definition objects. Each Definition defines proposed_alternative for zero, one, or many Structural_class_-rejection objects.

4.3.158 Structural_class_rejection to Reason_for_decision

Each Structural_class_rejection has explanations defined by one or many Reason_for_decision objects. Each Reason_for_decision defines explanations for zero, one, or many Structural_class_-rejection objects.

4.3.159 Structural_cutout_boundary_relationship to Structural_cutout

Each Structural_cutout_boundary_relationship has item_1 defined by exactly one Structural_cutout. Each Structural_cutout defines item_1 for zero, one, or many Structural_cutout_boundary_-relationship objects.

4.3.160 Structural_cutout_manufacturing_relationship to Structural_-cutout

Each Structural_cutout_manufacturing_relationship has item_1 defined by exactly one Structural_-cutout. Each Structural_cutout defines item_1 for zero, one, or many Structural_cutout_-manufacturing_relationship objects.

4.3.161 Structural_feature to Item

Each Structural_feature has parent defined by exactly one Item. Each Item defines parent for zero, one, or many Structural_feature objects.

4.3.162 Structural_feature_relationship to Structural_feature

Each Structural_feature_relationship has item_1 defined by exactly one Structural_feature. Each Structural_feature defines item_1 for zero, one, or many Structural_feature_relationship objects.

4.3.163 Structural_manufacturing_feature to Structural_part

Each Structural_manufacturing_feature has parent defined by exactly one Structural_part. Each Structural_part defines parent for zero, one, or many Structural_manufacturing_feature objects.

4.3.164 Structural_part_connection_implementation to Structural_part_-joint

Each Structural_part_connection_implementation has realization_of defined by exactly one Structural_part_joint. Each Structural_part_joint defines realization_of for zero, one, or many Structural_part_connection_implementation objects.

4.3.165 Structural_part_design_definition to Advanced_brep_shape

Each Structural_part_design_definition has representations defined by zero, one, or many Advanced_brep_shape objects. Each Advanced_brep_shape defines representations for zero, one, or many Structural_part_design_definition objects.

4.3.166 Structural_part_design_definition to Edge_based_wireframe_-shape

Each Structural_part_design_definition has representations defined by zero, one, or many Edge_-based_wireframe_shape objects. Each Edge_based_wireframe_shape defines representations for zero, one, or many Structural_part_design_definition objects.

4.3.167 Structural_part_design_definition to Geometrically_bounded_-wireframe_shape

Each Structural_part_design_definition has representations defined by zero, one, or many Geometrically_bounded_wireframe_shape objects. Each Geometrically_bounded_wireframe_shape defines representations for zero, one, or many Structural_part_design_definition objects.

4.3.168 Structural_part_design_definition to Non_manifold_surface_-shape

Each Structural_part_design_definition has representations defined by zero, one, or many Non_-manifold_surface_shape objects. Each Non_manifold_surface_shape defines representations for zero, one, or many Structural_part_design_definition objects.

4.3.169 Structural_part_design_definition to Structural_part

Each Structural_part_design_definition has defined_for defined by one or many Structural_part objects. Each Structural_part defines defined_for for zero, one, or many Structural_part_design_-definition objects.

4.3.170 Structural_part_functional_definition to Structural_part

Each Structural_part_functional_definition has defined_for defined by one or many Structural_part objects. Each Structural_part defines defined_for for zero, one, or many Structural_part_functional_-definition objects.

4.3.171 Structural_part_joint_design_definition to Structural_part_joint

Each Structural_part_joint_design_definition has defined_for defined by one or many Structural_-part_joint objects. Each Structural_part_joint defines defined_for for zero, one, or many Structural_-part_joint_design_definition objects.

4.3.172 Structural_part_manufacturing_definition to Structural_part

Each Structural_part_manufacturing_definition has defined_for defined by one or many Structural_-part objects. Each Structural_part defines defined_for for zero, one, or many Structural_part_-manufacturing_definition objects.

4.3.173 Structural_part_penetration_relationship to Structural_cutout

Each Structural_part_penetration_relationship has penetration_result defined by one or many Structural_cutout objects. Each Structural_cutout defines penetration_result for zero, one, or many Structural_part_penetration_relationship objects.

4.3.174 Structural_part_penetration_relationship to Structural_part

Each Structural_part_penetration_relationship has item_1 defined by exactly one Structural_part. Each Structural_part defines item_1 for zero, one, or many Structural_part_penetration_relationship objects.

4.3.175 Structural_part_relationship to Structural_part

Each Structural_part_relationship has item_1 defined by exactly one Structural_part. Each Structural_part defines item_1 for zero, one, or many Structural_part_relationship objects.

4.3.176 Structural_part_symmetry_relationship to Structural_part

Each Structural_part_symmetry_relationship has item_2 defined by exactly one Structural_part. Each Structural_part defines item_2 for zero, one, or many Structural_part_symmetry_relationship objects.

4.3.177 Structural_system_adjacency_relationship to Structural_system

Each Structural_system_adjacency_relationship has item_2 defined by exactly one Structural_-system. Each Structural_system defines item_2 for zero, one, or many Structural_system_-adjacency_relationship objects.

4.3.178 Structural_system_design_definition to Advanced_brep_shape

Each Structural_system_design_definition has representations defined by zero, one, or many Advanced_brep_shape objects. Each Advanced_brep_shape defines representations for zero, one, or many Structural_system_design_definition objects.

4.3.179 Structural_system_design_definition to Edge_based_wireframe_-shape

Each Structural_system_design_definition has representations defined by zero, one, or many Edge_-based_wireframe_shape objects. Each Edge_based_wireframe_shape defines representations for zero, one, or many Structural_system_design_definition objects.

4.3.180 Structural_system_design_definition to Geometrically_bounded_-wireframe_shape

Each Structural_system_design_definition has representations defined by zero, one, or many Geometrically_bounded_wireframe_shape objects. Each Geometrically_bounded_wireframe_shape defines representations for zero, one, or many Structural_system_design_definition objects.

4.3.181 Structural_system_design_definition to Non_manifold_surface_-shape

Each Structural_system_design_definition has representations defined by zero, one, or many Non_-manifold_surface_shape objects. Each Non_manifold_surface_shape defines representations for zero, one, or many Structural_system_design_definition objects.

4.3.182 Structural_system_design_definition to Structural_system

Each Structural_system_design_definition has defined_for defined by one or many Structural_system objects. Each Structural_system defines defined_for for zero, one, or many Structural_system_-design_definition objects.

4.3.183 Structural_system_functional_definition to Structural_system

Each Structural_system_functional_definition has defined_for defined by one or many Structural_-system objects. Each Structural_system defines defined_for for zero, one, or many Structural_-system_functional_definition objects.

4.3.184 Structural_system_penetration_relationship to Structural_cutout

Each Structural_system_penetration_relationship has penetration_result defined by one or many Structural_cutout objects. Each Structural_cutout defines penetration_result for zero, one, or many Structural_system_penetration_relationship objects.

4.3.185 Structural_system_penetration_relationship to Structural_system

Each Structural_system_penetration_relationship has item_1 defined by exactly one Structural_-system. Each Structural_system defines item_1 for zero, one, or many Structural_system_-penetration_relationship objects.

4.3.186 Structural_system_relationship to Structural_system

Each Structural_system_relationship has item_1 defined by exactly one Structural_system. Each Structural_system defines item_1 for zero, one, or many Structural_system_relationship objects.

4.3.187 Structural_system_symmetry_relationship to Structural_system

Each Structural_system_symmetry_relationship has item_2 defined by exactly one Structural_system. Each Structural_system defines item_2 for zero, one, or many Structural_system_symmetry_-relationship objects.

4.3.188 Structural_weld_shrinkage_allowance_feature_design_definition to Structural_weld_shrinkage_allowance_feature

Each Structural_weld_shrinkage_allowance_feature_design_definition has defined_for defined by one or many Structural_weld_shrinkage_allowance_feature objects. Each Structural_weld_-shrinkage_allowance_feature defines defined_for for zero, one, or many Structural_weld_shrinkage_-allowance_feature_design_definition objects.

4.3.189 System_design_definition to System

Each System_design_definition has defined_for defined by one or many System objects. Each System defines defined_for for zero, one, or many System_design_definition objects.

4.3.190 System_relationship to System

Each System_relationship has item_1 defined by exactly one System. Each System defines item_1 for zero, one, or many System_relationship objects.

4.3.191 Transversal_table to Transversal_position

Each Transversal_table has spacing_table_representations defined by zero, one, or many Transversal_position objects. Each Transversal_position defines spacing_table_representations for zero, one, or many Transversal_table objects.

4.3.192 Version_creation to Versionable_object

Each Version_creation has base defined by zero, one, or many Versionable_object objects. Each Versionable_object defines base for zero, one, or many Version_creation objects.

Each Version_creation has subject defined by exactly one Versionable_object. Each Versionable_-object defines subject for zero, one, or many Version_creation objects.

4.3.193 Version_deletion to Versionable_object

Each Version_deletion has subject defined by exactly one Versionable_object. Each Versionable_-object defines subject for zero, one, or many Version_deletion objects.

4.3.194 Version_history to Version_relationship

Each Version_history has relationships defined by zero, one, or many Version_relationship objects. Each Version_relationship defines relationships for zero, one, or many Version_history objects.

4.3.195 Version_history to Versionable_object

Each Version_history has current_version defined by exactly one Versionable_object. Each Versionable_object defines current_version for zero, one, or many Version_history objects.

Each Version_history has versions defined by one or many Versionable_object objects. Each Versionable_object defines versions for zero, one, or many Version_history objects.

4.3.196 Version_modification to Versionable_object

Each Version_modification has base defined by one or many Versionable_object objects. Each Versionable_object defines base for zero, one, or many Version_modification objects.

Each Version_modification has subject defined by exactly one Versionable_object. Each Versionable_object defines subject for zero, one, or many Version_modification objects.

4.3.197 Version_relationship to Versionable_object

Each Version_relationship has predecessor defined by exactly one Versionable_object. Each Versionable_object defines predecessor for zero, one, or many Version_relationship objects.

Each Version_relationship has successor defined by exactly one Versionable_object. Each Versionable_object defines successor for zero, one, or many Version_relationship objects.

4.3.198 Vertical_table to Vertical_position

Each Vertical_table has spacing_table_representations defined by zero, one, or many Vertical_-position objects. Each Vertical_position defines spacing_table_representations for zero, one, or many Vertical_table objects.

4.3.199 Weight_and_centre_of_gravity to Centre_location

Each Weight_and_centre_of_gravity has centre_of_gravity defined by exactly one Centre_location object. Each Centre_location defines centre_of_gravity for zero, one, or many Weight_and_centre_-of_gravity objects.

4.3.200 Weight_and_centre_of_gravity to Moment_3d

Each Weight_and_centre_of_gravity has moment defined by exactly one Moment_3d. Each Moment_3d defines moment for zero, one, or many Weight_and_centre_of_gravity objects.

4.3.201 Weld to Welded_joint

Each Weld has realization_of defined by exactly one Welded_joint. Each Welded_joint defines realization_of for zero, one, or many Weld objects.

4.3.202 Weld_design_definition to Weld

Each Weld_design_definition has defined_for defined by one or many Weld objects. Each Weld defines defined_for for zero, one, or many Weld_design_definition objects.

4.3.203 Weld_filler_material to Electrode_chemical_composition

Each Weld_filler_material has chemical_composition defined by zero or one Electrode_chemical_-composition object. Each Electrode_chemical_composition defines chemical_composition for zero, one, or many Weld_filler_material objects.

4.3.204 Weld_filler_material to Weld

Each Weld_filler_material has defined_for defined by one or many Weld objects. Each Weld defines defined_for for zero, one, or many Weld_filler_material objects.

4.3.205 Weld_manufacturing_definition to Document

Each Weld_manufacturing_definition has welding_procedures defined by one or many Document objects. Each Document defines welding_procedures for zero, one, or many Weld_manufacturing_-definition objects.

4.3.206 Weld_manufacturing_definition to Weld

Each Weld_manufacturing_definition has defined_for defined by one or many Weld objects. Each Weld defines defined_for for zero, one, or many Weld_manufacturing_definition objects.

4.3.207 Weld_manufacturing_definition to Weld_testing

Each Weld_manufacturing_definition has weld_test defined by one or many Weld_testing objects. Each Weld_testing defines weld_test for zero, one, or many Weld_manufacturing_definition objects.

4.3.208 Weld_manufacturing_definition to Welding_sequence

Each Weld_manufacturing_definition has sequences defined by one or many Welding_sequence objects. Each Welding_sequence defines sequences for zero, one, or many Weld_manufacturing_-definition objects.

4.3.209 Weld_testing to Document

Each Weld_testing has test_sequence defined by zero, one, or many Document objects. Each Document defines test_sequence for zero, one, or many Weld_testing objects.

4.3.210 Welded_joint_design_definition to Advanced_brep_shape

Each Welded_joint_design_definition has representations defined by zero, one, or many Advanced_-brep_shape objects. Each Advanced_brep_shape defines representations for zero, one, or many Welded_joint_design_definition objects.

4.3.211 Welded_joint_design_definition to Edge_based_wireframe_shape

Each Welded_joint_design_definition has representations defined by zero, one, or many Edge_-based_wireframe_shape objects. Each Edge_based_wireframe_shape defines representations for zero, one, or many Welded_joint_design_definition objects.

4.3.212 Welded_joint_design_definition to Geometrically_bounded_-wireframe_shape

Each Welded_joint_design_definition has representations defined by zero, one, or many Geometrically_bounded_wireframe_shape objects. Each Geometrically_bounded_wireframe_shape defines representations for zero, one, or many Welded_joint_design_definition objects.

4.3.213 Welded_joint_design_definition to Non_manifold_surface_shape

Each Welded_joint_design_definition has representations defined by zero, one, or many Non_-manifold_surface_shape objects. Each Non_manifold_surface_shape defines representations for zero, one, or many Welded_joint_design_definition objects.

4.3.214 Welded_joint_design_definition to Welded_joint

Each Welded_joint_design_definition has defined_for defined by one or many Welded_joint objects. Each Welded_joint defines defined_for for zero, one, or many Welded_joint_design_definition objects.

5 Application interpreted model

5.1 Mapping specification

This clause contains the mapping specification that shows how each UoF and application object of this part of ISO 10303 (see clause 4) maps to one or more AIM constructs (see annex A). Each mapping specifies up to five elements.

Application element: The mapping for each application element is specified in a separate subclause below. Application object names are given in title case. Attribute names and assertions are listed after the application object to which they belong and are given in lower case.

AIM element: The name of one or more AIM entity data types (see annex A), the term "IDENTICAL MAPPING", or the term "PATH". AIM entity data type names are given in lower case. Attributes of AIM entity data types are referred to as <entity name>.<attribute name>. The mapping of an application element may involve more than one AIM element. Each of these AIM elements is presented on a separate line in the mapping specification. The term "IDENTICAL MAPPING" indicates that both application objects involved in an application assertion map to the same instance of an AIM entity data type. The term "PATH" indicates that the application assertion maps to a collection of related AIM entity instances specified by the entire reference path.

Source: For those AIM elements that are interpreted from any common resource, this is the ISO standard number and part number in which the resource is defined. For those AIM elements that are created for the purpose of this part of ISO 10303, this is "ISO 10303-" followed by the number of this part.

Rules: One or more global rules may be specified that apply to the population of the AIM entity data types specified as the AIM element or in the reference path. For rules that are derived from relationships between application objects, the same rule is referred to by the mapping entries of all the involved AIM elements. A reference to a global rule may be accompanied by a reference to the subclause in which the rule is defined.

Reference path: To describe fully the mapping of an application object, it may be necessary to specify a reference path involving several related AIM elements. Each line in the reference path documents the role of an AIM element relative to the AIM element in the line following it. Two or more such related AIM elements define the interpretation of the integrated resources that satisfies the requirement specified by the application object. For each AIM element that has been created for use within this part of ISO 10303, a reference path to its supertype from an integrated resource is specified. For the expression of reference paths and the relationships between AIM elements the following notational conventions apply:

[] enclosed section constrains multiple AIM elements or sections of the reference path are required to satisfy an information requirement;

() enclosed section constrains multiple AIM elements or sections of the reference path are identified as alternatives within the mapping to satisfy an information requirement;

{} enclosed section constrains the reference path to satisfy an information requirement;

<> enclosed section constrains at one or more required reference path;

|| enclosed section constrains the supertype entity;

-> attribute references the entity or select type given in the following row;

<- entity or select type is referenced by the attribute in the following row;

[i] attribute is an aggregation of which a single member is given in the following row;

[n] attribute is an aggregation of which member n is given in the following row;

=> entity is a supertype of the entity given in the following row;

<= entity is a subtype of the entity given in the following row;

= the string, select, or enumeration type is constrained to a choice or value;

\ the reference path expression continues on the next line;

* used in conjunction with braces to indicate that any number of relationship entity data types may be assembled in a relationship tree structure

// enclosed section is an application of one of the mapping templates defined in clause 5.1.1;

-- the text following is a comment (normally a clause reference).

For the purposes of defining mapping templates, the following abbreviations apply:

ACT action
ASSGN assignment
CART cartesian
CAT category
CD context_dependent
DEF definition
DESC description
DO definable_object
DOC document
EXT external
FUNC function
GEO geometry
INST instance
ORG organization
PD product_definition
PDCD product_definition to characterized_definition
PDR product_definition_representation
PERS person
PROD product
PROP property

REF reference
REL relationship
REP representation
SA shape_aspect
SAR shape_aspect_relationship
SDR shape_definition_representation
SRC source
VAL value
VO versionable_object

5.1.1 Mapping templates

This mapping specification includes mapping templates. A mapping template is a reusable portion of a reference path that defines a commonly used part of the structure of the application interpreted model. A mapping template is similar to a programming language macro. The mapping templates used in this part of ISO 10303 are defined in this subclause. Each mapping template definition has three components, as follows:

— the template signature that specifies the name of the template and may also specify the names and the order of the formal parameters of the template;

— descriptions of the formal parameters of the template, if any;

— the template body that defines the reusable portion of a reference path and may indicate, through the use of the formal parameter names included in the template signature, the points at which the value parameters are supplied in each template application.

Each mapping template is used at least once in the reference paths specified in 5.1.2 to 5.1.16. Each such template application is a reference to the template definition, based on the pattern established by the template signature, and supplies the value parameters that are to substituted for the formal parameters specified in the template definition. The full reference path can be derived by replacing any formal parameters in the template body by the value parameters specified in the template application and then substituting the completed template body for the template application.

The non-blank characters following the first "/" define the name of the mapping template. The name of the mapping template is given in upper case. The name of the template is followed by a list of value parameters, separated by commas, enclosed in parentheses. Parameter values are given in lower case except in the case that the value parameter is a string literal that includes upper case characters.

The following notational conventions apply to the definitions and applications of Mapping templates:

/ marks the beginning and end of a template signature or a template application;

& prefixes the name of a formal parameter within the definition of a template body;

( ) enclose the formal parameters in a template signature or the value parameters in a template application;

, separates formal parameters in a template signature or value parameters in a template application;

'' enclose a string literal that is used as a value parameter in a template application.

Value parameters that are not enclosed by quotes are EXPRESS data type identifiers.

This part of ISO 10303 uses the templates that are specified in the following subclauses.

5.1.1.1 APPROVES

The APPROVES mapping template specifies a reference path constraint in which instances of entities that are the **approval_item**s within **applied_approval_assignment**s and play an ARM roles.

EXAMPLE An illustration of the usage of APPROVES is:

```
approval <-
/APPROVES(product_definition_shape, 'subject')/
product_definition_shape
{product_definition_shape.name = '...'}
...
```

Signature:

/APPROVES(ENTITY, ARM_ROLE)/

Parameter definitions:

ARM_ROLE: the value of the name attribute for object_role.

ENTITY: the entities being grouped that play an ARM role.

Template body:

```
approval_assignment.assigned_approval
approval_assignment
{approval_assignment.role ->
object_role
[object_role.name = &ARM_ROLE]
[object_role.description = '.UNUSED.']} =>
applied_approval_assignment
applied_approval_assignment.items[i] ->
approval_item
approval_item = &ENTITY
```

5.1.1.2 CLASS

The CLASS mapping template specifies a reference path constraint in which an instance that has a classified identification and is a subtype of S_ID.

NOTE The semantics of the template is to assign a class id to an instance.

EXAMPLE A use of this template is: /CLASS(product_definition, 'plate', 'structural part')/.

Signature:

/CLASS(T, ID, S_ID)/

Parameter definitions:

T: type of the instance that is classified.

ID: class id of instance that is classified.

S_ID: class id of super class of ID

Template body:

```
&T
classification_item = &T
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
 classification_role.name = 'class membership'}
classification_assignment
classification_assignment.assigned_classification ->
group
{[group.name = &ID]
 [group <-
  group_relationship.related_group
  group_relationship
  {group_relationship.name = 'specialisation'}
  group_relationship
  group_relationship.relating_group ->
  group
  {group.name = &S_ID} ]}
group =>
class
```

5.1.1.3 CLASS_HELP

The CLASS_HELP mapping template specifies a reference path constraint in which the instance of an entity that gets a **classification_assignment**.

Signature:

/CLASS_HELP(T)/

Parameter definitions:

T: type of the instance that is classified

Template body:

```
{&T
classification_item = &T
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'class membership'}
classification_assignment.assigned_classification ->
group =>
```

5.1.1.4 CLASS_ID

The CLASS_ID mapping template specifies a reference path constraint in which an instance that is classified with a classification identification.

NOTE The semantics of the template is to select an instance having class id ID.

EXAMPLE A use of this template is: /CLASS_ID(product_definition, 'item')/.

Signature:

/CLASS_ID(T, ID)/

Parameter definitions:

T: type of the instance that is classified

ID: class id

Template body:

```
&T
classification_item = &T
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'class membership'}
classification_assignment.assigned_classification ->
group =>
{group.name = &ID}
class
```

5.1.1.5 COMPOUND

The COMPOUND mapping template specifies a reference path constraint in which a path from a **compound_representation_item** to **representation_item**.

Signature:

/COMPOUND(NAME)/

Parameter definitions:

NAME: the value for the **name** attribute for **representation_item**

Template body:

```
compound_representation_item
compound_representation_item.item_element ->
compound_item_definition = list_representation_item
list_representation_item[i] ->
representation_item =>
{representation_item.name = &NAME}
```

5.1.1.6 DAT_TIME_ASSGN

The DAT_TIME_ASSGN mapping template specifies a reference path constraint in which an instance of an entity that is associated with a date and time and plays a role.

Signature:

/DAT_TIME_ASSGN(T, ROLE)/

Parameter definitions:

T: type of the instance that has a date and time assignment.

ROLE: the value of the name attribute for date_time_role.

Template body:

```
&T
{date_and_time_item = &T}
date_and_time_item <-
applied_date_and_time_assignment.items[i]
applied_date_and_time_assignment <=
date_and_time_assignment
{date_and_time_assignment.role ->
date_time_role
[date_time_role.name = &ROLE]
[date_time_role.description = '.UNUSED.']}
date_and_time_assignment.assigned_date_and_time ->
date_and_time
```

5.1.1.7 DESCRIPTION_ASSGN

The DESCRIPTION_ASSGN mapping template specifies a reference path constraint in which the assignment of an **description_attribute** to an instance of an entity that contains a derived description attribute and uses the Basic_attribute_schema of ISO 10303-41.

NOTE The following entity data types use of the DESCRIPTION_ASSGN for population of their description attribute:

— action_request_solution;

— application_context;

— approval_role;

— configuration_design;

— date_role;

— date_time_role;

— context_dependent_shape_representation;

— effectivity;

— external_source;

— organization_role;

— person_and_organization_role;

— person_and_organization;

— person_role;

— property_definition_representation;

— representation;

— time_role.

EXAMPLE An illustration of the usage of DESCRIPTION_ASSGN is:

```
property_definition
description_attribute_select = property_definition
description_attribute_select <-
description_attribute.described_item
description_attribute
description_attribute.attribute_value
```

Signature:

/DESCRIPTION_ASSGN(ENTITY)/

Parameter definitions:

ENTITY:	the entity type from the description_attribute_select to which the description_attribute is assigned.

Template body:

```
&ENTITY
description_attribute_select = &ENTITY
description_attribute_select <-
description_attribute.described_item
description_attribute
description_attribute.attribute_value
```

5.1.1.8 DESCRIPTION_ASSGN_WITH_VAL

The DESCRIPTION_ASSGN_WITH_VAL mapping template specifies the assignment of a **description_attribute** to an instance of an entity that contains a derived name attribute and uses the Basic_attribute_schema of ISO 10303-41, and additionally specifies the value of the name_attribute.

NOTE The following entity data types use of the DESCRIPTION_ASSGN_WITH_VAL for population of their description attribute:

— action_request_solution;

— application_context;

— approval_role;

— configuration_design;

— date_role;

— date_time_role;

— context_dependent_shape_representation;

— effectivity;

— external_source;

— organization_role;

— person_and_organization_role;

— person_and_organization;

— person_role;

— property_definition_representation;

— representation;

— time_role.

EXAMPLE An illustration of the usage of DESCRIPTION_ASSGN_WITH_VAL is:

property_definition
description_attribute_select = property_definition
description_attribute_select <-
description_attribute.described_item
description_attribute
description_attribute.attribute_value
{description_attribute.attribute_value = '...'}

Signature:

/DESCRIPTION_ASSGN_WITH_VALUE(ENTITY, NAME)/

Parameter definitions:

ENTITY: the entity type from the description_attribute_select to which the description_attribute is assigned.

Template body:

&ENTITY
description_attribute_select = &ENTITY
description_attribute_select <-
description_attribute.described_item
description_attribute
description_attribute.attribute_value
{description_attribute.attribute_value = '&NAME_VALUE'}

5.1.1.9 DSC_REP_ITEM

The DSC_REP_ITEM mapping template specifies a **descriptive_representation_item** with an attribute **name** equal to ID.

EXAMPLE A use of this template is: /DSC_REP_ITEM('mirrored')/.

Signature:

/DSC_REP_ITEM(ID)/

Parameter definitions:

ID: the value for the **name** attribute for **representation_item**.

Template body:

```
representation_item =>
{representation_item.name = &ID}
descriptive_representation_item
descriptive_representation_item.description
```


5.1.1.10 DOC_REF

The DOC_REF mapping template specifies a reference path constraint in which an instance of type T references a **document** or **document_usage_constraint** that plays the role ID

Signature:

/DOC_REF(T, ID)/

Parameter definitions:

T: type of the instance that references a **document**

ID: the value for the **name** attribute for **object_role**

Template body:

```
&T
document_reference_item = &T
document_reference_item <-
applied_document_reference.items[iI]
applied_document_reference <=
document_reference
{document_reference.role ->
object_role
[object_role.name = &ID]
[object_role.description = '.UNUSED.']}
document_reference
document_reference.assigned_document ->
(document)
(document <-
document_usage_constraint.source
document_usage_constraint)
```

5.1.1.11 EXT_INST_REF

The EXT_INST_REF mapping template specifies the mapping path for attributes of type external_-instance_reference.

EXAMPLE A use of this template is: /EXT_INST_REF(product_definition_shape, 'ship moulded form schema', 'ship curve')/.

Signature:

/EXT_INST_REF(LT, SN, ET)/

Parameter definitions:

LT: type whose instance represents the external instance in the local instance model
SN: external schema name
ET: external entity type

Template body:

```
[&LT.description = 'external instance reference target']
[&LT = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
ID_ROLE('external instance reference')
external_identification_assignment.source ->
external_source
[EXT_SRC_REL('schema name')
{identifier = &SN}]
[EXT_SRC_REL('entity type')
{identifier = &ET}]]
```

5.1.1.12 EXT_REFERENCE

The EXT_REFERENCE mapping template specifies an external reference to an entity of type T.

Signature:

/EXT_REFERENCE(T)/

Parameter definitions:

T: type of the instance that references an external_reference.

Template body:

```
&T
external_identification_item = &T
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.role ->
identificaton_role
{identification_role.name = 'external reference'}
```

5.1.1.13 EXT_SRC_REL

The EXT_SRC_REL mapping template specifies the relationship of an **external_source** to entity. The concept can be used mainly in two places:

— for relating different components of an external_source to a single instance on the next higher level;

— for mapping of external_instance_reference.

EXAMPLE A use of this template is: /EXT_SRC_REL('postal box')/ or EXT_SRC_REL('entity type')/.

Signature:

/EXT_SRC_REL(DESCR)/

Parameter definitions:

DESCR: the value of the **description** attribute for the **external_source**.

Template body:

```
external_source <-
external_source_relationship.relating_source
external_source_relationship
{external_source_relationship.name = 'composition'}
external_source_relationship.related_source
external_source
{external_source.description = &DESCR}
external_source
external_source.source_id ->
source_item
source_item = identifier
```

5.1.1.14 GEO_REP_ITEM

The GEO_REP_ITEM specifes a **geometric_representation_item** with an attribute **name** equal to ID.

EXAMPLE A use of this template is: /GEO_REP_ITEM(‘cross section geometry‘, bounded_curve)/.

Signature:

/GEO_REP_ITEM(ID, GEO)/

Parameter definitions:

ID: the value for the **name** attribute for **representation_item**.

GEO: type of **geometric_representation_item**.

Template body:

```
representation_item =>
{representation_item.name = &ID}
geometric_representation_item =>
&GEO
```

5.1.1.15 GROUPS

The GROUPS mapping template specifies instances of entities that are the **grouped_item**s within **applied_group_assignment**s and play an ARM role.

EXAMPLE An illustration of the usage of GROUPS is:
group <-
/GROUPS(approval, 'approvals')/
approval
approval.status
...

Signature:

/GROUPS(ENTITY, ARM_ROLE)/

Parameter definitions:

ARM_ROLE: the value of the **name** attribute of **object_role**.

ENTITY: the entities being grouped that play an ARM role.

Template body:

```
group_assignment.assigned_group
group_assignment =>
{group_assignment.role ->
object_role
[object_role.name = &ARM_ROLE]
[object_role.description = '.UNUSED.']}
applied_group_assignment
applied_group_assignment.items[i] ->
group_item
group_item = &ENTITY
```

5.1.1.16 IDENTIFICATION

The IDENTIFICATION mapping template specifies an **identification_assignment** to an element T and is used in other templates.

Signature:

/IDENTIFICATION(T)/

Parameter definitions:

T: type of the instance that references an **identification_assignment**.

Template body:

```
&T
identification_item = &T
```



```
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment <=
identification_assignment
{identification_assignment.role -> identification_role
```

5.1.1.17 HAS_ID_1_ROLE

The HAS_ID_1_ROLE mapping template specifies an instance of type T is given an identifier that plays a single role ROLE.

Signature:

/HAS_ID_1_ROLE(T, ROLE)/

Parameter definitions:

T: type of the instance that references an **identification_assignment**.

ROLE: the value of the **name** attribute for **identification_role**

Template body:

```
/IDENTIFICATION(&T)/
[identification_role.name = &ROLE]
[identification_role.description = '.UNUSED.']}
identification_assignment
identification_assignment.assigned_id
```

5.1.1.18 HAS_ID_2_ROLES

The HAS_ID_2_ROLES mapping template specifies an instance that is given an identifier and plays one of two specified roles.

Signature:

/HAS_ID_2_ROLES(T, ROLE1, ROLE2)/

Parameter definitions:

T: type of the instance that references an **identification_assignment**.

ROLE1: the value of the **name** attribute for **identification_role**

ROLE2: the value of the **name** attribute for **identification_role**

Template body:

```
/IDENTIFICATION(&T)/
[(identification_role.name = &ROLE1)
(identification_role.name = &ROLE2)]
[identification_role.description = '.UNUSED.']}
identification_assignment
identification_assignment.assigned_id
```

5.1.1.19 ID_ROLE

The ID_ROLE mapping template specifies an assignment of an **identification_role**.

EXAMPLE A use of this template is: /ID_ROLE('external reference')/.

Signature:

/ID_ROLE(ROLE)/

Parameter definitions:

ROLE: the value of the **name** attribute for **identification_role**.

Template body:

```
identification_assignment
{identification_assignment.role ->
identificaton_role
[identification_role.name = &ROLE]
[identification_role.description = '.UNUSED.']}
```

5.1.1.20 LINK_TO_GROUP

The LINK_TO_GROUP mapping template specifies a path that links an instance to a **group** instance via an **applied_group_assignment**.

NOTE Used for all ARM types that are subtype of both, item and item_structure.

EXAMPLE A use of this template is: /LINK_TO_GROUP(product_definition)/.

Signature:

/LINK_TO_GROUP(T)/

Parameter definitions:

T: the instance to be grouped.

Template body:

```
group_item = &T <-
applied_group_assignment.items[i]
applied_group_assignment <=
group_assignment
{group_assignment.role ->
object_role
object_role.name = 'equivalence'}
group_assignment
group_assignment.assigned_group ->
group
```

5.1.1.21 NAME_ASSGN

The NAME_ASSGN mapping template specifies the assignment of a **name_attribute** to an instance of an entity that contains a derived name attribute and uses the Basic_attribute_schema of ISO 10303-41.

NOTE The following entity data types use of the NAME_ASSGN for population of their name attribute:

— action_request_solution;

— address;

— configuration_design;

— context_dependent_shape_representation;

— derived_unit;

— effectivity;

— person_and_organization;

— product_definition;

— product_definition_substitute;

— property_definition_representation.

EXAMPLE An illustration of the usage of NAME_ASSGN is:
product_definition
name_attribute_select = product_definition
name_attribute_select <-
name_attribute.named_item
name_attribute
name_attribute.attribute_value

Signature:

/NAME_ASSGN(ENTITY)/

Parameter definitions:

ENTITY: the entity type from the name_attribute_select to which the name_attribute is assigned.

Template body:

```
&ENTITY
name_attribute_select = &ENTITY
name_attribute_select <-
name_attribute.named_item
name_attribute
name_attribute.attribute_value
```

5.1.1.22 NAME_ASSGN_WITH_VAL

The NAME_ASSGN_WITH_VAL mapping template specifies the assignment of a **name_attribute** to an instance of an entity that contains a derived name attribute and uses the Basic_attribute_schema of ISO 10303-41, and additionally specifies the value of the name_attribute.

NOTE The following entity data types use of the NAME_ASSGN_WITH_VAL for population of their name attribute:

— action_request_solution;

— address;

— configuration_design;

— context_dependent_shape_representation;

— derived_unit;

— effectivity;

— person_and_organization;

— product_definition;

— product_definition_substitute;

— property_definition_representation.

EXAMPLE An illustration of the usage of NAME_ASSGN_WITH_VAL is:

```
product_definition
name_attribute_select = product_definition
name_attribute_select <-
name_attribute.named_item
name_attribute
{name_attribute.attribute_value = '...'}
```


Signature:

/NAME_ASSGN_WITH_VAL(ENTITY, NAME_VALUE)/

Parameter definitions:

NAME_VALUE: the value of the attribute_value attribute for name_attribute.

ENTITY: the entity type from the name_attribute_select to which the name_attribute is assigned.

Template body:

```
&ENTITY
name_attribute_select = &ENTITY
name_attribute_select <-
name_attribute.named_item
name_attribute
{name_attribute.attribute_value = '&NAME_VALUE'}
```

5.1.1.23 ORG_ASSGN_PART

The ORG_ASSGN_PART mapping template specifies an instance that is used in an **organization_-assignment**.

Signature:

/ORG_ASSGN_PART(T)/

Parameter definitions:

T: type of the instance that references an **organization_assignment**.

Template body:

```
&T
organization_item = &T
organization_item <-
applied_organization_assignment.items [i]
applied_organization_assignment <=
organization_assignment
```

5.1.1.24 ORG_ASSGN

The ORG_ASSGN mapping template specifies an assignment of an organization to an entity with a role.

EXAMPLE A use of this template is: /ORG_ASSGN(product_definition, 'document author')/.

Signature:

/ORG_ASSGN(T, ROLE)/

Parameter definitions:

T: the entity assigned to **organization**

ROLE: the value of the **name** attribute for **organization_role**.

Template body:

```
&T
organization_item = &T
organization_item <-
applied_organization_assignment.items[i]
applied_organization_assignment <=
organization_assignment
{organization_assignment.role ->
organization_role
[organization_role.name = &ROLE]
[organization_role.descripion = '.UNUSED.']}
organization_assignment
organization_assignment.assigned_organization ->
organization
```

5.1.1.25 PD_HAS_FUNC

The PD_HAS_FUNC mapping template specifies the instance of **product_definition** that has a function defined by the **name** attribute for **descriptive_representation_item**.

Signature:

/PD_HAS_FUNC(FUNC)/

Parameter definitions:

FUNC: the name of the function

Template body:

```
PDCD <-
propert_definition.definition
property_definition
property_definition =
represented_definition <-
property_definition_representation.definition
property_definition_representation
property_definition_representation.used_representation ->
representation
representation.items [i] ->
representation_item =>
descriptive_representation_item
{[descriptive_representation_item.name = 'the function']
[descriptive_representation_item.description = &FUNC]}
```


5.1.1.26 PDCD

The PDCD mapping template specifies the reference path from **product_definition** to **characterised_-definition**.

Signature:

/PDCD/

Parameter definitions:

none

Template body:

```
product_definition =
characterised_product_definition =
characterised_definition
```

5.1.1.27 PDCD_CLASS_ID

The PDCD_CLASS_ID mapping template specifies the reference path from **product_definition**, which has an instance of a class id, to **characterised_definition**.

Signature:

/PDCD_CLASS_ID(ID)/

Parameter definitions:

ID: class id

Template body:

```
product_definition =
{CLASS_ID(product_definition, &ID)}
characterised_product_definition =
characterised_definition
```

5.1.1.28 PDR

The PDR mapping template specifies the reference path for **property_definition_representation**. The **name** and **description** attributes are specified as '.UNUSED.'.

Signature:

/PDR/

Parameter definitions:

none

Template body:

```
property_definition_representation.definition
property_definition_representation
{[NAME_ASSGN_WITH_VAL(property_definition_representation, '.UNUSED.')]
{[DESCRIPTION_ASSGN_WITH_VAL(property_definition_representation,
'.UNUSED.')]}
property_definition_representation.used_representation
```

5.1.1.29 PDR_NAME

The PDR_NAME mapping template specifies the reference path for **property_definition_representation**. The **description** attributes are defined as '.UNUSED.', and the **name** attribute is specified.

Signature:

/PDR_NAME(NAME)/

Parameter definitions:

NAME: the value for the **name** attribute for **property_definition_representation**

Template body:

```
property_definition_representation.definition
property_definition_representation
{[NAME_ASSGN_WITH_VAL(property_definition_representation, '&NAME')]
{[DESCRIPTION_ASSGN_WITH_VAL(property_definition_representation,
'.UNUSED.')]}
property_definition_representation.used_representation
```

5.1.1.30 PERS_ASSGN

The PERS_ASSGN mapping template specifies the assignment of a person to an entity with a role.

EXAMPLE A use of this template is: /PERS_ASSGN(document, 'document author')/.

Signature:

/PERS_ASSGN(T, ROLE)/

Parameter definitions:

T: the entity assigned to **person**

ROLE: the value of the **name** attribute for **person_role**.

Template body:

```
&T
person_item = &T
person_item <-
applied_person_assignment.items[i]
applied_person_assignment <=
person_assignment
{person_assignment.role ->
```



```
person_role
[person_role.name = &ROLE]
[person_role.description = '.UNUSED.']}
person_assignment
person_assignment.assigned_person ->
person
```

5.1.1.31 PERS_ORG_ASSGN

The PERS_ORG_ASSGN mapping template specifies an assignment of a person inside an organization to an entity with a role.

EXAMPLE A use of this template is: /PERS_ORG_ASSGN(document, 'document author ')/.

Signature:

/PERS_ORG_ASSGN(T, ROLE)/

Parameter definitions:

T: the entity assigned to **person_and_organization**

ROLE: the value of the **name** attribute for **person_and_organization_role**.

Template body:

```
&T
Person_and_organization_item = &T
person_and_organization_item <-
applied_person_and_organization_assignment.items[i]
applied_person_and_organization_assignment <=
person_and_organization_assignment
{person_and_organization_assignment.role ->
person_and_organization_role
[person_and_organization_role.name = &ROLE]
[person_and_organization_role.description = '.UNUSED.']}
person_and_organization_assignment
person_and_organization_assignment.assigned_person_and_organization
person_and_organization
```

5.1.1.32 PROD_CAT_NAME

The PROD_CAT_NAME mapping template specifies a **product_category** that has an attribute **name** with a specific value, and is of type **product_related_product_category**.

Signature:

/PROD_CAT_NAME(NAME)/

Parameter definitions:

NAME: value given the **name** attribute for **product_category**.

Template body:

```
product_related_product_category <=
product_related_product_category <=
product_category
{product_category.name = &NAME}
```

5.1.1.33 PROD_DEF_PRODUCT

The PROD_DEF_PRODUCT mapping template specifies a path that links a **product_definition** with a **product**.

Signature:

/PROD_DEF_PRODUCT/

Parameter definitions:

none

Template body:

```
product_definition
product_definition.formation ->
product_definition_formation
product_definition_formation.of_product ->
product
```

5.1.1.34 PROD_DEF_PROP_DEF

The PROD_DEF_PROP_DEF mapping template specifies a path that links a **product_definitio**n with a **property_definition**.

Signature:

/PROD_DEF_PROP_DEF/

Parameter definitions:

none

Template body:

```
/PROD_DEF_PROP_DEF_HELP/
property_definition
```

5.1.1.35 PROD_DEF_PROP_DEF_HELP

The PROD_DEF_PROP_DEF_HELP mapping template specifies a path that links a **product_definition** with a **property_definition.definition**. PROD_DEF_PROP_DEF_HELP is intended to be used by other mapping templates.

Signature:

/PROD_DEF_PROP_DEF_HELP/

Parameter definitions:

none

Template body:

```
product_definition
characterized_product_definition = product_definition
characterized_product_definition
characterized_definition = characterized_product_definition
characterized_definition <-
property_definition.definition
```

5.1.1.36 PROD_DEF_TO_DESC_REP_ITEM

The PROD_DEF_TO_DESC_REP_ITEM mapping template specifies a path that links a **product_definition** with a **descriptive_represenation_item** that has an attribute **name** with a specific value. This link is through a **property_definition_representaion** that has an attribute **name** with a specific value. PROD_DEF_DESC_-REP_ITEM is intended to be used by other mapping templates.

Signature:

/PROD_DEF_TO_DESC_REP_ITEM(ID1, ID2)/

Parameter definitions:

ID1: the value of the **name** attribute for **property_definition**

ID2: the value of the **name** attribute for **property_definition_representation**.

Template body:

```
/PROD_DEF_PROP_DEF_HELP/
PROP_DEF_TO_DESC_REP_ITEM(&ID1, &ID2)
```

5.1.1.37 PROD_DEF_TO_REP

The PROD_DEF_TO_REP mapping template specifies a path that links a **product_definition** with a **property_definition_representation** with an attribute **name** that has a specific value, and through the attribute **used_representation**. PROD_DEF_REP is intended to be used by other mapping templates.

Signature:

/PROD_DEF_TO_REP(ID)/

Parameter definitions:

ID: the value of the **name** attribute for **property_definition_representation**

Template body:

```
/PROD_DEF_PROP_DEF_HELP/
PROP_DEF_REP_HELP(&ID)
```

5.1.1.38 PROD_DEF_TO_SPECIAL_VAL_REP_ITEM

The PROD_DEF_TO_SPECIAL_VAL_REP_ITEM mapping template specifies a path that links a **product_-definition** with an attribute **name** that has a specific value, to a **value_representation_item** with a **derived_-unit** that has a **name** attribute with a specific value, and the **measure_value** that is a **context_dependent_-measure**. This link is through a **property_definition_representation** with an attribute **name** that has a specific value. PROD_DEF_TO_SPECIAL_VAL_REP_ITEM is intended to be used by other mapping templates.

Signature:

/PROD_DEF_TO_SPECIAL_VAL_REP_ITEM(ID1, ID2, DER_UNIT_NAME)/

Parameter definitions:

ID1: the value of the **name** attribute for **property_definition**.

ID2: the value of the **name** attribute for **property_definition_representation**.

DER_UNIT_NAME: the value of the **name** attribute for **derived_unit**.

Template body:

```
/PROD_DEF_PROP_DEF_HELP/
PROP_DEF_TO_SPECIAL_VAL_REP_ITEM(&ID1, &ID2, &DER_UNIT_NAME)
```

5.1.1.39 PROD_DEF_TO_UNITS

The PROD_DEF_TO_UNITS mapping template specifies a path that links a **product_definition** with a **global_unit_assigned_context**. This link is through a **property_definition_representation** with an attribute **name** that has a specific value. PROD_DEF_TO_UNITS is intended to be used by other mapping templates.

Signature:

/PROD_DEF_TO_UNITS(ID)/

Parameter definitions:

ID: the value of the **name** attribute for **property_definition_representation**.

Template body:

```
/PROD_DEF_PROP_DEF_HELP/
PROP_DEF_TO_UNITS(&ID)
```

5.1.1.40 PROD_DEF_TO_VAL_REP_ITEM

The PROD_DEF_TO_VAL_REP_ITEM mapping template specifies a path that links a **product_definition** with an attribute **name** that has a specific value, to a **value_representation_item** with a **measure_value** and the type of **measure_value** is specified. This link is through a **property_definition_representation** with an attribute **name** that has a specific value. PROD_DEF_TO_VAL_REP_ITEM is intended to be used by other mapping templates.

Signature:

/PROD_DEF_TO_VAL_REP_ITEM(ID1, ID2, MEAS)/

Parameter definitions:

ID1: the value of the **name** attribute for **property_definition_representation**.

ID2: the value of the **name** attribute for **value_representation_item**.

MEAS: type of **measure_value** that is specifed.

Template body:

```
/PROD_DEF_PROP_DEF_HELP/
PROP_DEF_TO_VAL_REP_ITEM(&ID1, &ID2, &MEAS)
```

5.1.1.41 PROP_DEF_REP_HELP

The PROP_DEF_REP_HELP mapping template specifies a path that links a **property_definition** with a **property_definition_representation** with an attribute **name** that has a specific value, and through the attribute **used_representation**. PROP_DEF_REP_HELP is intended to be used by other mapping templates.

Signature:

/PROP_DEF_REP_HELP(ID)/

Parameter definitions:

ID: the value of the **name** attribute for **property_definition_representation**

Template body:

```
property_definition
represented_definition = property_definition
represented_definition <-
property_definition_representation.definition
property_definition_representation
{[/PDR_NAME(&ID)/]}
property_definition_representation.used_representation ->
```

5.1.1.42 PROP_DEF_TO_DESC_REP_ITEM

The PROP_DEF_TO_DESC_REP_ITEM mapping template specifies a path that links a **property_definition** with a **descriptive_represenation_item** that has an attribute **name** with a specific value. This link is through a **property_definition_representaion** that has an attribute **name** with a specific value.

Signature:

/PROP_DEF_TO_DESC_REP_ITEM(ID1, ID2)/

Parameter definitions:

ID1: the value of the **name** attribute for **property_definition**

ID2: the value of the **name** attribute for **property_definition_representation**.

Template body:

```
/PROP_DEF_REP_HELP(&ID1)/
/REP_ITEM(&ID2)/
descriptive_representation_item
descriptive_representation_item.description
```

5.1.1.43 PROP_DEF_TO_REP

The PROP_DEF_TO_REP mapping template specifies a path from **property_definition** to **representation**.

Signature:

/PROP_DEF_TO_REP/

Parameter definitions:

none

Template body:

```
property_definition
represented_definition = property_definition
represented_definition <-
property_definition_representation.definition
property_definition_representation
property_definition_representation.used_representation ->
representation
```

5.1.1.44 PROP_DEF_TO_SPECIAL_VAL_REP_ITEM

The PROP_DEF_TO_SPECIAL_VAL_REP_ITEM mapping template specifies a path that links a **property_definition** with an attribute **name** that has a specific value, to a **value_representation_item** with a **derived_unit** that has a **name** attribute with a specific value, and the **measure_value** that is a **context_dependent_measure**. This link is through a **property_definition_representation** with an attribute **name** that has a specific value.

Signature:

/PROP_DEF_TO_SPECIAL_VAL_REP_ITEM(ID1, ID2, DER_UNIT_NAME)/

Parameter definitions:

ID1: the value of the **name** attribute for **property_definition**.

ID2: the value of the **name** attribute for **property_definition_representation**.

DER_UNIT_NAME: the value of the **name** attribute for **derived_unit**.

Template body:

```
/PROP_DEF_REP_HELP(&ID1)/
REP_TO_SPECIAL_VAL_REP_ITEM(&ID2, &DER_UNIT_NAME)
```

5.1.1.45 PROP_DEF_TO_UNITS

The PROP_DEF_TO_UNITS mapping template specifies a path that links a **property_definition** with a **global_unit_assigned_context** and with a **property_definition_representaion** attribute **name** that specifies a value.

Signature:

/PROP_DEF_TO_UNITS(ID)/

Parameter definitions:

ID: the value of the **name** attribute for **property_definition_representation**.

Template body:

```
/PROP_DEF_REP_HELP(&ID)/
representation
representation.context_of_items ->
representation_context =>
global_unit_assigned_context
global_unit_assigned_context.units
```

5.1.1.46 PROP_DEF_TO_VAL_REP_ITEM

The PROP_DEF_TO_VAL_REP_ITEM mapping template specifies a path that links a **property_definition** with a **value_representation_item** with an attribute **name** that specifies a value and measure_value, where the **property_definition_representation** attribute name specifies a value.

Signature:

/PROP_DEF_TO_VAL_REP_ITEM(ID1, ID2, MEAS)/

Parameter definitions:

ID1: the value of the **name** attribute for **property_definition_representation**.

ID2: the value of the **name** attribute for **value_representation_item**.

MEAS: type of **measure_value** that is specifed.

Template body:

```
/PROP_DEF_REP_HELP(&ID1)/
/REP_ITEM(&ID2)/
value_representation_item
value_representation_item.value_component -> measure_value
{measure_value = &MEAS}
```

5.1.1.47 PROP_TO_PROD_DEF

The PROP_TO_PROD_DEF mapping template specifies a path from **property_definition** to **product_-definition**.

Signature:

/PROP_TO_PROD_DEF/

Parameter definitions:

none

Template body:

```
property_definition
property_defintion.definition -> characterized_definition
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
```

5.1.1.48 RELATE_ACT_2_VO

The RELATE_ACT_2_VO assigns an **action** with a role to any versionable objects.

EXAMPLE An illustration of RELATE_ACT_2_VO is:

action <-
action_assignment.assigned_action
action_assignment
{action_assignment.role = 'base'} =>
(RELATE_ACT_2_VO(representation))
(RELATE_ACT_2_VO(product_definition))
...

Signature:

/RELATE_ACT_2_VO(T)/

Parameter definitions:

T: type of the instance that is classified

Template body:

```
applied_action_assignment
applied_action_assignment.items[i] ->
(action_item = &T
{CLASS_ID(T, 'versionable_object')})
```

5.1.1.49 RELATE_GROUP_2_VO

The RELATE_GROUP_2_VO mapping template specifies the collection of AIM objects, that are the image of a versionable object into a group with the ARM role.

EXAMPLE An illustration of RELATE_GROUP_2_VO is:
group <-
(RELATE_GROUP_2_VO(representation, 'members'))
(RELATE_GROUP_2_VO(product_definition, 'members'))
...

Signature:

/RELATE_GROUP_2_VO(T, ROLE)/

Parameter definitions:

T: type of object specified as a definable object

ROLE: the name of the ARM role

Template body:

```
/GROUPS(&T, &ROLE)/
{CLASS_ID('versionable object')}
```

5.1.1.50 RELATE_GROUP_2_DO

The RELATE_GROUP_2_DO mapping template specifies the collection of AIM objects, which are the image of a definable object into a group with the ARM role.

EXAMPLE An illustration of RELATE_GROUP_2_DO is:
group <-
(/RELATE_GROUP_2_DO(product, 'context of revision')/)
(/RELATE_GROUP_2_DO(product_definition, 'context of revision')/)
...

Signature:

/RELATE_GROUP_2_DO(T, ROLE)/

Parameter definitions:

T: type of object specified as a definable object

ROLE: the name of the ARM role

Template body:

```
/GROUPS(&T, &ROLE)/
{CLASS_ID('definable object')}
```

5.1.1.51 RELATE_ID_2_VO

The RELATE_ID_2_VO mapping template specifies the relationship of an AIM object, that is the image of a versionable object via an **identification_assignment**. This mapping template can be understood as the inverse mapping template to VERSION_ID(T).

EXAMPLE An illustration of RELATE_ID_2_VO is:

identification_assignment_relationship
identification_assignment_relationship.related_assignment ->
(RELATE_ID_2_VO(representation))
(RELATE_ID_2_VO(product_definition))
...

Signature:

/RELATE_ID_2_VO(T)/

Parameter definitions:

T: type if AIM object that has an **identification_assignment**

Template body:

```
identification_assignment
{identification_assignment.role ->
identification_role
[identification_role.name = 'version id']
[identification_role.description = '.UNUSED.']} =>
applied_identification_assignment
applied_identification_assignment.items[i] ->
identification_item = &T
{CLASS_ID(&T, 'versionable object')}
```

5.1.1.52 REP_ITEM

The REP_ITEM mapping template specifies a path that links a **representation** with a **representation_item**. REP_ITEM is intended to be used by other mapping templates.

Signature:

/REP_ITEM(ID)/

Parameter definitions:

ID: value given the **name** attribute for **representation_item**.

Template body:

```
representation
representation.items [i] ->
representation_item =>
{representation_item.name = &ID}
```

5.1.1.53 REP_TO_SPECIAL_VAL_REP_ITEM

The REP_TO_SPECIAL_VAL_REP_ITEM mapping template specifies a path that links a **representation** with a **value_representation_item** with a specific **derived_unit** that has an attribute **name** with a specific value. The **representation_item** attribute **name** specifies the value of the item. The **measure_value** is **context_-dependent_measure**.

Signature:

/REP_TO_SPECIAL_VAL_REP_ITEM(ID, DER_UNIT_NAME)/

Parameter definitions:

ID: the value of the **name** attribute for **representation_item**

DER_UNIN_NAME: the value of the **name** attributre for **derived_unit**.

Template body:

```
representation
{representation.context_of_items ->
representation_context =>
global_unit_assigned_context
global_unit_assigned_context.units[i] ->
unit =
derived_unit
derived_unit.name = &DER_UNIT_NAME}
representation.items [i] ->
representation_item =>
{representation_item.name = &ID}
value_representation_item
value_representation_item.value_component -> measure_value
{measure_value = context_dependent_measure }
```

5.1.1.54 REP_TO_VAL_REP_ITEM

The REP_TO_VAL_REP_ITEM mapping template specifies a path that links a **representation** with a **value_-representation_item**. The **representation_item** attribute **name** is given an identification value, and the type of **measure_value** is specified.

Signature:

/REP_TO_VAL_REP_ITEM(ID, MEAS)/

Parameter definitions:

ID: value given the **name** attribute for **representation_item**.

MEAS: type of **measure_value** that is specifed.

Template body:

```
/REP_ITEM(&ID)/
value_representation_item
value_representation_item.value_component -> measure_value
{measure_value = &MEAS}
```

5.1.1.55 ROLE_ASSGN

The ROLE_ASSGN mapping template specifies the assignment of a roll_association to an instance of an entity that contains a derived role attribute and uses the Basic_attribute_schema of ISO 10303-41.

NOTE The following entity data types use of the ROLE_ASSGN for population of their role attribute:

— action_assignment;

— action_request_assignment;

— approval_assignment;

— approval_date_time;

— certification_assignment;

— contract_assignment;

— document_reference;

— effectivity_assignment;

— external_referent_assignment;

— group_assignment;

— name_assignment;

— security_classification_assignment.

EXAMPLE An illustration of the usage of ROLE_ASSGN is:

product_definition
name_attribute_select = product_definition
name_attribute_select <-
name_attribute.named_item
name_attribute
name_attribute.attribute_value

Signature:

/ROLE_ASSGN(ENTITY)/

Parameter definitions:

ENTITY: the entity type from the role_select to which the role_association is assigned.

Template body:

```
&ENTITY
role_select = &ENTITY
role_select <-
role_association.item_with_role
role_association
role_association.role ->
object_role
```


5.1.1.56 ROOT_CLASS

The ROOT_CLASS mapping template specifies an instance that has a classification id which is a root class.

NOTE The semantics of the template is to assign a class id to an instance.

EXAMPLE A use of this template is: /ROOT_CLASS(product_definition, 'definable object')/.

Signature:

/ROOT_CLASS(T, ID)/

Parameter definitions:

T: type of the instance that is classified

ID: root class id

Template body:

```
&T
classification_item = &T
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'class membership'}
classification_assignment.assigned_classification ->
group =>
{group.name = &ID}
class
```

5.1.1.57 SAME_GROUP

The SAME_GROUP mapping template specifies two instances that are both members in an instance of group with the same class id.

Signature:

/SAME_GROUP(T1, T2, ID)/

Parameter definitions:

T1: type of the instance that is grouped

T2: type of the instance that is grouped

ID: class id

Template body:

```
[&T1 = group_item]
[&T2 = group_item]
group_item <-
applied_group_assignment.items[i]
applied_group_assignment <=
group_assignment
group_assignment.assigned_group ->
group
{CLASS_ID(group, &ID)}
```

5.1.1.58 SDR_NAME

The SDR_NAME mapping template specifies a reference path constraint in which the name attribute of **shape_definition_representation.name** is assigned the value **NAME**.

Signature:

/SDR_NAME(NAME)/

Parameter definitions:

NAME: the value for the **name** attribute for **shape_definition_representation**

Template body:

```
property_definition_representation.definition
property_definition_representation
{property_definition_representation=>
shape_definition_representation}
{NAME_ASSGN_WITH_VAL(property_definition_representation, '&NAME')}
property_definition_representation.used_representation
```

5.1.1.59 SUBTYPE

The SUBTYPE mapping template specifies a reference to the mapping of a subtype of the current application object. Several such references may be included for one supertype application object.

NOTE This template definition only consists of a template signature, there is no matching template body. The template is included to ease the automatic processing of the mapping specification.

Signature:

/SUBTYPE(application_object)/

Parameter definition:

application_object: the application object that is a subtype of the current supertype application object and that has the entire or a part of the mapping specification of this supertype.

5.1.1.60 SUPERTYPE

The SUPERTYPE mapping template specifies a reference to the mapping of a supertype of the current application object. Several such references may be included for the subtype application object.

NOTE This template only consists of a signature, there is no matching body. The template is included to ease the automatic processing of the mapping specification.

Signature:

/SUPERTYPE(application_object)/

Parameter definition:

application_object: the application object that is a supertype of the current subtype application object and that has the entire or a part of the mapping specification of this subtype.

5.1.1.61 VAL_REP_ITEM

The VAL_REP_ITEM mapping template specifies the **value_representation_item** with **name** = of type measure.

EXAMPLE A use of this template is: VAL_REP_ITEM('nominal mass per length', mass_measure).
Signature:

/VAL_REP_ITEM(ID, MEAS)/

Parameter definitions:

ID: class id

MEAS: type of **measure_value** that is specifed.

Template body:

```
representation_item =>&ID}
value_representation_item
value_representation_item.value_component
{value_representation_item.value_component = &MEAS}
```

5.1.1.62 VAL_REP_ITEM_CD

The VAL_REP_ITEM_CD mapping template specifies a **value_representation_item** with **name** of type **context_dependent_measure** where the context is provided by an mapping rule that relates it to the **global_-unit_assigned_context** of the **representation**, and a **derived_unit** with **name** = to a set of units)

EXAMPLE A use of this template is: /VAL_REP_ITEM('shear force', 'force unit')/.

Signature:

/VAL_REP_ITEM_CD(ID, DER_UNIT_NAME)/

Parameter definitions:

ID: the value of the **name** attribute for **representation_item**

DER_UNIT_NAME: the value of the **name** attributre for **derived_unit**.

Template body:

```
representation_item
{[representation_item.name = &ID]
[representation_item <-
representation.items[i]
representation
representation.context_of_items ->
representation_context =>
global_unit_assigned_context
global_unit_assigned_context.units[i] ->
unit =
derived_unit
derived_unit.name = &DER_UNIT_NAME]}
representation_item =>
value_representation_item
value_representation_item.value_component
{value_representation_item.value_component = context_dependent_measure}
```

5.1.1.63 VERSION_ID

The VERSION_ID mapping template specifies an instance that is given a version identifier that plays the role version id.

Signature:

/VERSION_ID(T)/

Parameter definitions:

T: type of instance that is given a version identifier.

Template body:

```
/IDENTIFICATION(&T)/
[identification_role.name = 'version id']
[identification_role.description = '.UNUSED.']}
identification_assignment
identification_assignment.assigned_id
```

5.1.2 class_approvals UoF

5.1.2.1 CORRUGATED_PART_APPROVAL

AIM element: group
Source: ISO 10303-41
Reference path: {[/CLASS(group, 'corrugated part approval', 'structural class approval')/]
[/CLASS(group, 'structural class approval', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]}

5.1.2.1.1 status

AIM element: /SUPERTYPE(structural_class_approval)/ (See 5.1.2.13.1)

5.1.2.1.2 corrugated_part_approval to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.1.3 corrugated_part_approval to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.1.4 corrugated_part_approval to corrugated_part_design_definition (as subject)

AIM element: PATH
Rules: 5.2.4.7, 5.2.4.13, 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/APPROVES(product_definition_shape, 'subject')/
{/CLASS_ID(product_definition_shape, 'corrugated part design definition')/}

5.1.2.2 CORRUGATED_PART_REJECTION

AIM element: group
Source: ISO 10303-41
Reference path: {[/CLASS(group, 'corrugated part rejection', 'structural class rejection')/]
[/CLASS(group, 'structural class rejection', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]}

5.1.2.2.1 status

AIM element: /SUPERTYPE(structural_class_rejection)/ (See 5.1.2.16.1)

5.1.2.2.2 corrugated_part_rejection to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.2.3 corrugated_part_rejection to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.2.4 corrugated_part_rejection to definition (as proposed_alternative)

AIM element: PATH
Rules: 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
(APPROVES(product_definition_shape, 'proposed alternative')/
{/CLASS_ID(product_definition_shape, 'corrugated part design definition')/}

5.1.2.2.5 corrugated_part_rejection to corrugated_part_design_definition (as subject)

AIM element: PATH
Rules: 5.2.4.7, 5.2.4.13, 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/APPROVES(product_definition_shape, 'subject')/
{/CLASS_ID(product_definition_shape, 'corrugated part design definition')/}

5.1.2.3 HULL_CROSS_SECTION_APPROVAL

AIM element: [group]
[representation]
Source: ISO 10303-41, ISO 10303-43
Rules: 5.2.4.215
Reference path: {representation
{representation.name = 'hull cross section result'}
/LINK_TO_GROUP(representation)/
[/CLASS(group, 'hull cross section approval', 'hull cross section result')/]
[/CLASS(group, 'hull cross section result', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]}

5.1.2.3.1 actual_first_moment_horizontal

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.2 actual_first_moment_vertical

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.3 actual_inertia_cross

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.4 actual_inertia_horizontal

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.5 actual_inertia_torsion

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.6 actual_inertia_vertical

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.7 actual_neutral_axis_position_horizontal

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.8 actual_neutral_axis_position_vertical

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.9 actual_section_area

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.10 actual_section_modulus_bottom

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.11 actual_section_modulus_coaming

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.12 actual_section_modulus_deck

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.13 actual_stress_bottom

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.14 actual_stress_coaming

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.15 actual_stress_deck

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.16 actual_twist_centre_horizontal

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.17 actual_twist_centre_vertical

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.18 direction_largest_inertia

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.19 direction_least_inertia

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.20 max_moment

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.21 max_shear_force

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.22 meter_weight

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.23 min_moment

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.24 min_shear_force

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.25 required_section_modulus_bottom

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.26 required_section_modulus_coaming

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.27 required_section_modulus_deck

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.28 required_stress_bottom

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.29 required_stress_coaming

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.30 required_stress_deck

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5)

5.1.2.3.31 status

AIM element: /SUPERTYPE(structural_class_approval)/ (See 5.1.2.13.1)

5.1.2.3.32 hull_cross_section_approval to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.3.33 hull_cross_section_approval to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.3.34 hull_cross_section_approval to hull_cross_section_design_-definition (as subject)

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.34)

5.1.2.4 HULL_CROSS_SECTION_REJECTION

AIM element: [group]
[representation]
Source: ISO 10303-41, ISO 10303-43
Rules: 5.2.4.215
Reference path: {representation
{representation.name = 'hull cross section result'}
/LINK_TO_GROUP(representation)/
[/CLASS(group, 'hull cross section rejection', 'hull cross section result')/]
[/CLASS(group, 'hull cross section result', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]}

5.1.2.4.1 actual_first_moment_horizontal

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.1)

5.1.2.4.2 actual_first_moment_vertical

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.2)

5.1.2.4.3 actual_inertia_cross

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.3)

5.1.2.4.4 actual_inertia_horizontal

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.4)

5.1.2.4.5 actual_inertia_torsion

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.5)

5.1.2.4.6 actual_inertia_vertical

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.6)

5.1.2.4.7 actual_neutral_axis_position_horizontal

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.7)

5.1.2.4.8 actual_neutral_axis_position_vertical

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.8)

5.1.2.4.9 actual_section_area

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.9)

5.1.2.4.10 actual_section_modulus_bottom

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.10)

5.1.2.4.11 actual_section_modulus_coaming

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.11)

5.1.2.4.12 actual_section_modulus_deck

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.12)

5.1.2.4.13 actual_stress_bottom

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.13)

5.1.2.4.14 actual_stress_coaming

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.14)

5.1.2.4.15 actual_stress_deck

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.15)

5.1.2.4.16 actual_twist_centre_horizontal

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.16)

5.1.2.4.17 actual_twist_centre_vertical

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.17)

5.1.2.4.18 direction_largest_inertia

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.18)

5.1.2.4.19 direction_least_inertia

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.19)

5.1.2.4.20 max_moment

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.20)

5.1.2.4.21 max_shear_force

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.21)

5.1.2.4.22 meter_weight

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.22)

5.1.2.4.23 min_moment

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.23)

5.1.2.4.24 min_shear_force

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.24)

5.1.2.4.25 required_section_modulus_bottom

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.25)

5.1.2.4.26 required_section_modulus_coaming

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.26)

5.1.2.4.27 required_section_modulus_deck

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.27)

5.1.2.4.28 required_stress_bottom

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.28)

5.1.2.4.29 required_stress_coaming

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.29)

5.1.2.4.30 required_stress_deck

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.30)

5.1.2.4.31 status

AIM element: /SUPERTYPE(structural_class_rejection)/ (See 5.1.2.16.1)

5.1.2.4.32 hull_cross_section_rejection to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.4.33 hull_cross_section_rejection to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.4.34 hull_cross_section_rejection to definition (as proposed_-alternative)

AIM element: PATH
Rules: 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/APPROVES(product_definition_shape, 'proposed alternative')/
{/CLASS_ID(product_definition_shape, 'hull cross section design definition')/}

5.1.2.4.35 hull_cross_section_rejection to hull_cross_section_design_-definition (as subject)

AIM element: /SUPERTYPE(hull_cross_section_result)/ (See 5.1.2.5.34)

5.1.2.5 HULL_CROSS_SECTION_RESULT

AIM element: [group]
[representation]
Source: ISO 10303-41, ISO 10303-43
Rules: 5.2.4.215, 5.2.4.65
Reference path: {representation
{representation.name = 'hull cross section result'}
/LINK_TO_GROUP(representation)/
[/CLASS(group, 'hull cross section result', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]}

5.1.2.5.1 actual_first_moment_horizontal

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual first moment horizontal', 'moment unit')/

5.1.2.5.2 actual_first_moment_vertical

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual first moment vertical', 'moment unit')/

5.1.2.5.3 actual_inertia_cross

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual inertia cross', 'moment unit')/

5.1.2.5.4 actual_inertia_horizontal

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual inertia horizontal', 'inertia moment unit')/

5.1.2.5.5 actual_inertia_torsion

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual inertia torsion', 'inertia moment unit')/

5.1.2.5.6 actual_inertia_vertical

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual inertia vertical', 'inertia moment unit')/

5.1.2.5.7 actual_neutral_axis_position_horizontal

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('actual neutral axis position horizontal', length_measure)/

5.1.2.5.8 actual_neutral_axis_position_vertical

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('actual neutral axis position vertical', length_measure)/

5.1.2.5.9 actual_section_area

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('actual section area', area_measure)/

5.1.2.5.10 actual_section_modulus_bottom

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual section modulus bottom', 'section modulus unit')/

5.1.2.5.11 actual_section_modulus_coaming

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual section modulus coaming', 'section modulus unit')/

5.1.2.5.12 actual_section_modulus_deck

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual section modulus deck', 'section modulus unit')/

5.1.2.5.13 actual_stress_bottom

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual stress bottom', 'stress unit')/

5.1.2.5.14 actual_stress_coaming

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual stress coaming', 'stress unit')/

5.1.2.5.15 actual_stress_deck

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('actual stress deck', 'stress unit')/

5.1.2.5.16 actual_twist_centre_horizontal

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('actual twist centre horizontal', length_measure)/

5.1.2.5.17 actual_twist_centre_vertical

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('actual twist centre vertical', length_measure)/

5.1.2.5.18 direction_largest_inertia

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('direction largest inertia', plane_angle_measure)/

5.1.2.5.19 direction_least_inertia

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('direction least inertia', plane_angle_measure)/

5.1.2.5.20 max_moment

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('max moment', 'moment unit')/

5.1.2.5.21 max_shear_force

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('max shear force', 'force unit')/

5.1.2.5.22 meter_weight

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('meter weight', mass_measure)/

5.1.2.5.23 min_moment

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('min moment', 'moment unit')/

5.1.2.5.24 min_shear_force

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('min shear force', 'force unit')/

5.1.2.5.25 required_section_modulus_bottom

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
VAL_REP_ITEM_CD('required section modulus bottom', 'section modulus unit')/

5.1.2.5.26 required_section_modulus_coaming

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('required section modulus coaming', 'section modulus unit')/

5.1.2.5.27 required_section_modulus_deck

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('required section modulus deck', 'section modulus unit')/

5.1.2.5.28 required_stress_bottom

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('required stress bottom', 'stress unit')/

5.1.2.5.29 required_stress_coaming

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('required stress coaming', 'stress unit')/

5.1.2.5.30 required_stress_deck

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM_CD('required stress deck', 'stress unit')/

5.1.2.5.31 status

#1: hull_cross_section_result is a hull_cross_section_approval
#2: hull_cross_section_result is a hull_cross_section_rejection

AIM element: #1: /SUBTYPE(hull_cross_section_approval)/ (See 5.1.2.4.32)
#2: /SUBTYPE(hull_cross_section_rejection)/ (See 5.1.2.3.32)

5.1.2.5.32 hull_cross_section_result to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.5.33 hull_cross_section_result to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.5.34 hull_cross_section_result to hull_cross_section_design_-definition (as subject)

AIM element: PATH
Source: ISO 10303-43
Rules: 5.2.4.7, 5.2.4.13, 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/APPROVES(product_definition_shape, 'subject')/
{/CLASS_ID(product_definition_shape, 'hull cross section design definition')/}

5.1.2.6 PLATE_APPROVAL

AIM element: group
Source: ISO 10303-41
Reference path: {[/CLASS(group, 'plate approval', 'structural class approval')/]
[/CLASS(group, 'structural class approval', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]}

5.1.2.6.1 status

AIM element: /SUPERTYPE(structural_class_approval)/ (See 5.1.2.13.1)

5.1.2.6.2 plate_approval to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.6.3 plate_approval to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.6.4 plate_approval to plate_design_definition (as subject)

AIM element: PATH
Rules: 5.2.4.7, 5.2.4.13, 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/APPROVES(product_definition_shape, 'subject')/
{/CLASS_ID(product_definition_shape, 'plate design definition')/}

5.1.2.7 PLATE_REJECTION

AIM element: group
Source: ISO 10303-41
Reference path: {[/CLASS(group, 'plate rejection', 'structural class rejection')/]
[/CLASS(group, 'structural class rejection', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]}

5.1.2.7.1 status

AIM element: /SUPERTYPE(structural_class_rejection)/ (See 5.1.2.16.1)

5.1.2.7.2 plate_ rejection to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.7.3 plate_rejection to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.7.4 plate_rejection to definition (as proposed_alternative)

AIM element: PATH
Rules: 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
(/APPROVES(product_definition_shape, 'proposed alternative')/
{/CLASS_ID(product_definition_shape, 'plate design definition')/}

5.1.2.7.5 plate_rejection to plate_design_definition (as subject)

AIM element: PATH
Rules: 5.2.4.7, 5.2.4.13, 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/APPROVES(product_definition_shape, 'subject')/
{/CLASS_ID(product_definition_shape, 'plate design definition')/}

5.1.2.8 PLATE_RENEWAL_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.215, 5.2.4.298
Reference path: {[/CLASS(product_definition_shape, 'plate renewal definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.2.8.1 description

AIM element: product_definition_shape.description
Source: ISO 10303-41

5.1.2.8.2 renewal_thickness

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
property_definition <-
/PDR_NAME('plate renewal definition attributes')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('renewal thickness', positive_length_measure)/

5.1.2.8.3 rule_thickness

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
property_definition <-
/PDR_NAME('plate renewal definition attributes')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('rule thickness', positive_length_measure)/

5.1.2.8.4 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.2.8.5 plate_renewal_definition to plate (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'plate')/}

5.1.2.8.6 plate_renewal_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.2.8.7 plate_renewal_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.2.8.8 plate_renewal_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.2.9 PROFILE_APPROVAL

AIM element: group
Source: ISO 10303-41
Reference path: {[/CLASS(group, 'profile approval', 'structural class approval')/]
[/CLASS(group, 'structural class approval', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]}

5.1.2.9.1 status

AIM element: /SUPERTYPE(structural_class_approval)/ (See 5.1.2.13.1)

5.1.2.9.2 profile_approval to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.9.3 profile_approval to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.9.4 profile_approval to profile_design_definition (as subject)

AIM element: PATH
Rules: 5.2.4.7, 5.2.4.13, 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/APPROVES(product_definition_shape, 'subject')/
{/CLASS_ID(product_definition_shape, 'profile design definition')/}

5.1.2.10 PROFILE_REJECTION

AIM element: group
Source: ISO 10303-41
Reference path: {[/CLASS(group, 'profile rejection', 'structural class rejection')/]
[/CLASS(group, 'structural class rejection', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]}

5.1.2.10.1 status

AIM element: /SUPERTYPE(structural_class_rejection)/ (See 5.1.2.16.1)

5.1.2.10.2 profile_ rejection to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.10.3 profile_rejection to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.10.4 profile_rejection to definition (as proposed_alternative)

AIM element: PATH
Rules: 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/(APPROVES(product_definition_shape, 'proposed alternative')/
{/CLASS_ID(product_definition_shape, 'profile design definition')/}

5.1.2.10.5 profile_rejection to profile_design_definition (as subject)

AIM element: PATH
Rules: 5.2.4.7, 5.2.4.13, 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/APPROVES(product_definition_shape, 'subject')/
{/CLASS_ID(product_definition_shape, 'profile design definition')/}

5.1.2.11 PROFILE_RENEWAL_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.215, 5.2.4.310
Reference path: {[/CLASS(product_definition_shape, 'profile renewal definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.2.11.1 description

AIM element: product_definition_shape.description
Source: ISO 10303-41

5.1.2.11.2 renewal_modulus

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
property_definition <-
/PDR_NAME('profile renewal definition attributes')/ ->
representation

representation.items [i] ->
/VAL_REP_ITEM_CD('renewal modulus', 'section modulus unit')/

5.1.2.11.3 rule_inertia

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
property_definition <-
/PDR('profile renewal definition attributes')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM_CD('rule inertia', 'inertia moment unit')/

5.1.2.11.4 rule_section_modulus

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
property_definition <-
/PDR_NAME('profile renewal definition attributes')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM_CD('rule section modulus', 'section modulus unit')/

5.1.2.11.5 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.2.11.6 profile_renewal_definition to profile (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'profile')/}

5.1.2.11.7 profile_renewal_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.2.11.8 profile_renewal_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.2.11.9 profile_renewal_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.2.12 REASON_FOR_DECISION

AIM element: applied_document_reference
Source: ISO 10303-218
Reference path: applied_document_reference <=
document_reference
{/ROLE_ASSGN(document_reference)/
object_role.name = 'rule reference'}

5.1.2.12.1 annotations

AIM element: document.description
Source: ISO 10303-41
Reference path: applied_document_reference <=
document_reference
document_reference.assigned_document ->
document
document.description

5.1.2.12.2 reason_for_decision to document_reference (as rule_reference)

AIM element: IDENTICAL MAPPING

5.1.2.13 STRUCTURAL_CLASS_APPROVAL

#1: structural class approval is a corrugated part approval
#2: structural class approval is a hull cross section approval
#3: structural class approval is a plate approval
#4: structural class approval is a profile approval

AIM element: #1: /SUBTYPE(corrugated_part_approval)/ (See 5.1.2.1)
#2: /SUBTYPE(hull_cross_section_approval)/ (See 5.1.2.3)
#3: /SUBTYPE(plate_approval)/ (See 5.1.2.6)
#4: /SUBTYPE(profile_approval)/ (See 5.1.2.9)

5.1.2.13.1 status

AIM element: approval_status
Source: ISO 10303-41
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval
approval.status ->
approval_status
(approval_status.name = 'approved')
(approval_status.name = 'user defined')]}

5.1.2.13.2 structural_class_approval to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.13.3 structural_class_approval to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.13.4 structural_class_approval to definition (as subject)

AIM element: #1: /SUBTYPE(corrugated_part_approval)/ (See 5.1.2.1.4)
#2: /SUBTYPE(hull_cross_section_approval)/ (See 5.1.2.3.34)
#3: /SUBTYPE(plate_approval)/ (See 5.1.2.6.4)
#4: /SUBTYPE(profile_approval)/ (See 5.1.2.9.4)

5.1.2.14 STRUCTURAL_CLASS_APPROVAL_RESULT

#1: structural_class_approval_result is a hull_cross_section_result
#2: structural_class_approval_result is a structural_class_approval
#3: structural_class_approval_result is a structural_class_conditional_approval
#4: structural_class_approval_result is a structural_class_rejection

AIM element: #1: /SUBTYPE(hull_cross_section_result)/ (See 5.1.2.5)
#2: /SUBTYPE(structural_class_approval)/ (See 5.1.2.13)
#3: /SUBTYPE(structural_class_conditional_approval)/ (See 5.1.2.15)
#4: /SUBTYPE(structural_class_rejection)/ (See 5.1.2.16)

5.1.2.14.1 status

AIM element: #1: /SUBTYPE(hull_cross_section_result)/ (See 5.1.2.5.31)
#2: /SUBTYPE(structural_class_approval)/ (See 5.1.2.13.1)
#3: /SUBTYPE(structural_class_conditional_approval)/ – (See 5.1.2.15.1)
#4: /SUBTYPE(structural_class_rejection)/ (See 5.1.2.16.1)

5.1.2.14.2 structural_class_approval_result to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.14.3 structural_class_approval_result to reason_for_decision (as explanations)

AIM element: PATH
Reference path: group
document_reference_item = group
document_reference_item <-
applied_document_reference.items[i]
applied_document_reference <=
document_reference
{/ROLE_ASSGN(document_reference)/
object_role.name = 'explanations'}

5.1.2.14.4 structural_class_approval_result to definition (as subject)

AIM element: #1: /SUBTYPE(hull_cross_section_result)/ (See 5.1.2.5.34)
#2: /SUBTYPE(structural_class_approval)/ (See 5.1.2.13.4)
#3: /SUBTYPE(structural_class_conditional_approval)/ (See 5.1.2.15.4)
#4: /SUBTYPE(structural_class_rejection)/ (See 5.1.2.16.1)

5.1.2.15 STRUCTURAL_CLASS_CONDITIONAL_APPROVAL

AIM element: group
Source: ISO 10303-41
Reference path: {[/CLASS(group, 'structural class conditional approval', 'structural class approval result')/]
[/CLASS(group, 'structural class approval result', 'approval history')/]
[/ROOT_CLASS(group, 'approval history')/]

5.1.2.15.1 status

AIM element: approval_status
Source: ISO 10303-41
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval
approval.status ->
approval_status
{approval_status.name = 'user defined'}

5.1.2.15.2 structural_class_conditional_approval to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.15.3 structural_class_conditional_approval to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.15.4 structural_class_conditional_approval to definition (as subject)

AIM element: PATH
Rules: 5.2.4.7, 5.2.4.13, 5.2.4.9
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
/APPROVES(product_definition_shape, 'subject')/
{(/CLASS_ID(product_definition_shape, 'structural part design definition')/)
(/CLASS_ID(product_definition_shape, 'structural system design definition')/)
(/CLASS_ID(product_definition_shape, 'hull cross section design definition')/)}

5.1.2.16 STRUCTURAL_CLASS_REJECTION

#1: structural class rejection is a corrugated part rejection
#2: structural class rejection is a hull cross section rejection
#3: structural class rejection is a plate rejection
#4: structural class rejection is a profile rejection

AIM element: #1: /SUBTYPE(corrugated_part_rejection)/ (See 5.1.2.2)
#2: /SUBTYPE(hull_cross_section_rejection)/ (See 5.1.2.4)
#3: /SUBTYPE(plate_rejection)/ (See 5.1.2.7)
#4: /SUBTYPE(profile_rejection)/ (See 5.1.2.10)

5.1.2.16.1 status

AIM element: approval_status
Source: ISO 10303-41
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval
approval.status ->
approval_status
(approval_status.name = 'rejected')
(approval_status.name = 'user defined')]}

5.1.2.16.2 structural_class_rejection to approval_event (as approvals)

AIM element: /SUPERTYPE(approval_history)/ (See 5.1.3.3.2)

5.1.2.16.3 structural_class_rejection to reason_for_decision (as explanations)

AIM element: /SUPERTYPE(structural_class_approval_result)/ (See 5.1.2.14.3)

5.1.2.16.4 structural_class_rejection to definition (as subject)

AIM element: #1: /SUBTYPE(corrugated_part_rejection)/ (See 5.1.2.2.5)
#2: /SUBTYPE(hull_cross_section_rejection)/ (See 5.1.2.4.35)
#3: /SUBTYPE(plate_rejection)/ (See 5.1.2.7.5)
#4: /SUBTYPE(profile_rejection)/ (See 5.1.2.10.5)

5.1.2.16.5 structural_class_rejection to definition (as proposed_-alternative)

AIM element: #1: /SUBTYPE(corrugated_part_rejection)/ (See 5.1.2.2.4)
#2: /SUBTYPE(hull_cross_section_rejection)/ (See 5.1.2.4.34)
#3: /SUBTYPE(plate_rejection)/ (See 5.1.2.7.4)
#4: /SUBTYPE(profile_rejection)/ (See 5.1.2.10.4)

5.1.3 configuration_management UoF

5.1.3.1 ALTERNATIVE_VERSION_RELATIONSHIP

AIM element: identification_assignment_relationship
Source: ISO 10303-41
Rules: 5.2.4.4, 5.2.4.6
Reference path: /ROOT_CLASS(identification_assignment_relationship, 'alternative version relationship')/

5.1.3.1.1 reason

AIM element: identification_assignment_relationship.description
Source: ISO 10303-41
Rules: 5.2.4.5

5.1.3.1.2 version_relationship to versionable_object (as alternative 1)

AIM element: PATH
Reference path: identification_assignment_relationship
identification_assignment_relationship.related_assignment ->
(/RELATE_ID_2_VO(product_definition)/)
(/RELATE_ID_2_VO(property_definition)/)
(/RELATE_ID_2_VO(product_definition_relationship)/)
(/RELATE_ID_2_VO(product_definition_shape)/)
(/RELATE_ID_2_VO(product_related_product_category)/)
(/RELATE_ID_2_VO(document)/)
(/RELATE_ID_2_VO(group)/)
(/RELATE_ID_2_VO(shape_aspect_relationship)/)

5.1.3.1.3 version_relationship to versionable_object (as alternative 2)

AIM element: PATH
Reference path: identification_assignment_relationship
identification_assignment_relationship.relating_assignment ->
(/RELATE_ID_2_VO(product_definition)/)
(/RELATE_ID_2_VO(property_definition)/)
(/RELATE_ID_2_VO(product_definition_relationship)/)
(/RELATE_ID_2_VO(product_definition_shape)/)
(/RELATE_ID_2_VO(product_related_product_category)/)
(/RELATE_ID_2_VO(document)/)
(/RELATE_ID_2_VO(group)/)
(/RELATE_ID_2_VO(shape_aspect_relationship)/)

5.1.3.2 APPROVAL_EVENT

AIM element: approval
Source: ISO 10303-41
Reference path: {[/CLASS(approval, 'approval event', 'event')/]
[/ROOT_CLASS(approval, 'event')/]}

5.1.3.2.1 caused_by

NOTE Attribute inherited from supertype Event (see 5.1.3.12).

AIM element: person_and_organization
Source: ISO 10303-41
Rules: 5.2.4.11
Reference path: approval <-
approval_person_organization.authorised_approval
approval_person_organization
approval_person_organization.person_organization ->
person_organization_select = person_and_organization
person_and_organization

5.1.3.2.2 caused_when

NOTE Attribute inherited from supertype Event (see 5.1.3.12).

AIM element: approval_date_time.date_time
Source: ISO 10303-41
Rules: 5.2.4.10
Reference path: approval <-
approval_date_time.dated_approval
approval_date_time
approval_date_time.date_time ->
{(date_time_select = date_and_time
date_and_time)
(date_time_select = calendar_date
calendar_date)}

5.1.3.2.3 description

NOTE Attribute inherited from supertype Event (see 5.1.3.12).

AIM element: approval.level
Source: ISO 10303-41

5.1.3.2.4 result

AIM element: approval_status.name
Source: ISO 10303-41
Reference path: approval
approval.status ->
approval_status
approval_status.name
{(approval_status.name = 'unapproved')
(approval_status.name = 'approved')
(approval_status.name = 'rejected')}

5.1.3.2.5 user_defined_result

AIM element: approval_status.name
Source: ISO 10303-41
Reference path: approval
approval.status ->
approval_status
approval_status.name

5.1.3.2.6 approval_event to approval_history (as approval_reference)

NOTE Attribute is inverse relationship for approval_history to approval_event (as approvals) (see 5.1.3.3.2).

AIM element: PATH
Rules: 5.2.4.73
Reference path: approval
group_item = approval
group_item <-
applied_group_assignment.items[i]
applied_group_assignment <=
group_assignment
/ROLE_ASSGN(group_assignment)/
{[object_role.name = 'approvals']
[object_role.description = '.UNUSED.']}
group_assignment.assigned_group ->
group
{/CLASS_ID(group, 'approval history')/}

5.1.3.3 APPROVAL_HISTORY

AIM element: group
Source: ISO 10303-41
Reference path: /ROOT_CLASS(group, 'approval history')/

5.1.3.3.1 status

AIM element: PATH
Reference path: group <-
GROUPS(approval, 'approvals')
approval
approval.status ->
approval_status
approval_status.name
{(approval_status.name = 'unapproved')
(approval_status.name = 'approved')
(approval_status.name = 'rejected')}

5.1.3.3.2 approval_history to approval_event (as approvals)

AIM element: PATH
Rules: 5.2.4.14, 5.2.4.356, 5.2.4.15
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval

5.1.3.3.3 approval_history to definition (as subject)

AIM element: PATH
Rules: 5.2.4.12
Reference path: group <-
/GROUPS(approval, 'approvals')/
approval <-
(/APPROVES(product_defnition, 'subject')/

product_defnition
{/CLASS_ID(product_definition, ‘definition’)/})
(/APPROVES(property_definition, 'subject')/

property_definition
{/CLASS_ID(property_definition, ‘definition’) })
(/APPROVES(product_definition_shape, 'subject')/

product_definition_shape
{/CLASS_ID(product_definition_shape, ‘definition’)/})
(/APPROVES(product_related_product_category, 'subject')/

product_related_product_category
{/CLASS_ID(product_related_product_category, ‘definition’)/})

5.1.3.4 CHANGE

AIM element: action
Source: ISO 10303-41
Reference path: {[CLASS(action,'change','item')/]
[/CLASS(action,'item','definable object')/]
[/ROOT_CLASS(action, 'definable object')/]}

5.1.3.4.1 description

NOTE Attribute inherited from supertype Item (see 5.1.9.3).

AIM element: action.description
Source: ISO 10303-41

5.1.3.4.2 name

NOTE Attribute inherited from supertype Item (see 5.1.9.3).

AIM element: action.name
Source: ISO 10303-41

5.1.3.4.3 the_class

AIM element: group.name
Source: ISO 10303-41
Reference path: action
classification_item = action
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'change class'}
classification_assignment.assigned_classification ->
group
group.name
{group.name = 'the class'}

5.1.3.4.4 change to external_reference (as documentation)

NOTE Attribute inherited from supertype Item (see 5.1.9.3).

#1: If the as documentation refers to an External_reference

AIM element: PATH
Reference path: action
action = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment

#2: If the as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: action
/DOC_REF(action,'documentation')/
document
{/CLASS_ID(document, 'document reference with address')/}

5.1.3.4.5 change to global_id (as id)

NOTE Attribute inherited from supertype Item (see 5.1.9.3), which inherits from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.3
Reference path: action
identification_item = action
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.3.4.6 change to ship (as ship_context)

NOTE Attribute inherited from supertype Item (see 5.1.9.3).

AIM element: PATH
Reference path: action <-
applied_action_assignment.assigned_action
applied_action_assignment
applied_action_assignment.items[i]->
action_item
action_item = product
product
{/CLASS_ID(product, 'ship')/}

5.1.3.5 CHANGE_DEFINITION

#1: If the Change_definition is a change request
#2: If the Change_definition is a change plan
#3: If the Change_definition is a change realization

AIM element: #1: /SUBTYPE(Change_request)/5.1.3.9
#2: /SUBTYPE(Change_plan)/5.1.3.7
#3: /SUBTYPE(Change_realization)/5.1.3.8

5.1.3.5.1 author

AIM element: #1: /SUBTYPE(Change_request)/5.1.3.9.2
#2: /SUBTYPE(Change_plan)/5.1.3.7.1
#3: /SUBTYPE(Change_realization)/5.1.3.8.1

5.1.3.5.2 date_time

AIM element: #1: /SUBTYPE(Change_request)/5.1.3.9.3
#2: /SUBTYPE(Change_plan)/5.1.3.7.2
#3: /SUBTYPE(Change_realization)/5.1.3.8.2

5.1.3.5.3 description

AIM element: #1: /SUBTYPE(Change_request)/5.1.3.9.4
#2: /SUBTYPE(Change_plan)/5.1.3.7.3
#3: /SUBTYPE(Change_realization)/5.1.3.8.3

5.1.3.5.4 version_id

AIM element: #1: /SUBTYPE(Change_request)/5.1.3.9.8
#2: /SUBTYPE(Change_plan)/5.1.3.7.4
#3: /SUBTYPE(Change_realization)/5.1.3.8.4

5.1.3.5.5 change_definition to change (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(Change_request)/5.1.3.9.9
#2: /SUBTYPE(Change_plan)/5.1.3.7.7
#3: /SUBTYPE(Change_realization)/5.1.3.8.6

5.1.3.5.6 change_definition to global_id (as id)

AIM element: #1: /SUBTYPE(Change_request)/5.1.3.9.10
#2: /SUBTYPE(Change_plan)/5.1.3.7.8
#3: /SUBTYPE(Change_realization)/5.1.3.8.7

5.1.3.6 CHANGE_IMPACT

AIM element: applied_action_request_assignment
Source: ISO 10303-218
Reference path: applied_action_request_assignment <=
action_request_assignment
{/ROOT_CLASS(applied_action_request_assignment, ‘change impact’)/}

5.1.3.6.1 change_impact to versionable_object_change_event (as impact)

AIM element: PATH
Rules: 5.2.4.31
Reference path: applied_action_request_assignment
applied_action_request_assignment.items [i] ->
action_request_item
action_request_item = action
{/CLASS_ID(action, ‘versionable object change event’)/}

5.1.3.7 CHANGE_PLAN

AIM element: action_request_solution
Source: ISO 10303-41
Reference path: {[/CLASS(action_request_solution, ‘change plan’, ‘change definition’)/]
[/CLASS(action_request_solution,’change definition’,’definition’)/]
[/CLASS(action_request_solution,’definition’,’versionable object’)/]
[/ROOT_CLASS(action_request_solution, ‘versionable object’)/]}

5.1.3.7.1 author

NOTE Attribute inherited from supertype Change_definition (see 5.1.3.5).

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-218
Rules: 5.2.4.17
Reference path: /PERS_ORG_ASSGN(action_request_solution, 'author')/

5.1.3.7.2 date_time

NOTE Attribute inherited from supertype Change_definition (see 5.1.3.5).

AIM element: applied_date_and_time_assignment.assigned_date_and_time
Source: ISO 10303-218
Rules: 5.2.4.1
Reference path: /DAT_TIME_ASSGN(action_request_solution,'date time')/

5.1.3.7.3 description

NOTE Attribute inherited from supertype Item (see 5.1.9.3).

AIM element: action_request_solution.description
Source: ISO 10303-41
Reference path: /DESCRIPTION_ASSGN(action_request_solution)/

5.1.3.7.4 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7) through supetype Change_definition (see 5.1.3.5), which inherits from supertype Definition (see 5.1.4.1).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(action_request_solution)/

5.1.3.7.5 change_plan to check (as checks)

AIM element: PATH
Reference path: action_request_solution
action_item = action_request_solution
action_item <-
applied_action_assignment.items[i]
applied_action_assignment <=
action_assignment
action_assignment.assigned_action ->
action
{/CLASS_ID(action, ‘check’)}

5.1.3.7.6 change_plan to change_request (as chosen_solution_for)

AIM element: PATH
Reference path: action_request_solution
action_request_solution.request ->
versioned_action_request
{/CLASS_ID(versioned_action_request, ‘change request’)/}

5.1.3.7.7 change_plan to change (as defined_for)

NOTE Attribute inherited from supertype Change_definition (see 5.1.3.5).

AIM element: PATH
Rules: 5.2.4.1
Reference path: action_request_solution
action_request_solution.method ->
action_method <-
action.chosen_method
action
{/CLASS_ID(action, ‘change‘)/}

5.1.3.7.8 change_plan to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1) through supetype Change_definition (see 5.1.3.5).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.2
Reference path: action_request_solution
identification_item = action_request_solution
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.3.7.9 change_plan to change_impact (as planned_impact)

AIM element: PATH
Rules: 5.2.4.32
Reference path: action_request_solution
action_request_solution.request ->
versioned_action_request <-
action_request_assignment.assigned_action_request
action_request_assignment =>
applied_action_request_assignment
{/CLASS_ID(applied_action_request_assignment, ‘change impact’)/}

5.1.3.8 CHANGE_REALIZATION

AIM element: executed_action
Source: ISO 10303-41
Reference path: {[/CLASS(executed_action, ‘change realization’, ‘change definition’)/]
[/CLASS(executed_action,’change definition’,’definition’)/]
[/CLASS(executed_action,’definition’,’versionable object’)/]
[/ROOT_CLASS(executed_action, ‘versionable object’)/]}

5.1.3.8.1 author

NOTE Attribute inherited from supertype Change_definition (see 5.1.3.5).

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-218
Rules: 5.2.4.18
Reference path: /PERS_ORG_ASSGN(executed_action, 'author')/

5.1.3.8.2 date_time

NOTE Attribute inherited from supertype Change_definition (see 5.1.3.5).

AIM element: applied_date_and_time_assignment.assigned_date_and_time
Source: ISO 10303-218
Rules: 5.2.4.38
Reference path: /DAT_TIME_ASSGN(executed_action,'date time')/

5.1.3.8.3 description

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: executed_action.description
Source: ISO 10303-41

5.1.3.8.4 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7) through supetype Change_definition (see 5.1.3.5), which inherits from supertype Definition (see 5.1.4.1).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(executed_action)/

5.1.3.8.5 change_realization to check (as checks)

AIM element: PATH
Reference path: executed_action
action_item = executed_action
action_item <-
applied_action_assignment.items[i]
applied_action_assignment <=
action_assignment
action_assignment.assigned_action ->
action
{/CLASS_ID(action, ‘check’)/}

5.1.3.8.6 change_realization to change (as defined_for)

NOTE Attribute inherited from supertype Change_definition (see 5.1.3.5).

AIM element: PATH
Reference path: executed_action <=
action <-
action_relationship.related_action
action_relationship
action_relationship.relating_action ->
action
{/CLASS_ID(action, ‘change‘)/}

5.1.3.8.7 change_realization to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1) through supetype Change_definition (see 5.1.3.5).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.51
Reference path: executed_action
identification_item = executed_action
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.3.8.8 change_realization to change_impact (as impact)

AIM element: PATH
Reference path: executed_action
action_request_item = executed_action
action_request_item <-
applied_action_request_assignment.items[i]
applied_action_request_assignment
{/CLASS_ID(applied_action_request_assignment, ‘change impact’)/}

5.1.3.8.9 change_realization to change_plan (as realization_of)

AIM element: PATH
Reference path: executed_action <=
action
action.chosen_method ->
action_method <-
action_request_solution.method
action_request_solution
{/CLASS_ID(action_request_solution, ‘change plan’)/}

5.1.3.9 CHANGE_REQUEST

AIM element: versioned_action_request
Source: ISO 10303-41
Reference path: {[/CLASS(versioned_action_request, ‘change request’, ‘change definition’)/]
[/CLASS(versioned_action_request,’change definition’,’definition’)/]
[/CLASS(versioned_action_request,’definition’,’versionable object’)/]
[/ROOT_CLASS(versioned_action_request, ‘versionable object’)/]}

5.1.3.9.1 addressee

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-218
Reference path: /PERS_ORG_ASSGN(versioned_action_request, 'addressee')/

5.1.3.9.2 author

NOTE Attribute inherited from supertype Change_definition (see 5.1.3.5).

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-218
Rules: 5.2.4.19
Reference path: /PERS_ORG_ASSGN(versioned_action_request, 'author')/

5.1.3.9.3 date_time

NOTE Attribute inherited from supertype Change_definition (see 5.1.3.5).

AIM element: applied_date_and_time_assignment.assigned_date_and_time
Source: ISO 10303-218
Rules: 5.2.4.39
Reference path: /DAT_TIME_ASSGN(versioned_action_request,'date time')/

5.1.3.9.4 description

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: versioned_action_request.description
Source: ISO 10303-41

5.1.3.9.5 initiator

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-218
Rules: 5.2.4.67
Reference path: /PERS_ORG_ASSGN(versioned_action_request, 'initiator')/

5.1.3.9.6 problem

AIM element: versioned_action_request.purpose
Source: ISO 10303-41

5.1.3.9.7 solution_description

AIM element: action_request_solution.description
Source: ISO 10303-41
Reference path: versioned_action_request <-
action_request_solution.request
action_request_solution
/DESCRIPTION_ASSGN(action_request_solution)/

5.1.3.9.8 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7) through supetype Change_definition (see 5.1.3.5), which inherits from supertype Definition (see 5.1.4.1).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(versioned_action_request)/

5.1.3.9.9 change_request to change (as defined_for)

NOTE Attribute inherited from supertype Change_definition (see 5.1.3.5).

AIM element: PATH
Reference path: versioned_action_request <-
action_request_solution.request
action_request_solution
action_request_solution.method ->
action_method <-
action.chosen_method
action
{/CLASS_ID(action, ‘change‘)/}

5.1.3.9.10 change_request to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1) through supetype Change_definition (see 5.1.3.5).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.370
Reference path: versioned_action_request
identification_item = versioned_action_request
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.3.9.11 change_request to change_impact (as solution_alternatives)

AIM element: PATH
Reference path: versioned_action_request <-
action_request_assignment.assigned_action_request
action_request_assignment =>
applied_action_request_assignment
{/CLASS_ID(applied_action_request_assignment, ‘change impact’)/}

5.1.3.10 CHECK

AIM element: action
Source: ISO 10303-41
Reference path: {[/CLASS(action, ‘check’, ‘event’)/]
[/ROOT_CLASS(action, ‘event’)/]}

5.1.3.10.1 caused_by

NOTE Attribute inherited from supertype Event (see 5.1.3.12).

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-218
Rules: 5.2.4.20
Reference path: /PERS_ORG_ASSGN(action, 'caused by')/

5.1.3.10.2 caused_when

NOTE Attribute inherited from supertype Event (see 5.1.3.12).

AIM element: applied_date_and_time_assignment.assigned_date_and_time
Source: ISO 10303-218
Rules: 5.2.4.25
Reference path: /DAT_TIME_ASSGN(action, 'caused when')/

5.1.3.10.3 description

NOTE Attribute inherited from supertype Event (see 5.1.3.12).

AIM element: action.description
Source: ISO 10303-41

5.1.3.11 ENVISAGED_VERSION_CREATION

AIM element: action
Source: ISO 10303-41
Reference path: {[/CLASS(action, ‘envisaged version creation’, ‘versionable object change event’)/]
[/CLASS(action, ‘versionable object change event’, ‘event’)/]
[/ROOT_CLASS(action, ‘event’)/]}

5.1.3.11.1 category

AIM element: action.name
Source: ISO 10303-41

5.1.3.11.2 caused_by

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supertype Versionable_object_change_-event (see 5.1.3.20).

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-41
Rules: 5.2.4.21
Reference path: /PERS_ORG_ASSGN(action, 'caused by')/

5.1.3.11.3 caused_when

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: applied_date_and_time_assignment.assigned_date_and_time
Source: ISO 10303-218
Rules: 5.2.4.26
Reference path: /DAT_TIME_ASSGN(action,'caused when')/

5.1.3.11.4 description

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: action.description
Source: ISO 10303-41
Rules: 5.2.4.50

5.1.3.11.5 envisaged_version_creation to versionable_object (as base)

AIM element: PATH
Reference path: action <-
action_assignment.assigned_action
action_assignment
/ROLE_ASSGN(action_assignment)/
{object_role.name = 'base'}
action_assignment =>
(/RELATE_ACT_2_VO(product_definition)/)
(/RELATE_ACT_2_VO(property_definition)/)
(/RELATE_ACT_2_VO(product_definition_relationship)/)
(/RELATE_ACT_2_VO(product_definition_shape)/)
(/RELATE_ACT_2_VO(product_related_product_category)/)
(/RELATE_ACT_2_VO(document)/)
(/RELATE_ACT_2_VO(group)/)
(/RELATE_ACT_2_VO(shape_aspect_relationship)/)

5.1.3.12 EVENT

#1: If the Event is a Versionable_object_change_event
#2: If the Event is an Approval_event
#3: If the Event is a Check

AIM element: #1: action
#2: approval
#3: action
Source: ISO 10303-41

5.1.3.12.1 caused_by

AIM element: /SUBTYPE(Versionable_object_change_event)/5.1.3.20
/SUBTYPE(Approval_event)/5.1.3.1, 5.1.3.2
/SUBTYPE(Check)/5.1.3.10

5.1.3.12.2 caused_when

AIM element: /SUBTYPE(Versionable_object_change_event)/5.1.3.20
/SUBTYPE(Approval_event)/5.1.3.1, 5.1.3.2
/SUBTYPE(Check)/5.1.3.10

5.1.3.12.3 description

AIM element: /SUBTYPE(Versionable_object_change_event)/5.1.3.20
/SUBTYPE(Approval_event)/5.1.3.1, 5.1.3.2
/SUBTYPE(Check)/5.1.3.10

5.1.3.13 REVISION

AIM element: group
Source: ISO 10303-41
Rules: 5.2.4.72
Reference path: {[/CLASS(group, ‘revision’, ‘versionable object’)/]
[/ROOT_CLASS(group, ‘versionable object’)/]}

5.1.3.13.1 description

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: group.description
Source: ISO 10303-41
Rules: 5.2.4.314

5.1.3.13.2 name

AIM element: group.name
Source: ISO 10303-41

5.1.3.13.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(group)/

5.1.3.13.4 revision to versionable_object (as members)

AIM element: PATH
Rules: 5.2.4.72
Reference path: group <-
(/RELATE_GROUP_2_VO(product_definition, 'members')/)
(/RELATE_GROUP_2_VO(property_definition, 'members')/)
(RELATE_GROUP_2_VO(product_definition_relationship, 'members')/)
(/RELATE_GROUP_2_VO(product_definition_shape, 'members')/)
(/RELATE_GROUP_2_VO(product_related_product_category, 'members')/)
(/RELATE_GROUP_2_VO(document, 'members')/)
(/RELATE_GROUP_2_VO(group, 'members')/)
(/RELATE_GROUP_2_VO(shape_aspect_relationship, 'members')/)

5.1.3.14 REVISION_WITH_CONTEXT

AIM element: group
Source: ISO 10303-41
Reference path: {[/CLASS(group, 'revision with context', 'revision')/]
[/CLASS(group, ‘revision’, ‘versionable object’)/]
[/ROOT_CLASS(group, ‘versionable object’)/]}

5.1.3.14.1 description

NOTE Attribute inherited from supertype Revision (see 5.1.3.13).

AIM element: group.description
Source: ISO 10303-41
Rules: 5.2.4.314

5.1.3.14.2 name

NOTE Attribute inherited from supertype Revision (see 5.1.3.13).

AIM element: group.name
Source: ISO 10303-41

5.1.3.14.3 version_id

NOTE Attribute inherited from supertype Revision (see 5.1.3.13), which inherits from supertype Versionable_object (see 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(group)/

5.1.3.14.4 revision_with_context to definable_object (as context_of_-revision)

AIM element: PATH
Rules: 5.2.4.315
Reference path: group <-
(/RELATE_GROUP_2_DO(product, 'context of revision')/)
(/RELATE_GROUP_2_DO(product_definition, ‘context of revision’)/)
(/RELATE_GROUP_2_DO(product_definition_relationship, ‘context of revision’) /)
(/RELATE_GROUP_2_DO(group, ‘context of revision’)/)
(/RELATE_GROUP_2_DO(shape_aspect, ‘context of revision’)/)
(/RELATE_GROUP_2_DO(shape_aspect_relationship, ‘context of revision’)/)

5.1.3.14.5 revision_with_context to versionable_object (as members)

NOTE Attribute inherited from supertype Revision (see 5.1.3.13).

AIM element: /SUPERTYPE(revision)/5.1.3.13

5.1.3.15 VERSION_CREATION

AIM element: action
Source: ISO 10303-41
Reference path: {[/CLASS(action, 'version creation', 'versionable object change event')/]
[/CLASS(action, 'versionable object change event', 'event')/]
[/ROOT_CLASS(action, 'event')/]}

5.1.3.15.1 caused_by

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-218
Rules: 5.2.4.22
Reference path: /PERS_ORG_ASSGN(action, 'caused by')/

5.1.3.15.2 caused_when

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: applied_date_and_time_assignment.assigned_date_and_time
Source: ISO 10303-218
Rules: 5.2.4.27
Reference path: /DAT_TIME_ASSGN(action, 'caused when')/

5.1.3.15.3 description

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: action.description
Source: ISO 10303-41
Rules: 5.2.4.360

5.1.3.15.4 version_creation to versionable_object (as base)

AIM element: PATH
Reference path: action <-
action_assignment.assigned_action
action_assignment
/ROLE_ASSGN(action_assignment)/
{object_role.name = 'base'}
action_assignment =>
(/RELATE_ACT_2_VO(product_definition)/)
(/RELATE_ACT_2_VO(property_definition)/)
(/RELATE_ACT_2_VO(product_definition_relationship)/)
(/RELATE_ACT_2_VO(product_definition_shape)/)
(/RELATE_ACT_2_VO(product_related_product_category)/)
(/RELATE_ACT_2_VO(document)/)
(/RELATE_ACT_2_VO(group)/)
(/RELATE_ACT_2_VO(shape_aspect_relationship)/)

5.1.3.15.5 version_creation to versionable_object (as subject)

AIM element: PATH
Reference path: action <-
action_assignment.assigned_action
action_assignment
/ROLE_ASSGN(action_assignment)/
{object_role.name = 'subject'}
action_assignment =>
(/RELATE_ACT_2_VO(product_definition)/)
(/RELATE_ACT_2_VO(property_definition)/)
(/RELATE_ACT_2_VO(product_definition_relationship)/)
(/RELATE_ACT_2_VO(product_definition_shape)/)
(/RELATE_ACT_2_VO(product_related_product_category)/)
(/RELATE_ACT_2_VO(document)/)
(/RELATE_ACT_2_VO(group)/)
(/RELATE_ACT_2_VO(shape_aspect_relationship)/)

5.1.3.16 VERSION_DELETION

AIM element: action
Source: ISO 10303-41
Reference path: {[/CLASS(action, 'version deletion', 'versionable object change event')/]
[/CLASS(action, 'versionable object change event', 'event')/]
[/ROOT_CLASS(action, 'event')/]}

5.1.3.16.1 caused_by

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-218
Rules: 5.2.4.23
Reference path: /PERS_ORG_ASSGN(action, 'caused by')/

5.1.3.16.2 caused_when

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: applied_date_and_time_assignment.assigned_date_and_time
Source: ISO 10303-218
Rules: 5.2.4.28
Reference path: /DAT_TIME_ASSGN(action, ‘caused when')/

5.1.3.16.3 description

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: action.description
Source: ISO 10303-41
Rules: 5.2.4.361

5.1.3.16.4 version_deletion to versionable_object (as subject)

AIM element: PATH
Reference path: action <-
action_assignment.assigned_action
action_assignment
/ROLE_ASSGN(action_assignment)/
{object_role.name = ‘subject’}
action_assignment =>
(/RELATE_ACT_2_VO(product_definition)/)
(/RELATE_ACT_2_VO(property_definition)/)
(/RELATE_ACT_2_VO(product_definition_relationship)/)
(/RELATE_ACT_2_VO(product_definition_shape)/)
(/RELATE_ACT_2_VO(product_related_product_category)/)
(/RELATE_ACT_2_VO(document)/)
(/RELATE_ACT_2_VO(group)/)
(/RELATE_ACT_2_VO(shape_aspect_relationship)/)

5.1.3.17 VERSION_HISTORY

AIM element: group
Source: ISO 10303-41
Rules: 5.2.4.364
Reference path: /ROOT_CLASS(group, 'version history')/

5.1.3.17.1 version_history to versionable_object (as current_version)

AIM element: PATH
Rules: 5.2.4.363, 5.2.4.362
Reference path: group <-
(/RELATE_GROUP_2_VO(product_definition, 'versions')/)
(/RELATE_GROUP_2_VO(property_definition, 'versions')/)
(/RELATE_GROUP_2_VO(product_definition_relationship, 'versions')/)
(/RELATE_GROUP_2_VO(product_definition_shape, 'versions')/)
(/RELATE_GROUP_2_VO(product_related_product_category, 'versions')/)
(/RELATE_GROUP_2_VO(document, 'versions')/)
(/RELATE_GROUP_2_VO(group, 'versions')/)
(/RELATE_GROUP_2_VO(shape_aspect_relationship, 'versions')/)

5.1.3.17.2 version_history to version_relationship (as relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(identification_assignment_relationship, 'relationships')/
identification_assignment_relationship
{/CLASS_ID(identification_assignment_relationship, 'version relationship')/}

5.1.3.17.3 version_history to versionable_object (as versions)

AIM element: PATH
Rules: 5.2.4.371
Reference path: group <-
(/RELATE_GROUP_2_VO(product_definition, 'versions')/)
(/RELATE_GROUP_2_VO(property_definition, 'versions')/)
(RELATE_GROUP_2_VO(product_definition_relationship, 'versions')/)
(/RELATE_GROUP_2_VO(product_definition_shape, 'versions')/)
(/RELATE_GROUP_2_VO(product_related_product_category, 'versions')/)
(/RELATE_GROUP_2_VO(document, 'versions')/)
(/RELATE_GROUP_2_VO(group, 'versions')/)
(/RELATE_GROUP_2_VO(shape_aspect_relationship, 'versions')/)

5.1.3.18 VERSION_MODIFICATION

AIM element: action
Source: ISO 10303-41
Reference path: {[/CLASS(action, ‘version modification’, ‘versionable object change event’)/]
[/CLASS(action, ‘versionable object change event’, ‘event’)/]
[/ROOT_CLASS(action, ‘event’)/]}

5.1.3.18.1 caused_by

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: applied_person_and_organization_assignment.assigned_person_and_organization
Source: ISO 10303-218
Rules: 5.2.4.24
Reference path: /PERS_ORG_ASSGN(action, 'caused by')/

5.1.3.18.2 caused_when

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: applied_date_and_time_assignment.assigned_date_and_time
Source: ISO 10303-218
Rules: 5.2.4.29
Reference path: /DAT_TIME_ASSGN(action,'caused when')/

5.1.3.18.3 description

NOTE Attribute inherited from supertype Event (see 5.1.3.12), through supetype Versionable_object_change_-event (see 5.1.3.20).

AIM element: action.description
Source: ISO 10303-41
Rules: 5.2.4.365

5.1.3.18.4 version_modification to versionable_object (as base)

AIM element: PATH
Reference path: action <-
action_assignment.assigned_action
action_assignment
/ROLE_ASSGN(action_assignment)/
{object_role.name = 'base'}
action_assignment =>
(/RELATE_ACT_2_VO(product_definition)/)
(/RELATE_ACT_2_VO(property_definition)/)
(/RELATE_ACT_2_VO(product_definition_relationship)/)
(/RELATE_ACT_2_VO(product_definition_shape)/)
(/RELATE_ACT_2_VO(product_related_product_category)/)
(/RELATE_ACT_2_VO(document)/)
(/RELATE_ACT_2_VO(group)/)
(/RELATE_ACT_2_VO(shape_aspect_relationship)/)

5.1.3.18.5 version_modification to versionable_object (as subject)

AIM element: PATH
Reference path: action <-
action_assignment.assigned_action
action_assignment
/ROLE_ASSGN(action_assignment)/
{object_role.name = 'subject'}
action_assignment =>
(/RELATE_ACT_2_VO(product_definition)/)
(/RELATE_ACT_2_VO(property_definition)/)
(/RELATE_ACT_2_VO(product_definition_relationship)/)
(/RELATE_ACT_2_VO(product_definition_shape)/)
(/RELATE_ACT_2_VO(product_related_product_category)/)
(/RELATE_ACT_2_VO(document)/)
(/RELATE_ACT_2_VO(group)/)
(/RELATE_ACT_2_VO(shape_aspect_relationship)/)

5.1.3.19 VERSION_RELATIONSHIP

AIM element: identification_assignment_relationship
Source: ISO 10303-41
Rules: 5.2.4.366, 5.2.4.368
Reference path: /ROOT_CLASS(identification_assignment_relationship, 'version relationship')/

5.1.3.19.1 reason

AIM element: identification_assignment_relationship.description
Source: ISO 10303-41
Rules: 5.2.4.367

5.1.3.19.2 version_relationship to versionable_object (as predecessor)

AIM element: PATH
Reference path: identification_assignment_relationship
identification_assignment_relationship.related_assignment ->
(/RELATE_ID_2_VO(product_definition)/)
(/RELATE_ID_2_VO(property_definition)/)
(/RELATE_ID_2_VO(product_definition_relationship)/)
(/RELATE_ID_2_VO(product_definition_shape)/)
(/RELATE_ID_2_VO(product_related_product_category)/)
(/RELATE_ID_2_VO(document)/)
(/RELATE_ID_2_VO(group)/)

5.1.3.19.3 version_relationship to versionable_object (as successor)

AIM element: PATH
Reference path: identification_assignment_relationship
identification_assignment_relationship.relating_assignment ->
(/RELATE_ID_2_VO(product_definition)/)
(/RELATE_ID_2_VO(property_definition)/)
(/RELATE_ID_2_VO(product_definition_relationship)/)
(/RELATE_ID_2_VO(product_definition_shape)/)
(/RELATE_ID_2_VO(product_related_product_category)/)
(/RELATE_ID_2_VO(document)/)
(/RELATE_ID_2_VO(group)/)

5.1.3.20 VERSIONABLE_OBJECT_CHANGE_EVENT

#1: Versionable_object_change_event is an envisaged version creation
#2: Versionable_object_change_event is a version creation
#3: Versionable_object_change_event is a version modification
#4: Versionable_object_change_event is a version deletion

AIM element: #1: /SUBTYPE(Envisaged_version_creation)/5.1.3.11
#2: /SUBTYPE(Version_creation)/5.1.3.15
#3: /SUBTYPE(Version_modification)/5.1.3.18
#4: /SUBTYPE(Version_deletion)/5.1.3.16

5.1.3.20.1 caused_by

NOTE Attribute inherited from supertype Event (see 5.1.3.12).

AIM element: #1: /SUBTYPE(Envisaged_version_creation)/5.1.3.11.2
#2: /SUBTYPE(Version_creation)/5.1.3.15.1
#3: /SUBTYPE(Version_modification)/5.1.3.18.1
#4: /SUBTYPE(Version_deletion)/5.1.3.16.1

5.1.3.20.2 caused_when

NOTE Attribute inherited from supertype Event (see 5.1.3.12).

AIM element: #1: /SUBTYPE(Envisaged_version_creation)/5.1.3.11.3
#2: /SUBTYPE(Version_creation)/5.1.3.15.2
#3: /SUBTYPE(Version_modification)/5.1.3.18.2
#4: /SUBTYPE(Version_deletion)/5.1.3.16.2

5.1.3.20.3 description

NOTE Attribute inherited from supertype Event (see 5.1.3.12).

AIM element: #1: /SUBTYPE(Envisaged_version_creation)/5.1.3.11.4
#2: /SUBTYPE(Version_creation)/5.1.3.15.3
#3: /SUBTYPE(Version_modification)/5.1.3.18.3
#4: /SUBTYPE(Version_deletion)/5.1.3.16.3

5.1.4 definitions UoF

5.1.4.1 DEFINITION

#1: definition is a change definition
#2: definition is a design definition
#3: definition is a functional definition
#4: definition is a general characteristics definition
#5: definition is a library definition
#6: definition is a lightship definition
#7: definition is a local coordinate system
#8: definition is a manufacturing definition
#9: definition is a plate renewal definition
#10: definition is a profile renewal definition
#11: definition is a ship material property
#12: definition is a spacing table

AIM element: #1: /SUBTYPE(change_definition)/ (See 5.1.3.5)
#2: /SUBTYPE(design_definition)/ (See 5.1.4.2)
#3: /SUBTYPE(functional_definition)/ (See 5.1.4.3)
#4: /SUBTYPE(general_characteristics_definition)/ (See 5.1.4.4)
#5: /SUBTYPE(library_definition)/ (See 5.1.4.5)
#6: /SUBTYPE(lightship_definition)/ (See 5.1.14.6)
#7: /SUBTYPE(local_co_ordinate_system)/ (See 5.1.11.4)
#8: /SUBTYPE(manufacturing_definition)/ (See 5.1.4.8)
#9: /SUBTYPE(plate_renewal_definition)/ (See 5.1.2.8)
#10: /SUBTYPE(profile_renewal_definition)/ (See 5.1.2.11)
#11: /SUBTYPE(ship_material_property)/ (See 5.1.16.2)
#12: /SUBTYPE(spacing_table)/ (See 5.1.11.10)

5.1.4.1.1 description

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: #1: /SUBTYPE(change_definition)/ (See 5.1.3.5.3)
#2: /SUBTYPE(design_definition)/ (See 5.1.4.2.1)
#3: /SUBTYPE(functional_definition)/ (See 5.1.4.3.1)
#4: /SUBTYPE(general_characteristics_definition)/ (See 5.1.4.4.1)
#5: /SUBTYPE(library_definition)/ (See 5.1.4.5.1)
#6: /SUBTYPE(lightship_definition)/ (See 5.1.14.6.1)
#7: /SUBTYPE(local_co_ordinate_system)/ (See 5.1.11.4.1)
#8: /SUBTYPE(manufacturing_definition)/ (See 5.1.4.8.1)
#9: /SUBTYPE(plate_renewal_definition)/ (See 5.1.2.8, 5.1.2.8.1)
#10: /SUBTYPE(profile_renewal_definition)/ (See 5.1.2.11.1)
#11: /SUBTYPE(ship_material_property)/ (See 5.1.16.2.2)
#12: /SUBTYPE(spacing_table)/ (See 5.1.11.10.1)

5.1.4.1.2 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: #1: /SUBTYPE(change_definition)/ (See 5.1.3.5.4)
#2: /SUBTYPE(design_definition)/ (See 5.1.4.2.3)
#3: /SUBTYPE(functional_definition)/ (See 5.1.4.3.2)
#4: /SUBTYPE(general_characteristics_definition)/ (See 5.1.4.4.2)
#5: /SUBTYPE(library_definition)/ (See 5.1.4.5.2)
#6: /SUBTYPE(lightship_definition)/ (See 5.1.14.6.3)
#7: /SUBTYPE(local_co_ordinate_system)/ (See 5.1.11.4.2)
#8: /SUBTYPE(manufacturing_definition)/ (See 5.1.4.8.3)
#9: /SUBTYPE(plate_renewal_definition)/ (See 5.1.2.8.4)
#10: /SUBTYPE(profile_renewal_definition)/ (See 5.1.2.11.5)
#11: /SUBTYPE(ship_material_property)/ (See 5.1.16.2.7)
#12: /SUBTYPE(spacing_table)/ (See 5.1.11.10.3)

5.1.4.1.3 definition to definable_object (as defined_for)

AIM element: #1: /SUBTYPE(change_definition)/ (See 5.1.3.5.5)
#2: /SUBTYPE(design_definition)/ (See 5.1.4.2.4)
#3: /SUBTYPE(functional_definition)/ (See 5.1.4.3.4)
#4: /SUBTYPE(general_characteristics_definition)/ (See 5.1.4.4.3)
#5: /SUBTYPE(library_definition)/ (See 5.1.4.5.3)
#6: /SUBTYPE(lightship_definition)/ (See 5.1.14.6.9)
#7: /SUBTYPE(local_co_ordinate_system)/ (See 5.1.11.4.3)
#8: /SUBTYPE(manufacturing_definition)/ (See 5.1.4.8.4)
#9: /SUBTYPE(plate_renewal_definition)/ (See 5.1.2.8.5)
#10: /SUBTYPE(profile_renewal_definition)/ (See 5.1.2.11.6)
#11: /SUBTYPE(ship_material_property)/ (See 5.1.16.2)
#12: /SUBTYPE(spacing_table)/ (See 5.1.11.10.4)

5.1.4.1.4 definition to global_id (as id)

AIM element: #1: /SUBTYPE(change_definition)/ (See 5.1.3.5.6)
#2: /SUBTYPE(design_definition)/ (See 5.1.4.2.5)
#3: /SUBTYPE(functional_definition)/ (See 5.1.4.3.5)
#4: /SUBTYPE(general_characteristics_definition)/ (See 5.1.4.4.4)
#5: /SUBTYPE(library_definition)/ (See 5.1.4.5.4)
#6: /SUBTYPE(lightship_definition)/ (See 5.1.14.6.6)
#7: /SUBTYPE(local_co_ordinate_system)/ (See 5.1.11.4.4)
#8: /SUBTYPE(manufacturing_definition)/ (See 5.1.4.8.5)
#9: /SUBTYPE(plate_renewal_definition)/ (See 5.1.2.8.8)
#10: /SUBTYPE(profile_renewal_definition)/ (See 5.1.2.11.7)
#11: /SUBTYPE(ship_material_property)/ (See 5.1.16.2.4)
#12: /SUBTYPE(spacing_table)/ (See 5.1.11.10.5)

5.1.4.1.5 definition to derived_unit (as local_units)

AIM element: #2: /SUBTYPE(design_definition)/ (See 5.1.4.2.6)
#4: /SUBTYPE(general_characteristics_definition)/ (See 5.1.4.4.5)
#5: /SUBTYPE(library_definition)/ (See 5.1.4.5.5)
#6: /SUBTYPE(lightship_definition)/ (See 5.1.14.6.5)
#7: /SUBTYPE(local_co_ordinate_system)/ (See 5.1.11.4.5)
#8: /SUBTYPE(manufacturing_definition)/ (See 5.1.4.8.6)
#9: /SUBTYPE(plate_renewal_definition)/ (See 5.1.2.8.6)
#10: /SUBTYPE(profile_renewal_definition)/ (See 5.1.2.11.8)
#11: /SUBTYPE(ship_material_property)/ (See 5.1.16.2.5)
#12: /SUBTYPE(spacing_table)/ (See 5.1.11.10.6)

5.1.4.1.6 definition to named_unit (as local_units)

AIM element: #2: /SUBTYPE(design_definition)/ (See 5.1.4.2.7)
#4: /SUBTYPE(general_characteristics_definition) -- (See /5.1.4.4.6)
#5: /SUBTYPE(library_definition)/ (See 5.1.4.5.7)
#6: /SUBTYPE(lightship_definition)/ (See 5.1.14.6.8)
#7: /SUBTYPE(local_co_ordinate_system)/ (See 5.1.11.4.6)
#8: /SUBTYPE(manufacturing_definition)/ (See 5.1.4.8.7)
#9: /SUBTYPE(plate_renewal_definition)/ (See 5.1.2.8.7)
#10: /SUBTYPE(profile_renewal_definition)/ (See 5.1.2.11.9)
#11: /SUBTYPE(ship_material_property)/ (See 5.1.16.2.6)
#12: /SUBTYPE(spacing_table)/ (See 5.1.11.10.7)

5.1.4.2 DESIGN_DEFINITION

#1: if design_definition is a feature design definition
#2: if design_definition is a hull cross section design definition
#3: if design_definition is a library design definition
#4: if design_definition is a structural part design definition
#5: if design_definition is a structural part joint design definition
#6: if design_definition is a system design definition

#7: if design_definition is a weld design definition

AIM element: #1: /SUBTYPE(feature_design_definition)/ (See 5.1.12.6)
#2: /SUBTYPE(hull_cross_section_design_definition)/ (See 5.1.7.3)
#3: /SUBTYPE(library_design_definition)/ (See 5.1.4.6)
#4: /SUBTYPE(structural_part_design_definition)/ (See 5.1.19.36)
#5: /SUBTYPE(structural_part_joint_design_definition)/ (See 5.1.21.11)
#6: /SUBTYPE(system_design_definition)/ (See 5.1.12.11)
#7: /SUBTYPE(weld_design_definition)/ (See 5.1.21.13)

5.1.4.2.1 description

AIM element: #1: /SUBTYPE(feature_design_definition)/ (See 5.1.12.6.1)
#2: /SUBTYPE(hull_cross_section_design_definition)/ (See 5.1.7.3.1)
#3: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.1)
#4: /SUBTYPE(structural_part_design_definition)/ (See 5.1.19.36.1)
#5: /SUBTYPE(structural_part_joint_design_definition)/ (See 5.1.21.11.1)
#6: /SUBTYPE(system_design_definition)/ (See 5.1.12.11.1)
#7: /SUBTYPE(weld_design_definition)/ (See 5.1.21.13.1)

5.1.4.2.2 representations

AIM element: #1: /SUBTYPE(feature_design_definition)/ (See 5.1.12.6.7, 5.1.12.6.8, 5.1.12.6.9, 5.1.12.6.10)
#2: /SUBTYPE(hull_cross_section_design_definition)/ (See 5.1.7.3.5, 5.1.7.3.10, 5.1.7.3.10)
#3: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.2)
#4: /SUBTYPE(structural_part_design_definition)/ (See 5.1.19.36.8, 5.1.19.36.9, 5.1.19.36.10, 5.1.19.36.11)
#5: /SUBTYPE(structural_part_joint_design_definition)/ (See 5.1.21.11.8, 5.1.21.11.9, 5.1.21.11.10, 5.1.21.11.11)
#6: /SUBTYPE(system_design_definition)/ (See 5.1.12.11.7, 5.1.12.11.8, 5.1.12.11.9, 5.1.12.11.10)
#7: /SUBTYPE(weld_design_definition)/ (See 5.1.21.13.16, 5.1.21.13.17, 5.1.21.13.18, 5.1.21.13.19)

5.1.4.2.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7) through Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(feature_design_definition)/ (See 5.1.12.6.2)
#2: /SUBTYPE(hull_cross_section_design_definition)/ (See 5.1.7.3.3)
#3: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.3)
#4: /SUBTYPE(structural_part_design_definition)/ (See 5.1.19.36.3)
#5: /SUBTYPE(structural_part_joint_design_definition)/ (See 5.1.21.11.12)
#6: /SUBTYPE(system_design_definition)/ (See 5.1.12.11.2)
#7: /SUBTYPE(weld_design_definition)/ (See 5.1.21.13.20)

5.1.4.2.4 design_definition to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(feature_design_definition)/ (See 5.1.12.6.3)
#2: /SUBTYPE(hull_cross_section_design_definition)/ (See 5.1.7.3.8)
#3: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.4)
#4: /SUBTYPE(structural_part_design_definition)/ (See 5.1.19.36.4)
#5: /SUBTYPE(structural_part_joint_design_definition)/ (See 5.1.21.11.4)
#6: /SUBTYPE(system_design_definition)/ (See 5.1.12.11.3)
#7: /SUBTYPE(weld_design_definition)/ (See 5.1.21.13.12)

5.1.4.2.5 design_definition to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(feature_design_definition)/ (See 5.1.12.6.4)
#2: /SUBTYPE(hull_cross_section_design_definition)/ (See 5.1.7.3.7)
#3: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.5)
#4: /SUBTYPE(structural_part_design_definition)/ (See 5.1.19.36.5)
#5: /SUBTYPE(structural_part_joint_design_definition)/ (See 5.1.21.11.5)
#6: /SUBTYPE(system_design_definition)/ (See 5.1.12.11.4)
#7: /SUBTYPE(weld_design_definition)/ (See 5.1.21.13.13)

5.1.4.2.6 design_definition to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(feature_design_definition)/ (See 5.1.12.6.5)
#2: /SUBTYPE(hull_cross_section_design_definition)/ (See 5.1.7.3.4)
#3: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.6)
#4: /SUBTYPE(structural_part_design_definition)/ (See 5.1.19.36.6)
#5: /SUBTYPE(structural_part_joint_design_definition)/ (See 5.1.21.11.6)
#6: /SUBTYPE(system_design_definition)/ (See 5.1.12.11.5)
#7: /SUBTYPE(weld_design_definition)/ (See 5.1.21.13.14)

5.1.4.2.7 design_definition to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(feature_design_definition)/ (See 5.1.12.6.6)
#2: /SUBTYPE(hull_cross_section_design_definition)/ (See 5.1.7.3.9)
#3: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.8)
#4: /SUBTYPE(structural_part_design_definition)/ (See 5.1.19.36.7)
#5: /SUBTYPE(structural_part_joint_design_definition)/ (See 5.1.21.11.7)
#6: /SUBTYPE(system_design_definition)/ (See 5.1.12.11.6)
#7: /SUBTYPE(weld_design_definition)/ (See 5.1.21.13.15)

5.1.4.3 FUNCTIONAL_DEFINITION

#1: functional definition is an edge cutout functional definition
#2: functional definition is an edge feature functional definition
#3: functional definition is an interior cutout functional definition
#4: functional definition is a shiptype
#5: functional definition is a structural part functional definition
#6: functional definition is a structural system functional definition

AIM element: #1: /SUBTYPE(edge_cutout_functional_definition)/ (See 5.1.18.8)
#2: /SUBTYPE(edge_feature_functional_definition)/ (See 5.1.18.10)
#3: /SUBTYPE(interior_cutout_functional_definition)/ (See 5.1.18.16)
#4: /SUBTYPE(shiptype)/ (See 5.1.14.16)
#5: /SUBTYPE(structural_part_functional_definition)/ (See 5.1.19.37)
#6: /SUBTYPE(structural_system_functional_definition)/ (See 5.1.20.19)

5.1.4.3.1 description

AIM element: #1: /SUBTYPE(edge_cutout_functional_definition)/ (See 5.1.18.8.1)
#2: /SUBTYPE(edge_feature_functional_definition)/ (See 5.1.18.10.1)
#3: /SUBTYPE(interior_cutout_functional_definition)/ (See 5.1.18.16.1)
#4: /SUBTYPE(shiptype)/ (See 5.1.14.16.1)
#5: /SUBTYPE(structural_part_functional_definition)/ (See 5.1.19.37.1)
#6: /SUBTYPE(structural_system_functional_definition)/ (See 5.1.20.19.1)

5.1.4.3.2 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7) through Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(edge_cutout_functional_definition)/ (See 5.1.18.8.4)
#2: /SUBTYPE(edge_feature_functional_definition)/ (See 5.1.18.10.4)
#3: /SUBTYPE(interior_cutout_functional_definition)/ (See 5.1.18.16.4)
#4: /SUBTYPE(shiptype)/ (See 5.1.14.16.3)
#5: /SUBTYPE(structural_part_functional_definition)/ (See 5.1.19.37.4)
#6: /SUBTYPE(structural_system_functional_definition)/ (See 5.1.20.19.6)

5.1.4.3.3 user_def_function

AIM element: #1: /SUBTYPE(edge_cutout_functional_definition)/ (See 5.1.18.8.3)
#2: /SUBTYPE(edge_feature_functional_definition)/ (See 5.1.18.10.3)
#3: /SUBTYPE(interior_cutout_functional_definition)/ (See 5.1.18.16.3)
#4: /SUBTYPE(shiptype)/ (See 5.1.14.16.2)
#5: /SUBTYPE(structural_part_functional_definition)/ (See 5.1.19.37.3)
#6: /SUBTYPE(structural_system_functional_definition)/ (See 5.1.20.19.4)

5.1.4.3.4 functional_definition to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(edge_cutout_functional_definition)/ (See 5.1.18.8.5)
#2: /SUBTYPE(edge_feature_functional_definition)/ (See 5.1.18.10.5)
#3: /SUBTYPE(interior_cutout_functional_definition)/ (See 5.1.18.16.5)
#4: /SUBTYPE(shiptype)/ (See 5.1.14.16.4)
#5: /SUBTYPE(structural_part_functional_definition)/ (See 5.1.19.37.5)
#6: /SUBTYPE(structural_system_functional_definition)/ (See 5.1.20.19.2)

5.1.4.3.5 functional_definition to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(edge_cutout_functional_definition)/ (See 5.1.18.8.6)
#2: /SUBTYPE(edge_feature_functional_definition)/ (See 5.1.18.10.6)
#3: /SUBTYPE(interior_cutout_functional_definition)/ (See 5.1.18.16.6)
#4: /SUBTYPE(shiptype)/ (See 5.1.14.16.5)
#5: /SUBTYPE(structural_part_functional_definition)/ (See 5.1.19.37.6)
#6: /SUBTYPE(structural_system_functional_definition)/ (See 5.1.20.19.5)

5.1.4.4 GENERAL_CHARACTERISTICS_DEFINITION

#1: if general_characteristics_definition is a class and statutory designation
#2: if general_characteristics_definition is a class parameters
#3: if general_characteristics_definition is a design load
#4: if general_characteristics_definition is a freeboard characteristics
#5: if general_characteristics_definition is a global axis placement
#6: if general_characteristics_definition is an owner designation
#7: if general_characteristics_definition is a principal characteristics
#8: if general_characteristics_definition is a ship designation
#9: if general_characteristics_definition is a shipyard designation

AIM element: #1: /SUBTYPE(class_and_statutory_designation)/ (See 5.1.14.2)
#2: /SUBTYPE(class_parameters)/ (See 5.1.14.4)
#3: /SUBTYPE(design_load)/ (See 5.1.5.2)
#4: /SUBTYPE(freeboard_characteristics)/ (See 5.1.14.5)
#5: /SUBTYPE(global_axis_placement)/ (See 5.1.11.3)
#6: /SUBTYPE(owner_designation)/ (See 5.1.14.11)
#7: /SUBTYPE(principal_characteristics)/ (See 5.1.14.12)
#8: /SUBTYPE(ship_designation)/ (See 5.1.14.15)
#9: /SUBTYPE(shipyard_designation)/ (See 5.1.14.17)

5.1.4.4.1 description

AIM element: #1: /SUBTYPE(class_and_statutory_designation)/ (See 5.1.14.2.2)
#2: /SUBTYPE(class_parameters)/ (See 5.1.14.4.2)
#3: /SUBTYPE(design_load)/ (See 5.1.5.2.1)
#4: /SUBTYPE(freeboard_characteristics)/ (See 5.1.14.5.3)
#5: /SUBTYPE(global_axis_placement)/ (See 5.1.11.3.2)
#6: /SUBTYPE(owner_designation)/ (See 5.1.14.11.1)
#7: /SUBTYPE(principal_characteristics)/ (See 5.1.14.12.2)
#8: /SUBTYPE(ship_designation)/ (See 5.1.14.15.2)
#9: /SUBTYPE(shipyard_designation)/ (See 5.1.14.17.1)

5.1.4.4.2 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7) through Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(class_and_statutory_designation)/ (See 5.1.14.2.3)
#2: /SUBTYPE(class_parameters)/ (See 5.1.14.4.8)
#3: /SUBTYPE(design_load)/ (See 5.1.5.2.2)
#4: /SUBTYPE(freeboard_characteristics)/ (See 5.1.14.5.6)
#5: /SUBTYPE(global_axis_placement)/ (See 5.1.11.3.4)
#6: /SUBTYPE(owner_designation)/ (See 5.1.14.11.6)
#7: /SUBTYPE(principal_characteristics)/ (See 5.1.14.12.12)
#8: /SUBTYPE(ship_designation)/ (See 5.1.14.15.7)
#9: /SUBTYPE(shipyard_designation)/ (See 5.1.14.17.6)

5.1.4.4.3 general_characteristics_definition to ship (as defined_for)

AIM element: PATH
Reference path: /PROD_DEF_PRODUCT/
{/CLASS_ID(product, 'ship')/}

5.1.4.4.4 general_characteristics_definition to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(class_and_statutory_designation)/ (See 5.1.14.2.5)
#2: /SUBTYPE(class_parameters)/ (See 5.1.14.4.10)
#3: /SUBTYPE(design_load)/ (See 5.1.5.2.4)
#4: /SUBTYPE(freeboard_characteristics)/ (See 5.1.14.5.8)
#5: /SUBTYPE(global_axis_placement)/ (See 5.1.11.3.6)
#6: /SUBTYPE(owner_designation)/ (See 5.1.14.11.8)
#7: /SUBTYPE(principal_characteristics)/ (See 5.1.14.12.14)
#8: /SUBTYPE(ship_designation)/ (See 5.1.14.15.9)
#9: /SUBTYPE(shipyard_designation)/ (See 5.1.14.17.8)

5.1.4.4.5 general_characteristics_definition to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(class_and_statutory_designation)/ (See note)
#2: /SUBTYPE(class_parameters)/ (See 5.1.14.5.7)
#3: /SUBTYPE(design_load)/ (See 5.1.5.2.3)
#4: /SUBTYPE(freeboard_characteristics)/ (See 5.1.14.5.7)
#5: /SUBTYPE(global_axis_placement)/ (See 5.1.11.3.7)
#6: /SUBTYPE(owner_designation)/ (See note)
#7: /SUBTYPE(principal_characteristics)/ (See 5.1.14.12.15)
#8: /SUBTYPE(ship_designation)/ (See note)
#9: /SUBTYPE(shipyard_designation)/ (See note)

NOTE For this subtype the attribute has been re-declared in the ARM to be a set of zero, which is interpreted as the attribute is not required

5.1.4.4.6 general_characteristics_definition to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(class_and_statutory_designation)/ (See note)
#2: /SUBTYPE(class_parameters)/ (See 5.1.14.5.10)
#3: /SUBTYPE(design_load)/ (See 5.1.5.2.5)
#4: /SUBTYPE(freeboard_characteristics)/ (See 5.1.14.5.10)
#5: /SUBTYPE(global_axis_placement)/ (See 5.1.11.3.8)
#6: /SUBTYPE(owner_designation)/ (See note)
#7: /SUBTYPE(principal_characteristics)/ (See 5.1.14.12.16)
#8: /SUBTYPE(ship_designation)/ (See note)
#9: /SUBTYPE(shipyard_designation)/ (See note)

NOTE For this subtype the attribute has been re-declared in the ARM to be a set of zero, which is interpreted as the attribute is not required

5.1.4.5 LIBRARY_DEFINITION

#1: if library_definition is a library design definition
#2: if library_definition is a library manufacturing definition

AIM element: #1: /SUBTYPE(library_design_definition)/ (See 5.1.4.6)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7)

5.1.4.5.1 description

AIM element: #1: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.1)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.1)

5.1.4.5.2 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7) through Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.3)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.3)

5.1.4.5.3 library_definition to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.4)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.4)

5.1.4.5.4 library_definition to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.5)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.5)

5.1.4.5.5 library_definition to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.6)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.6)

5.1.4.5.6 library_definition to library_element_reference (as source)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.7)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.7)

5.1.4.5.7 library_definition to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(library_design_definition)/ (See 5.1.4.6.8)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.8)

5.1.4.6 LIBRARY_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.112
Reference path: {[/CLASS(product_definition_shape, ‘library design definition‘, ‘library definition‘)/]
[/CLASS(product_definition_shape, ‘library definition‘, ‘definition‘)/]
[/CLASS(product_definition_shape, ‘library design definition‘, ‘design definition‘)/]
[/CLASS(product_definition_shape, ‘design definition‘, ‘definition‘)/]
[/CLASS(product_definition_shape, ‘definition‘,‘versionable object‘)/]
[/ROOT_CLASS(product_definition_shape,‘versionable object‘)/]}

5.1.4.6.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Resource path: product_definition_shape <=
property_definition
property_definition.description

5.1.4.6.2 representations

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR/ ->
representation
{representation.name= 'representations'}

5.1.4.6.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.4.6.4 library_design_definition to definable_object (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/

5.1.4.6.5 library_design_definition to global_id (as id)

AIM element: PATH

Rules: 5.2.4.56, 5.2.4.140
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.4.6.6 library_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.4.6.7 library_design_definition to library_element_reference (as source)

AIM element: PATH
Rules: 5.2.4.115
Reference path: product_definition_shape
classification_item = product_definition_shape
classification_item <-
applied_classification_assignment.items
applied_classification_assignment

5.1.4.6.8 library_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.4.7 LIBRARY_MANUFACTURING_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.113
Reference path: {[/CLASS(product_definition_shape, 'library manufacturing definition', 'library definition')}
[/CLASS(product_definition_shape, 'library definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition','versionable object')/]
[/ROOT_CLASS(product_definition_shape,'versionable object')/]}

5.1.4.7.1 description

AIM element: property_definition.description
Source: ISO 10303-41
AIM element: product_definition_shape <=
property_definition
property_definition.description

5.1.4.7.2 representations

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR/ ->
representation
{representation.name = 'representations'}

5.1.4.7.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.4.7.4 library_manufacturing_definition to definable_object (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/

5.1.4.7.5 library_manufacturing_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.172
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.4.7.6 library_manufacturing_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.4.7.7 library_manufacturing_definition to library_element_reference (as source)

AIM element: PATH
Rules: 5.2.4.68
Reference path: product_definition_shape
classification_item = product_definition_shape
classification_item <-
applied_classification_assignment.items
applied_classification_assignment

5.1.4.7.8 library_manufacturing_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.4.8 MANUFACTURING_DEFINITION

#1: if manufacturing_definition is an assembly manufacturing definition
#2: if manufacturing_definition is a library manufacturing definition
#3: if manufacturing_definition is a structural part manufacturing definition
#4: if manufacturing_definition is a weld manufacturing definition

AIM element: #1: /SUBTYPE(assembly_manufacturing_definition)/ (See 5.1.15.3)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7)
#3: /SUBTYPE(structural_part_manufacturing_definition)/ (See 5.1.15.11)
#4: /SUBTYPE(weld_manufacturing_definition)/ (See 5.1.15.14)

5.1.4.8.1 description

AIM element: #1: /SUBTYPE(assembly_manufacturing_definition)/ (See 5.1.15.3.4)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.1)
#3: /SUBTYPE(structural_part_manufacturing_definition)/ (See 5.1.15.11.3)
#4: /SUBTYPE(weld_manufacturing_definition)/ (See 5.1.15.14.2)

5.1.4.8.2 representations

AIM element: #1: /SUBTYPE(assembly_manufacturing_definition)/ (See 5.1.15.3.11, 5.1.15.3.13, 5.1.15.3.14, 5.1.15.3.12)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.2)
#3: /SUBTYPE(structural_part_manufacturing_definition)/ (See 5.1.15.11.11, 5.1.15.11.13, 5.1.15.11.12, 5.1.15.11.14)
#4: /SUBTYPE(weld_manufacturing_definition)/ (See 5.1.15.14.15, 5.1.15.14.17, 5.1.15.14.16, 5.1.15.14.18)

5.1.4.8.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7) through Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(assembly_manufacturing_definition)/ (See 5.1.15.3.7)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.3)
#3: /SUBTYPE(structural_part_manufacturing_definition)/ (See 5.1.15.11.8, 5.1.15.11.9, 5.1.15.11.9)
#4: /SUBTYPE(weld_manufacturing_definition)/ (See 5.1.15.14.10)

5.1.4.8.4 manufacturing_definition to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(assembly_manufacturing_definition)/ (See 5.1.15.3.8)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.4)
#3: /SUBTYPE(structural_part_manufacturing_definition)/ (See 5.1.15.11.10)
#4: /SUBTYPE(weld_manufacturing_definition)/ (See 5.1.15.14.14)

5.1.4.8.5 manufacturing_definition to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(assembly_manufacturing_definition)/ (See 5.1.15.3.15)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.5)
#3: /SUBTYPE(structural_part_manufacturing_definition)/ (See 5.1.15.11.15)
#4: /SUBTYPE(weld_manufacturing_definition)/ (See 5.1.15.14.19)

5.1.4.8.6 manufacturing_definition to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(assembly_manufacturing_definition)/ (See 5.1.15.3.16)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.6)
#3: /SUBTYPE(structural_part_manufacturing_definition)/ (See 5.1.15.11.16)
#4: /SUBTYPE(weld_manufacturing_definition)/ (See 5.1.15.14.20)

5.1.4.8.7 manufacturing_definition to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: #1: /SUBTYPE(assembly_manufacturing_definition)/ (See 5.1.15.3.17)
#2: /SUBTYPE(library_manufacturing_definition)/ (See 5.1.4.7.8)
#3: /SUBTYPE(structural_part_manufacturing_definition)/ (See 5.1.15.11.17)
#4: /SUBTYPE(weld_manufacturing_definition)/ (See 5.1.15.14.21)

5.1.5 design_loads UoF

5.1.5.1 BENDING_MOMENT

#1: bending moment is a design still water bending moment
#2: bending moment is a design vertical wave bending moment
#3: bending moment is a maximum permissible still water bending moment

AIM element: #1: /SUBTYPE(design_still_water_bending_moment)/ (See 5.1.5.3)
#2: /SUBTYPE(design_vertical_wave_bending_moment)/ (See 5.1.5.5)
#3: /SUBTYPE(maximum_permissible_still_water_bending_moment)/ (See 5.1.5.10)

5.1.5.1.1 description

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.1.2 version_id

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.1.3 bending_moment to derived_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.1.4 bending_moment to global_id (as id)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.1.5 bending_moment to named_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.1.6 bending_moment to ship (as defined_for)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.2 DESIGN_LOAD

#1: design load is a bending moment
#2: design load is a shear force

AIM element: #1: /SUBTYPE(bending_moment)/ (See 5.1.5.1)
#2: /SUBTYPE(shear_force)/ (See 5.1.5.12)

5.1.5.2.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.5.2.2 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.5.2.3 design_load to derived_unit (as local_units)

AIM element: PATH
Rules: 5.2.4.215
Reference path: /PROD_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.5.2.4 design_load to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.5.2.5 design_load to named_unit (as local_units)

AIM element: PATH
Rules: 5.2.4.215
Reference path: /PROD_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.5.2.6 design_load to ship (as defined_for)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')/}

5.1.5.3 DESIGN_STILL_WATER_BENDING_MOMENT

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.41
Reference path: {[/CLASS(product_definition, 'design still water bending moment',
'bending moment')/]
[/CLASS(product_definition, 'bending moment‘,‘design load')/]
[/CLASS(product_definition, 'design load‘,‘general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition‘,‘definition')/]
[/CLASS(product_definition, 'definition‘,‘versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.5.3.1 description

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.3.2 hogging_amidship

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.263
Reference path: product_definition =
characterized_product_definition =
characterized_definition <-
property_definition.definition
property_definition <-
/PDR_NAME('design still water bending moment parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD('hogging amidship', 'moment unit')/

5.1.5.3.3 sagging_amidship

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.263
Reference path: product_definition =
characterized_product_definition =
characterized_definition <-
property_definition.definition
property_definition <-
/PDR_NAME('design still water bending moment parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD('sagging amidship', 'moment unit')/

5.1.5.3.4 version_id

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.3.5 design_still_water_bending_moment to derived_unit (as local_-units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.3.6 design_still_water_bending_moment to global_id (as id)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.3.7 design_still_water_bending_moment to named_unit (as local_-units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.3.8 design_still_water_bending_moment to ship (as defined for)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2

5.1.5.4 DESIGN_STILL_WATER_SHEAR_FORCE

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.83
Reference path: {[/CLASS(product_definition, 'design still water shear force', 'shear force')/]
[/CLASS(product_definition, 'shear force', 'design load')/]
[/CLASS(product_definition, 'design load‘,‘general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition‘,‘definition')/]
[/CLASS(product_definition, 'definition‘,‘versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.5.4.1 description

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.4.2 version_id

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.4.3 design_still_water_shear_force to derived_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.4.4 design_still_water_shear_force to design_SWSF_values (as table_of_shear_force_values)

AIM element: PATH
Rules: 5.2.4.279
Reference path: product_definition =
characterized_product_definition =
characterized_definition <-
property_definition.definition
property_definition <-
/PDR_NAME('design still water shear force values')/ ->
representation
{representation.name = 'table of shear force values'}
representation.items[i] ->
representation_item
{representation_item.name = 'design SWSF values'}
representation_item =>
compound_representation_item

5.1.5.4.5 design_still_water_shear_force to global_id (as id)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.4.6 design_still_water_shear_force to named_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.4.7 design_still_water_shear_force to ship (as defined for)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.5 DESIGN_VERTICAL_WAVE_BENDING_MOMENT

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.42
Reference path: {[/CLASS(product_definition, 'design vertical wave bending moment',
'bending moment')/]
[/CLASS(product_definition, 'bending moment', 'design load')/]
[/CLASS(product_definition, 'design load‘,‘general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition‘,‘definition')/]
[/CLASS(product_definition, 'definition‘,‘versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.5.5.1 description

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.5.2 hogging

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215 , 5.2.4.174
Reference path: product_definition =
characterized_product_definition =
characterized_definition <-
property_definition.definition
property_definition <-
/PDR_NAME(‘design vertical wave bending moment parameters‘)/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD(‘hogging‘, ‘moment unit’)/

5.1.5.5.3 sagging

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215 , 5.2.4.174
Reference path: product_definition =
characterized_product_definition =
characterized_definition <-
property_definition.definition
property_definition <-
/PDR_NAME('design vertical wave bending moment parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD('sagging', 'moment unit')/

5.1.5.5.4 version_id

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.5.5 design_vertical_wave_bending_moment to derived_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.5.6 design_vertical_wave_bending_moment to global_id (as id)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.5.7 design_vertical_wave_bending_moment to named_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.5.8 design_vertical_wave_bending_moment to ship (as defined for)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.6 DESIGN_VERTICAL_WAVE_SHEAR_FORCE

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.43
Reference path: {[/CLASS(product_definition, 'design vertical wave shear force', 'shear force')/]
[/CLASS(product_definition,'shear force', 'design load')/]
[/CLASS(product_definition, 'design load‘,‘general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition‘,‘definition')/]
[/CLASS(product_definition, 'definition‘,‘versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.5.6.1 description

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.6.2 negative_value

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215 , 5.2.4.183
Reference path: product_definition =
characterized_product_definition =
characterized_definition <-
property_definition.definition
property_definition <-
/PDR_NAME('design vertical wave shear force parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD('negative value', ‘force unit’)/

5.1.5.6.3 positive_value

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215 , 5.2.4.183
Reference path: product_definition =
characterized_product_definition =
characterized_definition <-
property_definition.definition
property_definition <-
/PDR_NAME('design vertical wave shear force parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD('positive value', ‘force unit’)/

5.1.5.6.4 version_id

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.6.5 design_vertical_wave_shear_force to derived_unit (as local_-units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.6.6 design_vertical_wave_shear_force to global_id (as id)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.6.7 design_vertical_wave_shear_force to named_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.6.8 design_vertical_wave_shear_force to ship (as defined for)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.7 DESIGN_SWSF_VALUES

AIM element: compound_representation_item
Source: ISO 10303-43

5.1.5.7.1 negative_value

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.260
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition =
list_representation_item
list_representation_item [1] ->
/VAL_REP_ITEM_CD('negative value', 'force unit')/

5.1.5.7.2 positive_value

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.260
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition =
list_representation_item
list_representation_item [2] ->
/VAL_REP_ITEM_CD('positive value', 'force unit')/

5.1.5.7.3 design_SWSF_values to spacing_position (as position)

AIM element: PATH
Rules: 5.2.4.260
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition =
list_representation_item
list_representation_item [3] ->
representation_item =>
{representation_item.name = 'position'}
compound_representation_item
{/CLASS_ID(compound_representation_item, 'longitudinal position')/}

5.1.5.8 MAXIMUM_PERMISSIBLE_SWBM_VALUES

AIM element: compound_representation_item
Source: ISO 10303-43

5.1.5.8.1 hogging

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.230
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition =
list_representation_item
list_representation_item [1] ->
/VAL_REP_ITEM_CD(‘hogging‘, ‘moment unit’)/

5.1.5.8.2 sagging

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.230
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition =
list_representation_item
list_representation_item [2] ->
/VAL_REP_ITEM_CD(‘sagging‘, ‘moment unit’)/

5.1.5.8.3 maximum_permissible_SWBM_values to spacing_position (as position)

AIM element: PATH
Source:
Rules: 5.2.4.230
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition =
list_representation_item
list_representation_item [3] ->
representation_item =>
{representation_item.name = 'position'}
compound_representation_item
{/CLASS_ID(compound_representation_item, 'longitudinal position')/}

5.1.5.9 MAXIMUM_PERMISSIBLE_SWSF_VALUES

AIM element: compound_representation_item
Source: ISO 10303-43

5.1.5.9.1 negative_value

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.231
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition =
list_representation_item
list_representation_item [1] ->
/VAL_REP_ITEM_CD('negative value', 'force unit')/

5.1.5.9.2 positive_value

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.231
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition =
list_representation_item
list_representation_item [2] ->
/VAL_REP_ITEM_CD('positive value', 'force unit')/

5.1.5.9.3 maximum_permissible_SWSF_values to spacing_position (as position)

AIM element: PATH
Rules: 5.2.4.231
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition =
list_representation_item
list_representation_item [3] ->
representation_item =>
{representation_item.name = 'position'}
compound_representation_item
{/CLASS_ID(compound_representation_item, 'longitudinal position')/}

5.1.5.10 MAXIMUM_PERMISSIBLE_STILL_WATER_BENDING_-MOMENT

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.40
Reference path: {[/CLASS(product_definition,
'maximum permissible still water bending moment', 'bending moment')/]
[/CLASS(product_definition, 'bending moment', 'design load')/]
[/CLASS(product_definition, 'design load‘,‘general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition‘,‘definition')/]
[/CLASS(product_definition, 'definition‘,‘versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.5.10.1 description

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.10.2 version_id

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.10.3 maximum_permissible_still_water_bending_moment to maximum_permissible_swbm_values (as table_of_max_bending_-moment_values)

AIM element: PATH
Rules: 5.2.4.262
Reference path: product_definition =
characterized_product_definition =
characterized_definition <-
property_definition.definition
property_definition <-
/PDR_NAME('maximum permissible still water bending moment values')/ ->
representation
{representation.name = 'table of moment value'}
representation.items[i] ->
representation_item
{representation_item.name = 'maximum permissible SWBM values'}
representation_item =>
compound_representation_item

5.1.5.10.4 maximum_permissible_still_water_bending_moment to derived_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.10.5 maximum_permissible_still_water_bending_moment to global_-id (as id)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.10.6 maximum_permissible_still_water_bending_moment to named_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.10.7 maximum_permissible_still_water_bending_moment to ship (as defined for)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.11 MAXIMUM_PERMISSIBLE_STILL_WATER_SHEAR_FORCE

AIM element: Product_definition
Source: ISO 10303-41
Rules: 5.2.4.44
Reference path: {[/CLASS(product_definition,
'maximum permissible still water shear force','shear force')/]
[/CLASS(product_definition, 'shear force', 'design load')/]
[/CLASS(product_definition, 'design load‘,‘general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition‘,‘definition')/]
[/CLASS(product_definition, 'definition‘,‘versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.5.11.1 description

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.11.2 version_id

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.11.3 maximum_permissible_still_water_shear_force to maximum_-permissible_swsf_values (as table_of_max_shear_force_values)

AIM element: PATH
Rules: 5.2.4.264
Reference path: product_definition =
characterized_product_definition =
characterized_definition <-
property_definition.definition
property_definition <-
/PDR_NAME('maximum permissible still water shear force values')/ ->
representation
{representation.name = 'table of shear force value'}
representation.items[i] ->
representation_item
{representation_item.name = 'maximum permissible SWSF values'}
representation_item =>
compound_representation_item

5.1.5.11.4 maximum_permissible_still_water_shear_force to derived_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.11.5 maximum_permissible_still_water_shear_force to global_id (as id)

AIM element: /SUPERTYPE(design_load)/ – (See 5.1.5.2)

5.1.5.11.6 maximum_permissible_still_water_shear_force to named_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.11.7 maximum_permissible_still_water_shear_force to ship (as defined for)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.12 SHEAR_FORCE

#1: shear force is a design still water shear force
#2: shear force is a design vertical wave shear force
#3: shear force is a maximum permissible still water shear force

AIM element: #1: /SUBTYPE(design_still_water_shear_force)/ (See 5.1.5.4)
#2: /SUBTYPE(design_vertical_wave_shear_force)/ (See 5.1.5.6)
#3: /SUBTYPE(maximum_permissible_still_water_shear_force)/ (See 5.1.5.11)

5.1.5.12.1 description

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.12.2 version_id

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.12.3 shear_force to derived_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.12.4 shear_force to global_id (as id)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.12.5 shear_force to named_unit (as local_units)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.5.12.6 shear_force to ship (as defined_for)

AIM element: /SUPERTYPE(design_load)/ (See 5.1.5.2)

5.1.6 external_references UoF

5.1.6.1 DOCUMENT

AIM element: document
Source: ISO 10303-41
Rules: 5.2.4.46
Reference path: {[/CLASS(document,'document', 'versionable object')/]
[/ROOT_CLASS(document,'versionable object')/]}

5.1.6.1.1 author

#1: if author is a person
#2: if author is an organization
#3: if author is a person_and_organization

AIM element: #1: applied_person_assignment.assigned_person
#2: applied_organization_assignment.assigned_organization
#3: applied_person_and_organization_assignment.assigned_person_and_-
organization

Source: #1: ISO 10303-218
#2: ISO 10303-218
#3: ISO 10303-218

Reference path:
#1: /PERS_ASSGN(document,'author')/
#2: /ORG_ASSGN(document,'author')/
#3: /PERS_ORG_ASSGN(document,'author')/

5.1.6.1.2 description

AIM element: document.description
Source: ISO 10303-41

5.1.6.1.3 source_type

AIM element: document_representation_type.name
Source: ISO 10303-41
Rules: 5.2.4.45
Reference path: document <-
document_representation_type.represented_document
document_representation_type
document_representation_type.name

5.1.6.1.4 title

AIM element: document.name
Source: ISO 10303-41

5.1.6.1.5 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(document)/

5.1.6.2 DOCUMENT_PORTION

AIM element: document_usage_constraint
Source: ISO 10303-41

5.1.6.2.1 element_type

AIM element: document_usage_constraint.subject_element
Source: ISO 10303-41

5.1.6.2.2 element_value

AIM element: document_usage_constraint.subject_element_value
Source: ISO 10303-41

5.1.6.2.3 document_portion to document (as source)

AIM element: document_usage_constraint.source
Reference path: document_usage_constraint
document_usage_constraint.source ->
document

5.1.6.3 DOCUMENT_REFERENCE

AIM element: document
Source: ISO 10303-41
Reference path: {/ROOT_CLASS(document,‘document reference’)/}

5.1.6.3.1 document_reference to document (as assigned_document)

AIM element: PATH
Rules: 5.2.4.47
Reference path: document

5.1.6.3.2 document_reference to document_portion (as assigned_-document)

AIM element: PATH
Rules: 5.2.4.47
Reference path: document <-
document_usage_constraint.source
document_usage_constraint

5.1.6.4 DOCUMENT_REFERENCE_WITH_ADDRESS

AIM element: document
Source: ISO 10303-41
Rules: 5.2.4.49
Reference path: {[/CLASS(document,'document reference with address', 'document reference')/]
[/ROOT_CLASS(document,'document reference')/]
[/CLASS(document,'document reference with address','external reference')/]
[/ROOT_CLASS(document,'external reference')/]}

5.1.6.4.1 description

NOTE Attribute inherited from supertype External_reference (see 5.1.6.6).

AIM element: identification_role.description
Source: ISO 10303-41
Reference path: document
external_identification_item = document
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.role ->
identificaton_role
{identification_role.name = 'external reference'}
identification_role.description

5.1.6.4.2 document_reference_with_address to document (as assigned_-document)

NOTE Attribute inherited from supertype Document_reference (see 5.1.6.3).

AIM element: PATH
Rules: 5.2.4.48
Reference path: document

5.1.6.4.3 document_reference_with_address to document_portion (as assigned_document)

NOTE Attribute inherited from supertype Document_reference (see 5.1.6.3).

AIM element: PATH
Rules: 5.2.4.48
Reference path: document <-
document_usage_constraint.source
document_usage_constraint

5.1.6.4.4 document_reference_with_address to external_storage (as location)

NOTE Attribute inherited from supertype External_reference (see 5.1.6.6).

AIM element: PATH
Reference path: document
external_identification_item = document
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment
[external_identification_assignment.source ->
external_source
{/CLASS_ID(external_source,'external storage')/}]
[external_identification_assignment <=
identification_assignment
identification_assignment.role ->
identificaton_role
{identification_role.name = 'external reference'}]

5.1.6.4.5 document_reference_with_address to universal_resource_locator (as location)

NOTE Attribute inherited from supertype External_reference (see 5.1.6.6).

AIM element: PATH
Reference path: document
external_identification_item = document
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment
[external_identification_assignment.source ->
external_source
{/CLASS_ID(external_source,'universal resource locator')/}]
[external_identification_assignment <=

identification_assignment
identification_assignment.role ->
identificaton_role
{identification_role.name = 'external reference'}]

5.1.6.5 EXTERNAL_INSTANCE_REFERENCE

AIM element: applied_external_identification_assignment
Source: ISO 10303-218
Reference path: applied_external_identification_assignment <=
external_identification_assignment <=
{/ID_ROLE('external instance reference')/}

5.1.6.5.1 entity_type

AIM element: external_source.source_id
Source: ISO 10303-41
Rules: 5.2.4.70
Reference path: applied_external_identification_assignment <=
external_identification_assignment
external_identification_assignment.source ->
external_source
{/DESCRIPTION_ASSGN(external_source)/
description_attribute.attribute_value = 'schema name'}
external_source <-
external_source_relationship.relating_source <-
external_source_relationship
{external_source_relationship.name = 'composition'}
external_source_relationship.related_source
external_source
{/DESCRIPTION_ASSGN(external_source)/
description_attribute.attribute_value = 'entity type' }
external_source.source_id
{-> source_item
source_item = identifier}

5.1.6.5.2 schema_name

AIM element: external_source.source_id
Source: ISO 10303-41
Reference path: applied_external_identification_assignment <=
external_identification_assignment
external_identification_assignment.source ->
external_source
{/DESCRIPTION_ASSGN(external_source)/
description_attribute.attribute_value = 'schema name'}
external_source.source_id
{ -> source_item
source_item = identifier}

5.1.6.5.3 external_instance_reference to global_id (as target_guid)

AIM element: PATH
Reference path: applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.assigned_id

5.1.6.6 EXTERNAL_REFERENCE

AIM element: applied_external_identification_assignment
Source: ISO 10303-218
Reference path: applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.role ->
identificaton_role
identification_role.name ='external reference'}

5.1.6.6.1 description

AIM element: identification_role.description
Source: ISO 10303-41
Rules: 5.2.4.66
Reference path: applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.role ->
identificaton_role
identification_role.description

5.1.6.6.2 external_reference to external_storage (as location)

AIM element: PATH
Reference path: applied_external_identification_assignment <=
external_identification_assignment
external_identification_assignment.source ->
external_source
{/CLASS_ID(external_source,'external storage')/}

5.1.6.6.3 external_reference to universal_resource_locator (as location)

AIM element: PATH
Reference path: applied_external_identification_assignment <=
external_identification_assignment
external_identification_assignment.source ->
external_source
{/CLASS_ID(external_source,'universal resource locator')/}

5.1.6.7 EXTERNAL_STORAGE

AIM element: external_source
Source: ISO 10303-41
Reference path: {/ROOT_CLASS(external_source, 'external storage')/}

5.1.6.7.1 location

AIM element: external_source.source_id
Source: ISO 10303-41
Reference path: external_source
external_source.source_id
{-> source_item
source_item = identifier}

5.1.6.8 UNIVERSAL_RESOURCE_LOCATOR

AIM element: external_source
Source: ISO 10303-41
Reference path: {/ROOT_CLASS(external_source, 'universal resource locator')/}

5.1.6.8.1 location

AIM element: external_source.source_id
Source: ISO 10303-41
Reference path: external_source
external_source.source_id
{-> source_item
source_item = identifier}

5.1.7 hull_cross_sections UoF

5.1.7.1 FLARE_AREA

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.330
Reference path: {/ROOT_CLASS(shape_aspect, 'flare area')/}

5.1.7.1.1 area

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.269, 5.2.4.53
Reference path: shape_aspect <-
/PDR_NAME('flare area parameters')/ ->
representation
representation.items [i] ->

/VAL_REP_ITEM('area', area_measure)/

5.1.7.1.2 height

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.269, 5.2.4.53
Reference path: shape_aspect <-
/PDR_NAME('flare area parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('height', positive_length_measure)/

5.1.7.2 HULL_CROSS_SECTION

AIM element: [product_definition]
[group]
Source: ISO 10303-41
Rules: 5.2.4.59, 5.2.4.89 ,5.2.4.63 5.2.4.64
Reference path: {[/CLASS(product_definition, 'hull cross section', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]
[/CLASS(product_definition, 'hull cross section', 'item structure')/]
[/CLASS(product_definition, 'item structure', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]
[/CLASS(product_definition, 'item structure', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]
[product_definition
/LINK_TO_GROUP(product_definition)/]}

5.1.7.2.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.7.2.2 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.7.2.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.7.2.4 hull_cross_section to external_instance_reference (as external_-items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition,'item structure')/
{(/EXT_INST_REF(product_definition,'ship structures schema', 'compartment')/)
(/EXT_INST_REF(product_definition,'ship structures schema', 'structural system')/)}

5.1.7.2.5 hull_cross_section to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
product_definition = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
Document
{/CLASS_ID(‘document reference with address’)/})

5.1.7.2.6 hull_cross_section to item (as items)

AIM element: PATH
Rules: 5.2.4.59
Reference path: group <-
/GROUPS(product_definition,'item structure')/
{/CLASS_ID(product_definition,'structural system')/}

5.1.7.2.7 hull_cross_section to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition
identification_item = product_definition <-

applied_identification_assignment.items[i]
applied_identification_assignment

5.1.7.2.8 hull_cross_section to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation->
product_definition_formation
{product_definition_formation.id= 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product,'ship')/}

5.1.7.3 HULL_CROSS_SECTION_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.61, 5.2.4.62 5.2.4.108
Reference path: {[/CLASS(product_definition_shape, 'hull cross section design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition','definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.7.3.1 description

AIM element: product_definition_shape.description
Source: ISO 10303-41

5.1.7.3.2 mirrored_symmetry

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.292, 5.2.4.60
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('hull cross section design definition parameters')/ ->
representation
representation.items [i] ->
descriptive_representation_item
{descriptive_representation_item.name = 'mirrored symmetry'}
{(descriptive_representation_item.description = 'TRUE')
(descriptive_representation_item.description = 'FALSE')}

5.1.7.3.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.7.3.4 hull_cross_section_design_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.7.3.5 hull_cross_section_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition = shape_aspect
represented_definition <-
/SDR_NAME('hull cross section shape representation')/
property_definition_representation.used_representation ->
representation
{representation.name = 'representations'} =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.7.3.6 hull_cross_section_design_definition to flare_area (as flare_-area_buckling)

AIM element: PATH
Reference path: product_definition_shape <-
shape_aspect.of_shape
{/CLASS_ID(shape_aspect, 'flare area')/}

5.1.7.3.7 hull_cross_section_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-

applied_identification_assignment.items[i]
applied_identification_assignment

5.1.7.3.8 hull_cross_section_design_definition to hull_cross_section (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'hull cross section')/}

5.1.7.3.9 hull_cross_section_design_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.7.3.10 hull_cross_section_design_definition to non_manifold_surface_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition = shape_aspect
represented_definition <-
/SDR_NAME('hull cross section shape representation')/
property_definition_representation.used_representation ->
representation
{representation.name = 'representations'} =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.8 hull_class_applicability UoF

5.1.8.1 HULL_APPLICABILITY

AIM element: serial_numbered_effectivity
Source: ISO 10303-41

5.1.8.1.1 end_hull

AIM element: serial_numbered_effectivity.effectivity_end_id
Source: ISO 10303-41

5.1.8.1.2 start_hull

AIM element: serial_numbered_effectivity.effectivity_start_id
Source: ISO 10303-41

5.1.8.1.3 hull_applicability to definition (as definitions_for_hulls)

AIM element: PATH
Reference path: serial_numbered_effectivity <=
effectivity
{effectivity.id = 'hull applicability'}
effectivity <-
effectivity_assignment.assigned_effectivity
effectivity_assignment
/ROLE_ASSGN(effectivity_assignment)/
{object_role.name = 'definitions for hulls'}
effectivity_assignment =>
applied_effectivity_assignment
applied_effectivity_assignment.items[i] ->
effectivity_item
(effectivity_item = product_definition
{CLASS_ID(product_definition, 'definition')})
(effectivity_item = property_definition
{CLASS_ID(property_definition, 'definition')})
(effectivity_item = product_definition_shape
{CLASS_ID(product_definition_shape, 'definition')})
(effectivity_item = product_related_product_category
{CLASS_ID(product_related_product_category, 'definition')})

5.1.8.1.4 hull_applicability to item (as items_for_hulls)

AIM element: PATH
Reference path: serial_numbered_effectivity <=
effectivity
{effectivity.id = 'hull applicability'}
effectivity <-
effectivity_assignment.assigned_effectivity
effectivity_assignment
/ROLE_ASSGN(effectivity_assignment)/
{object_role.name = 'items for hulls'}
effectivity_assignment =>
applied_effectivity_assignment
applied_effectivity_assignment.items[i] ->
effectivity_item

(effectivity_item = product_definition
{CLASS_ID(product_definition, ‘item’)})
(effectivity_item = property_definition
{CLASS_ID(property_definition, ‘item’)})
(effectivity_item = shape_aspect
{CLASS_ID(shape_aspect, ‘item’)})

5.1.9 items UoF

5.1.9.1 DEFINABLE_OBJECT

#1: if definable_object is an item
#2: if definable_object is an item_relationship
#3: if definable_object is an item_structure

AIM element: #1: /SUBTYPE(item)/ (See 5.1.9.3)
#2: /SUBTYPE(item_relationship)/ (See 5.1.9.4)
#3: /SUBTYPE(item_structure)/ (See 5.1.9.5)

5.1.9.1.1 definable_object to global_id (as id)

AIM element: #1: /SUBTYPE(item)/ (See 5.1.9.3)
#2: /SUBTYPE(item_relationship)/ (See 5.1.9.4)
#3: /SUBTYPE(item_structure)/ (See 5.1.9.5)

5.1.9.2 GLOBAL_ID

AIM element: applied_identification_assignment
Source: ISO 10303-218
Reference path: applied_identification_assignment <=
identification_assignment
{identification_assignment.role->
identification_role
identification_role.name = ’globally unambiguous identifier’}

5.1.9.2.1 id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Reference path: applied_identification_assignment <=
identification_assignment
identification_assignment.assigned_id

5.1.9.3 ITEM

#1: if item is an assembly
#2: if item is a change
#3: if item is a feature
#4: if item is a hull cross section
#5: if item is a part
#6: if item is a ship
#7: if item is a structural part connection implementation
#8: if item is a system

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1)
#2: /SUBTYPE(change)/ (See 5.1.3.4)
#3: /SUBTYPE(feature)/ (See 5.1.12.5)
#4: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2)
#5: /SUBTYPE(part)/ (See 5.1.12.8)
#6: /SUBTYPE(ship)/ (See 5.1.9.6)
#7: /SUBTYPE(structural_part_connection_implementation)/ (See 5.1.21.9)
#8: /SUBTYPE(system)/ (See 5.1.12.10)

5.1.9.3.1 description

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.1)
#2: /SUBTYPE(change)/ (See 5.1.3.4.1)
#3: /SUBTYPE(feature)/ (See 5.1.12.5.1)
#4: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.1)
#5: /SUBTYPE(part)/ (See 5.1.12.8.1)
#6: /SUBTYPE(ship)/ (See 5.1.9.6.1)
#7: /SUBTYPE(structural_part_connection_implementation)/ (See 5.1.21.9.1)
#8: /SUBTYPE(system)/ (See 5.1.12.10.1)

5.1.9.3.2 name

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.2)
#2: /SUBTYPE(change)/ (See 5.1.3.4.2)
#3: /SUBTYPE(feature)/ (See 5.1.12.5.2)
#4: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.2)
#5: /SUBTYPE(part)/ (See 5.1.12.8.2)
#6: /SUBTYPE(ship)/ (See 5.1.9.6.2)
#7: /SUBTYPE(structural_part_connection_implementation)/ (See 5.1.21.9.2)
#8: /SUBTYPE(system)/ (See 5.1.12.10.2)

5.1.9.3.3 item to external_reference (as documentation)

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.5)
#2: /SUBTYPE(change)/ (See 5.1.3.4.4)
#3: /SUBTYPE(feature)/ (See 5.1.12.5.3)
#4: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.5)
#5: /SUBTYPE(part)/ (See 5.1.12.8.3)

#6: /SUBTYPE(ship)/ (See 5.1.9.6.3)
#7: /SUBTYPE(structural_part_connection_implementation)/ (See 5.1.21.9.4)
#8: /SUBTYPE(system)/ (See 5.1.12.10.4)

5.1.9.3.4 item to global_id (as id)

NOTE Attribute inherited from supertype Definable_object(see 5.1.9.1).

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.8)
#2: /SUBTYPE(change)/ (See 5.1.3.4.5)
#3: /SUBTYPE(feature)/ (See 5.1.12.5.4)
#4: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.7)
#5: /SUBTYPE(part)/ (See 5.1.12.8.4)
#6: /SUBTYPE(ship)/ (See 5.1.9.6.4)
#7: /SUBTYPE(structural_part_connection_implementation)/ (See 5.1.21.9.6)
#8: /SUBTYPE(system)/ (See 5.1.12.10.7)

5.1.9.3.5 item to ship (as ship_context)

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.11)
#2: /SUBTYPE(change)/ (See 5.1.3.4.6)
#3: /SUBTYPE(feature)/ (See 5.1.12.5.5)
#4: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.8)
#5: /SUBTYPE(part)/ (See 5.1.12.8.5)
#7: /SUBTYPE(structural_part_connection_implementation)/ (See 5.1.21.9.5)
#8: /SUBTYPE(system)/ (See 5.1.12.10.10)

5.1.9.4 ITEM_RELATIONSHIP

#1: if item relationship is an assembly relationship
#2: if item relationship is a feature relationship
#3: if item relationship is a part relationship
#4: if item relationship is a structural part joint
#5: if item relationship is a structural part penetration relationship
#6: if item relationship is a structural system penetration relationship
#7: if item relationship is a system relationship

AIM element: #1: /SUBTYPE(assembly_relationship)/ (See 5.1.12.2)
#2: /SUBTYPE(feature_relationship)/ (See 5.1.12.7)
#3: /SUBTYPE(part_relationship)/ (See 5.1.12.9)
#4: /SUBTYPE(structural_part_joint)/ (See 5.1.21.10)
#5: /SUBTYPE(structural_part_penetration_relationship)/ (See 5.1.18.37)
#6: /SUBTYPE(structural_system_penetration_relationship)/ (See 5.1.18.38)
#7: /SUBTYPE(system_relationship)/ (See 5.1.12.12)

5.1.9.4.1 description

AIM element: #1: /SUBTYPE(assembly_relationship)/ (See 5.1.12.2.1)
#2: /SUBTYPE(feature_relationship)/ (See 5.1.12.7.1)
#3: /SUBTYPE(part_relationship)/ (See 5.1.12.9.1)
#4: /SUBTYPE(structural_part_joint)/ (See 5.1.21.10.1)
#5: /SUBTYPE(structural_part_penetration_relationship)/ (See 5.1.18.37.1)
#6: /SUBTYPE(structural_system_penetration_relationship)/ (See 5.1.18.38.1)
#7: /SUBTYPE(system_relationship)/ (See 5.1.12.12.1)

5.1.9.4.2 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: #1: /SUBTYPE(assembly_relationship)/ (See 5.1.12.2.2)
#2: /SUBTYPE(feature_relationship)/ (See 5.1.12.7.2)
#3: /SUBTYPE(part_relationship)/ (See 5.1.12.9.2)
#4: /SUBTYPE(structural_part_joint)/ (See 5.1.21.10.9)
#5: /SUBTYPE(structural_part_penetration_relationship)/ (See 5.1.18.37.2)
#6: /SUBTYPE(structural_system_penetration_relationship)/ (See 5.1.18.38.2)
#7: /SUBTYPE(system_relationship)/ (See 5.1.12.12.2)

5.1.9.4.3 item_relationship to external_instance_reference (as external_-item_1)

AIM element: #1: /SUBTYPE(assembly_relationship)/ (See 5.1.12.2.3)
#2: /SUBTYPE(feature_relationship)/ (See 5.1.12.7.3)
#3: /SUBTYPE(part_relationship)/ (See 5.1.12.9.3)
#4: /SUBTYPE(structural_part_joint)/ (See 5.1.21.10.5)
#5: /SUBTYPE(structural_part_penetration_relationship)/ (See 5.1.18.37.3)
#6: /SUBTYPE(structural_system_penetration_relationship)/ (See 5.1.18.38.3)
#7: /SUBTYPE(system_relationship)/ (See 5.1.12.12.3)

5.1.9.4.4 item_relationship to external_instance_reference (as external_-item_2)

AIM element: #1: /SUBTYPE(assembly_relationship)/ (See 5.1.12.2.4)
#2: /SUBTYPE(feature_relationship)/ (See 5.1.12.7.4)
#3: /SUBTYPE(part_relationship)/ (See 5.1.12.9.4)
#4: /SUBTYPE(structural_part_joint)/ (See 5.1.21.10.7)
#5: /SUBTYPE(structural_part_penetration_relationship)/ (See 5.1.18.37.4)
#6: /SUBTYPE(structural_system_penetration_relationship)/ (See 5.1.18.38.4)
#7: /SUBTYPE(system_relationship)/ (See 5.1.12.12.4)

5.1.9.4.5 item_relationship to global_id (as id)

NOTE Attribute inherited from supertype 5.1.9.1

AIM element: #1: /SUBTYPE(assembly_relationship)/ (See 5.1.12.2.5)
#2: /SUBTYPE(feature_relationship)/ (See 5.1.12.7.5)
#3: /SUBTYPE(part_relationship)/ (See 5.1.12.9.5)
#4: /SUBTYPE(structural_part_joint)/ (See 5.1.21.10.8)
#5: /SUBTYPE(structural_part_penetration_relationship)/ (See 5.1.18.37.5)
#6: /SUBTYPE(structural_system_penetration_relationship)/ (See 5.1.18.38.5)
#7: /SUBTYPE(system_relationship)/ (See 5.1.12.12.5)

5.1.9.4.6 item_relationship to item (as item_1)

AIM element: #1: /SUBTYPE(assembly_relationship)/ (See 5.1.12.2.6)
#2: /SUBTYPE(feature_relationship)/ (See 5.1.12.7.6)
#3: /SUBTYPE(part_relationship)/ (See 5.1.12.9.6)
#4: /SUBTYPE(structural_part_joint)/ (See 5.1.21.10.4)
#5: /SUBTYPE(structural_part_penetration_relationship)/ (See 5.1.18.37.6)
#6: /SUBTYPE(structural_system_penetration_relationship)/ (See 5.1.18.38.6)
#7: /SUBTYPE(system_relationship)/ (See 5.1.12.12.6)

5.1.9.4.7 item_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(assembly_relationship)/ (See 5.1.12.2.7)
#2: /SUBTYPE(feature_relationship)/ (See 5.1.12.7.7)
#3: /SUBTYPE(part_relationship)/ (See 5.1.12.9.7)
#4: /SUBTYPE(structural_part_joint)/ (See 5.1.21.10.6)
#5: /SUBTYPE(structural_part_penetration_relationship)/ (See 5.1.18.37.7)
#6: /SUBTYPE(structural_system_penetration_relationship)/ (See 5.1.18.38.7)
#7: /SUBTYPE(system_relationship)/ (See 5.1.12.12.7)

5.1.9.5 ITEM_STRUCTURE

#1: if item structure is an assembly
#2: if item structure is a hull cross section
#3: if item structure is a system

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1)
#2: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2)
#3: /SUBTYPE(system)/ (See 5.1.12.10)

5.1.9.5.1 description

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.1)
#2: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.1)
#3: /SUBTYPE(system)/ (See 5.1.12.10.1)

5.1.9.5.2 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.4)
#2: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.3)
#3: /SUBTYPE(system)/ (See 5.1.12.10.3)

5.1.9.5.3 item_structure to external_instance_reference (as external_items)

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.6)
#2: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.4)
#3: /SUBTYPE(system)/ (See 5.1.12.10.5)

5.1.9.5.4 item_structure to external_instance_reference (as external_-relationships)

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.7)
#2: /SUBTYPE(hull_cross_section)/ (See note)
#3: /SUBTYPE(system)/ (See 5.1.12.10.6)

NOTE For this subtype the attribute has been re-declared in the ARM to be a set of zero, which is interpreted as the attribute is not required

5.1.9.5.5 item_structure to global_id(as id)

NOTE Attribute inherited from supertype Definable_object(see 5.1.9.1).

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.8)
#2: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.7)
#3: /SUBTYPE(system)/ (See 5.1.12.10.7)

5.1.9.5.6 item_structure to item (as items)

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.9)
#2: /SUBTYPE(hull_cross_section)/ (See 5.1.7.2.6)
#3: /SUBTYPE(system)/ (See 5.1.12.10.8)

5.1.9.5.7 item_structure to item_relationship (as relationships)

AIM element: #1: /SUBTYPE(assembly)/ (See 5.1.12.1.10)
#2: /SUBTYPE(hull_cross_section)/ (See note)
#3: /SUBTYPE(system)/ (See 5.1.12.10.9)

NOTE For this subtype the attribute has been re-declared in the ARM to be a set of zero, which is interpreted as the attribute is not required

5.1.9.6 SHIP

AIM element: product
Source: ISO 10303-41
Reference path: {[/CLASS(product, 'ship', 'item')/]
[/CLASS(product, 'item', 'definable object')/]
[/ROOT_CLASS(product, 'definable object')/]}

5.1.9.6.1 description

NOTE Attribute inherited from supertype Item (see 5.1.9.3).

AIM element: product.description
Source: ISO 10303-41

5.1.9.6.2 name

NOTE Attribute inherited from supertype Item (see 5.1.9.3).

AIM element: product.name
Source: ISO 10303-41

5.1.9.6.3 ship to external_reference (as documentation)

NOTE Attribute inherited from supertype Item (see 5.1.9.3).

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: product
product = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: product
/DOC_REF(product,'documentation')/
document
{/CLASS_ID(document, 'document reference with address')/}

5.1.9.6.4 ship to global_id (as id)

NOTE Attribute inherited from supertype Definable_object (see 5.1.9.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.146
Reference path: product

identification_item = product <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.9.6.5 ship to item (as ship_items)

#1: if the item is an assembly, a hull_cross_section, a structural_part, a structural_system, or a structural_part_connection_implementation
#2: if the item is a change
#3: if the item is a structural_feature

AIM element: #1: PATH
#2: PATH
#3: PATH

Reference path: #1: product <-
product_definition_formation.of_product
product_definition_formation <-
product_definition.formation
product_definition

#2: product
action_item = product
action_item <-
applied_action_assignment.items[i]
applied_action_assignment
applied_action_assignment.assigned_action ->
action

#3: product <-
product_definition_formation.of_product
product_definition_formation <-
product_definition.formation
product_definition
characterized_product_definition = product_definition
characterized_definition = characterized_product_definition
characterized_defintion <-
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect.of shape
shape_aspect

5.1.9.6.6 single_hull_or_class

AIM element: product_context
Source: ISO 10303-41
Reference path: product
product.frame_of_reference ->
product_context
{(product_context.discipline_type = 'design for single hull')
(product_context.discipline_type = 'design for multiple hulls')}

5.1.9.6.7 ship to derived_unit (as units)

AIM element: PATH
Reference path: product <-
product_definition_formation.of_product
product_definition_formation<-
product_definition.formation
/PROD_DEF_TO_UNITS('global units')/
unit
unit = derived_unit
derived_unit

5.1.9.6.8 ship to named_unit (as units)

AIM element: PATH
Reference path: product <-
product_definition_formation.of_product
product_definition_formation<-
product_definition.formation
/PROD_DEF_TO_UNITS('global units')/
unit
unit = named_unit
named_unit

5.1.9.7 VERSIONABLE_OBJECT

#1: if versionable object is a definition
#2: if versionable object is a document
#3: if versionable object is an item relationship
#4: if versionable object is an item structure
#5: if versionable object is a revision

AIM element: #1: /SUBTYPE(definition)/ (See 5.1.4.1)
#2: /SUBTYPE(document)/ (See 5.1.6.1)
#3: /SUBTYPE(item_relationship)/ (See 5.1.9.4)
#4: /SUBTYPE(item_structure)/ (See 5.1.9.5)
#5: /SUBTYPE(revision)/ (See 5.1.3.13)

5.1.9.7.1 version_id

AIM element: #1: /SUBTYPE(definition)/ (See 5.1.4.1.2)
#2: /SUBTYPE(document)/ (See 5.1.6.1.5)
#3: /SUBTYPE(item_relationship)/ (See 5.1.9.4.2)
#4: /SUBTYPE(item_structure)/ (See 5.1.9.5.2)
#5: /SUBTYPE(revision)/ (See 5.1.3.13.3)

5.1.10 library_reference UoF

5.1.10.1 BSU

#1: BSU is a Class_BSU
#2: BSU is a Property_BSU
#3: BSU is a Supplier_BSU

AIM element: #1: /SUBTYPE(class_bsu)/ (See 5.1.10.3)
#2: /SUBTYPE(property_bsu)/ (See 5.1.10.8)
#2: /SUBTYPE(supplier_bsu)/ (See 5.1.10.12)

5.1.10.1.1 code

AIM element: #1: /SUBTYPE(class_bsu)/ (See 5.1.10.3.1)
#2: /SUBTYPE(property_bsu)/ (See 5.1.10.8.1)
#2: /SUBTYPE(supplier_bsu)/ (See 5.1.10.12.1)

5.1.10.2 BOOLEAN_PROPERTY_VALUE

AIM element: representation
Source: ISO 10303-43
Reference path: representation
{representation.name = ‘property value’}

5.1.10.2.1 val

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Reference path: representation
representation.items[i] ->
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description = ‘TRUE‘)
(descriptive_representation_item.description = ‘FALSE‘)}

5.1.10.2.2 boolean_property_value to property_bsu (as property_-identifier)

AIM element: PATH
Reference path: /SUPERTYPE(property_value)/ (See 5.1.10.9.1)

5.1.10.3 CLASS_BSU

AIM element: externally_defined_class
Source: ISO 10303-218
Rule: 5.2.4.313
Reference path: [externally_defined_class <=
class <=
group]
[externally_defined_class <=
externally_defined_item
{externally_defined_item.source ->
external_source =>
known_source <=
pre_defined_item
pre_defined_item.name = 'ISO 13584 library'}]

5.1.10.3.1 code

AIM element: externally_defined_item.item_id
Source: ISO 10303-41
Reference path: externally_defined_class <=
externally_defined_item
externally_defined_item.item_id

5.1.10.3.2 version

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Reference path: externally_defined_class
externally_defined_class = identification_item
identification_item <-
library_class_version_assignment.items[i]
library_class_version_assignment <=
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.assigned_id
{/ID_ROLE('class version')/}

5.1.10.3.3 class_bsu to supplier_bsu (as delivered_by)

AIM element: PATH
Reference path: externally_defined_class <=
externally_defined_item
externally_defined_item.source ->
external_source =>
/ORG_ASSGN(known_source, 'library supplier')/

5.1.10.4 INTEGER_PROPERTY_VALUE

AIM element: representation
Source: ISO 10303-43
Reference path: representation
{representation.name = 'property value'}

5.1.10.4.1 val

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('.UNUSED.', numeric_measure)/

5.1.10.4.2 integer_property_value to property_bsu (as property_identifier)

AIM element: PATH
Reference path: /SUPERTYPE(property_value)/ (See 5.1.10.9.1)

5.1.10.5 LIBRARY_ELEMENT_REFERENCE

AIM element: applied_classification_assignment
Source: ISO 10303-218

5.1.10.5.1 library_element_reference to class_bsu (as library_identifier)

AIM element: PATH
Reference path: applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'definitional class membership'}
classification_assignment.assigned_classification ->
group =>
{group.name = 'library identifier'}
class =>
externally_defined_class

5.1.10.5.2 library_element_reference to property_value (as property_-value_pairs)

AIM element: PATH
Reference path: applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'definitional class membership'}
classification_assignment.assigned_classification ->
group =>
{group.name = 'library identifier'}
class =>
externally_defined_class <=
externally_defined_item <-
externally_defined_item_relationship.related_item
{externally_defined_item_relationship.name = ‘name scope’}
externally_defined_item_relationship
externally_defined_item_relationship.relating_item
externally_defined_item =>
externally_defined_general_property <=
general_property <-
general_property_association.base_definition
general_property_association
{general_property_association.name = 'definitional'}
general_property_association.derived_definition->
property_definition <-
property_definition_representation.definition
property_definition_representation
property_definition_representation.used_representation->
representation
{representation.name = 'property value'}

5.1.10.6 LOGICAL_PROPERTY_VALUE

AIM element: representation
Source: ISO 10303-43
Reference path: representation
{representation.name = ‘property value’}

5.1.10.6.1 val

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Reference path: representation
representation.items[i] ->
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description = ‘TRUE‘)

(descriptive_representation_item.description = ‘FALSE‘)
(descriptive_representation_item.description = ‘UNKNOWN‘)}

5.1.10.6.2 logical_property_value to property_bsu (as property_identifier)

AIM element: PATH
Reference path: /SUPERTYPE(property_value)/ (See 5.1.10.9.1)

5.1.10.7 NUMBER_PROPERTY_VALUE

AIM element: representation
Source: ISO 10303-43
Reference path: representation
{representation.name = 'property value'}

5.1.10.7.1 val

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('.UNUSED.', numeric_measure)/

5.1.10.7.2 number_property_value to property_bsu (as property_-identifier)

AIM element: PATH
Reference path: /SUPERTYPE(property_value)/ (See 5.1.10.9.1)

5.1.10.8 PROPERTY_BSU

AIM element: externally_defined_general_property
Source: ISO 10303-218
Rule: 5.2.4.313
Reference path: [externally_defined_general_property <=
general_property]
[externally_defined_general_property <=
externally_defined_item
{externally_defined_item.source ->
external_source =>
known_source <=
pre_defined_item
pre_defined_item.name = 'ISO 13584 library'}]

5.1.10.8.1 code

AIM element: externally_defined_item.item_id
Source: ISO 10303-41
Reference path: externally_defined_general_property <=
externally_defined_item
externally_defined_item.item_id

5.1.10.8.2 version

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Reference path: externally_defined_general_property
externally_defined_general_property = identification_item
identification_item <-
library_property_version_assignment.items[i]
library_property_version_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.assigned_id
{/ID_ROLE('property version')/}

5.1.10.8.3 property_bsu to class_bsu (as name_scope)

AIM element: PATH
Reference path: externally_defined_general_property <=
externally_defined_item <-
externally_defined_item_relationship.relating_item
externally_defined_item_relationship
{externally_defined_item_relationship.name = 'name scope'}
externally_defined_item_relationship.related_item ->
externally_defined_item =>
externally_defined_class

5.1.10.9 PROPERTY_VALUE

AIM element: representation
Source: ISO 10303-43
Reference path:
representation
{representation.name = 'property value'}

5.1.10.9.1 property_value to property_bsu (as property_identifier)

AIM element: PATH
Reference path: representation <-
property_definition_representation.used_representation
property_definition_representation
property_definition_representation.definition ->
property_definition <-
general_property_association.derived_definition
general_property_association
{general_property_association.name = 'definitional'}
general_property_association.base_definition ->
general_property =>
externally_defined_general_property

5.1.10.10 REAL_PROPERTY_VALUE

AIM element: representation
Source: ISO 10303-43
Reference path:
representation
{representation.name = 'property value'}

5.1.10.10.1 val

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: representation
representation.items[i] ->
VAL_REP_ITEM< '.UNUSED.', numeric_measure >

5.1.10.10.2 real_property_value to property_bsu (as property_identifier)

AIM element: PATH
Reference path: /SUPERTYPE(property_value)/ (See 5.1.10.9.1)

5.1.10.11 STRING_PROPERTY_VALUE

AIM element: representation
Source: ISO 10303-43
Reference path: representation
{representation.name = 'property value'}

5.1.10.11.1 val

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Reference path: representation
representation.items[i] ->
representation_item =>
descriptive_representation_item
descriptive_representation_item.description

5.1.10.11.2 string_property_value to property_bsu (as property_identifier)

AIM element: PATH
Reference path: /SUPERTYPE(property_value)/ (See 5.1.10.9.1)

5.1.10.12 SUPPLIER_BSU

AIM element: organization
Source: ISO 10303-41
Rule: 5.2.4.52, 5.2.4.313
Reference path: externally_defined_class < =
{externally_defined_class < =
externally_defined_item
externally_defined_item.source - >
external_source = >
known_source < =
pre_defined_item
pre_defined_item.name = 'ISO 13584 library'}
class
organization_item = class
organization_item < -
applied_organization_assignment. items[i]
applied_organization_assignment < =
organization_assignment
{organization_assignment.role - >
organization_role
organization_role.name = 'library supplier'}}
organization_assignment.assigned_ organization - >
organization

5.1.10.12.1 code

AIM element: organization.id
Source: ISO 10303-41

5.1.11 location_concepts UoF

5.1.11.1 BUTTOCK_TABLE

AIM element: property_definition
Source: ISO 10303-41
Reference path: {[/CLASS(property_definition, 'buttock table', 'transversal table')/]
[/CLASS(property_definition, 'transversal table', 'spacing table')/]
[/CLASS(property_definition, 'spacing table', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.11.1.1 description

NOTE Attribute inherited from supertype Spacing_table(see 5.1.11.10).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.1

5.1.11.1.2 name

NOTE Attribute inherited from supertype Spacing_table(see 5.1.11.10).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.2

5.1.11.1.3 version_id

NOTE Attribute inherited from supertype Versionable_object)see 5.1.9.7).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.3

5.1.11.1.4 buttock_table to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.4

5.1.11.1.5 buttock_table to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.5

5.1.11.1.6 buttock_table to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.6

5.1.11.1.7 buttock_table to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.7

5.1.11.1.8 buttock_table to transversal_position (as spacing_table_-representations)

NOTE Attribute inherited from supertype Transversal_table (see 5.1.11.13).

AIM element: /SUPERTYPE(transversal_table)/5.1.11.13.8

5.1.11.2 FRAME_TABLE

AIM element: property_definition
Source: ISO 10303-41
Reference path: {[/CLASS(property_definition, 'frame table', 'longitudinal table')/]
[/CLASS(property_definition, 'longitudinal table', 'spacing table')/]
[/CLASS(property_definition, 'spacing table', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.11.2.1 description

NOTE Attribute inherited from supertype Spacing_table (see 5.1.11.10).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.1

5.1.11.2.2 name

NOTE Attribute inherited from supertype Spacing_table (see 5.1.11.10).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.2

5.1.11.2.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.3

5.1.11.2.4 frame_table to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.4

5.1.11.2.5 frame_table to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.5

5.1.11.2.6 frame_table to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.6

5.1.11.2.7 frame_table to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.7

5.1.11.2.8 frame_table to longitudinal_position (as spacing_table_-representations)

NOTE Attribute inherited from supertype Longitudinal_table (see 5.1.11.7).

AIM element: /SUPERTYPE(longitudinal_table)/5.1.11.7.8

5.1.11.3 GLOBAL_AXIS_PLACEMENT

AIM element: [product_definition]

Source: ISO 10303-41, ISO 10303-42
Rules: 5.2.4.55, 5.2.4.193, 5.2.4.215
Reference path:
{[/CLASS(product_definition, ‘global axis placement’, ‘general characteristics definition’)/]
[/CLASS(product_definition, ‘general characteristics definition’, ‘definition’)/]
[/CLASS(product_definition, ‘definition’, ‘versionable object’)/]
[/ROOT_CLASS(product_definition, ‘versionable object’)/]}
/PROD_DEF_TO_REP(‘global axis placement’)/
representation
{representation.name = ‘global axis representation‘}
representation.context_of_items
representation_context =>
{representation_context.context_type = ‘global co_ordinate space’}
geometric_representation_context
{geometric_representation_context.co_ordinate_space_dimension = 3}
/REP_ITEM(’global axes and origin’)/
geometric_representation_item =>
placement =>

axis2_placement_3d]

5.1.11.3.1 after_perpendicular_offset

AIM element: value_representation_item.value_component
Source: ISO 10303-41
Rules: 5.2.4.193, 5.2.4.215
Reference path: /PROD_DEF_TO_REP('global axis placement')/
representation
{representation.name = 'global axis representation'}
representation.context_of_items
representation_context =>
{representation_context.context_type = 'global co_ordinate space'}
geometric_representation_context
/REP_TO_VAL_REP_ITEM('after perpendicular offset', length_measure)/

5.1.11.3.2 description

NOTE Attribute inherited from supertype Item (see 5.1.9.3)

AIM element: product_definition.description
Source: ISO 10303-41

5.1.11.3.3 orientation

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.193, 5.2.4.215
Reference path: /PROD_DEF_TO_REP('global axis placement')/
representation
{representation.name = 'global axis representation'}
representation.context_of_items
representation_context =>
{representation_context.context_type = 'global co_ordinate space'}
geometric_representation_context
/REP_ITEM('orientation')/
descriptive_representation_item
descriptive_representation_item.description
{(descriptive_representation_item.description = 'forward pointing')
(descriptive_representation_item.description = 'aft pointing')}

5.1.11.3.4 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369

Reference path:
/VERSION_ID(product_definition)/

5.1.11.3.5 global_axis_placement to ship (as defined_for)

NOTE Attribute inherited from supertype General_characteristics_definition (see 5.1.4.4).

AIM element: PATH
Reference path: /PROD_DEF_PRODUCT/
{/CLASS_ID(product, 'ship')/}

5.1.11.3.6 global_axis_placement to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.11.3.7 global_axis_placement to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROD_DEF_TO_UNITS('local_units')/
unit
unit = derived_unit
derived_unit

5.1.11.3.8 global_axis_placement to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROD_DEF_TO_UNITS('local_units')/
unit
unit = named_unit
nameed_unit

5.1.11.4 LOCAL_CO_ORDINATE_SYSTEM

NOTE Link to context and definitions.

AIM element: [property_definition]

Source: ISO 10303-41, ISO 10303-42
Rules: 5.2.4.215, 5.2.4.157, 5.2.4.196
Reference path: {[/CLASS(property_definition, 'local coordinate system', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')]
[/ROOT_CLASS(property_definition, 'versionable object')/]}
/PROP_DEF_REP_HELP('local coordinate system')/
representation
{representation.name = 'local axis representation'}
representation.context_of_items
representation_context =>
{representation_context.context_type = 'local coordinate space'}
geometric_representation_context
{geometric_representation_context.coordinate_space_dimension = 3}
/REP_ITEM('local axes and origin')/
geometric_representation_item =>
placement =>
axis2_placement_3d

5.1.11.4.1 description

NOTE Attribute inherited from supertype Item (see 5.1.9.3)

AIM element: property_definition.description
Source: ISO 10303-41

5.1.11.4.2 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.11.4.3 local_co_ordinate_system to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: [/PROP_TO_PROD_DEF/]
[property_definition
property_definition.definition ->
characterized_definition

characterized_definition = shape_definition
shape_definition = shape_aspect]

5.1.11.4.4 local_co_ordinate_system to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: property_definition
identification_item = property_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.11.4.5 local_co_ordinate_system to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROP_DEF_TO_UNITS('local_units')/
unit
unit = derived_unit
derived_unit

5.1.11.4.6 local_co_ordinate_system to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROP_DEF_TO_UNITS('local_units')/
unit
unit = named_unit
named_unit

5.1.11.4.7 local_co_ordinate_system to global_axis_placement (as parent)

AIM element: PATH
Rules: 5.2.4.220
Reference path: axis2_placement_3d <=
placement <=
geometric_representation_item <=
representation_item <-
representation_map.mapping_origin
representation_map
{[representation_map.mapped_representation ->
representation
representation.name = 'local axis representation']}
representation_map <-
mapped_item.mapping_source

mapped_item
{mapped_item.name = ‘local coordinate system position in global coordinate system’}
mapped_item.mapping_target ->
representation_item =>
geometric_representation_item =>
placement =>
axis2_placement_3d

5.1.11.4.8 local_co_ordinate_system to local_co_ordinate_system (as parent)

AIM element: PATH
Rules: 5.2.4.220
Reference path: axis2_placement_3d <=
placement <=
geometric_representation_item <=
representation_item <-
representation_map.mapping_origin
representation_map
{[representation_map.mapped_representation ->
representation
representation.name = ‘local axis representation’]}
representation_map <-
mapped_item.mapping_source
mapped_item
{mapped_item.name = ‘local coordinate system position in parent local coordinate system’}
mapped_item.mapping_target ->
representation_item =>
geometric_representation_item =>
placement =>
axis2_placement_3d

5.1.11.4.9 local_co_ordinate_system to local_co_ordinate_system_with_-position_reference (as parent)

AIM element: PATH
Rules: 5.2.4.220
Reference path: axis2_placement_3d <=
placement <=
geometric_representation_item <=
representation_item <-
representation_map.mapping_origin
representation_map
{[representation_map.mapped_representation ->
representation
representation.name = ‘local axis representation’]}
representation_map <-

mapped_item.mapping_source
mapped_item
{mapped_item.name = ‘local coordinate system position in parent local coordinate system with position reference’}
mapped_item.mapping_target ->
representation_item =>
geometric_representation_item =>
placement =>
axis2_placement_3d

5.1.11.5 LOCAL_CO_ORDINATE_SYSTEM_WITH_POSITION_-REFERENCE

AIM element: [property_definition]
Source: ISO 10303-41, ISO 10303-42
Rules: 5.2.4.215, 5.2.4.158, 5.2.4.261
Reference path: {[CLASS(property_definition, ‘local coordinate system with position reference’, ‘local coordinate system’)]
[/CLASS(property_definition, ‘local coordinate system’, ‘definition’)/]
[/CLASS(property_definition, ‘definition’, ‘versionable object’)/]
[/ROOT_CLASS(property_definition, ‘versionable object’)/]}
/PROP_DEF_REP_HELP(‘local coordinate system with position reference’)/
representation
{representation.name = ‘local axis with position reference representation‘}
representation.context_of_items
representation_context =>
{representation_context.context_type = ‘local coordinate space’}
geometric_representation_context
{geometric_representation_context.coordinate_space_dimension = 3}
/REP_ITEM(’local axes and origin’)/
geometric_representation_item =>
placement =>
axis2_placement_3d

5.1.11.5.1 description

NOTE Attribute inherited from supertype Item (see 5.1.9.3)

AIM element: /SUPERTYPE(local_coordinate_system)/5.1.11.4.1

5.1.11.5.2 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: /SUPERTYPE(local_coordinate_system)/5.1.11.4.2

5.1.11.5.3 local_co_ordinate_system_with_position_reference to definable_object (as defined_for)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(local_co_ordinate_system)/5.1.11.4.3

5.1.11.5.4 local_co_ordinate_system_with_position_reference to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(local_co_ordinate_system)/5.1.11.4.4

5.1.11.5.5 local_co_ordinate_system_with_position_reference to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(local_co_ordinate_system)/5.1.11.4.5

5.1.11.5.6 local_co_ordinate_system_with_position_reference to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(local_co_ordinate_system)/5.1.11.4.6

5.1.11.5.7 local_co_ordinate_system_with_position_reference to length_-measure (as longitudinal_ref)

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: property_definition<=
/PROP_DEF_REP_HELP('local coordinate system with position reference')/
representation
{representation.name = 'local axis with position reference representation'}
representation.items [i] ->
representation_item
representation_item =>
{representation_item.name = 'longitudinal ref'}
value_representation_item
{value_representation_item.value_component ->
measure_value = length_measure}

5.1.11.5.8 local_co_ordinate_system_with_position_reference to spacing_-position (as longitudinal_ref)

AIM element: compound_representation_item
Source: ISO 10303-43
Reference path: property_definition<=
/PROP_DEF_REP_HELP('local coordinate system with position reference')/
representation
{representation.name = 'local axis with position reference representation'}
representation.items [i] ->
representation_item
(representation_item =>
compound_representation_item
{/CLASS_HELP(compound_representation_item)/
(group.name = 'longitudinal position')
(group.name = 'spacing position with offset')
class})

5.1.11.5.9 local_co_ordinate_system_with_position_reference to global_-axis_placement (as parent)

AIM element: /SUPERTYPE(local_co_ordinate_system)/5.1.11.4.7

5.1.11.5.10 local_co_ordinate_system_with_position_reference to local_-co_ordinate_system (as parent)

AIM element: /SUPERTYPE(local_co_ordinate_system)/5.1.11.4.8

5.1.11.5.11 local_co_ordinate_system_with_position_reference to local_-co_ordinate_system_with_position_reference (as parent)

AIM element: /SUPERTYPE(local_co_ordinate_system)/5.1.11.4.9

5.1.11.5.12 local_co_ordinate_system_with_position_reference to length_-measure (as transversal_ref)

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: property_definition<=
/PROP_DEF_REP_HELP('local coordinate system with position reference')/
representation
{representation.name = 'local axis with position reference representation'}
representation.items [i] ->
representation_item
representation_item =>
{representation_item.name = 'transversal ref'}
value_representation_item

{value_representation_item.value_component ->
measure_value = length_measure}

5.1.11.5.13 local_co_ordinate_system_with_position_reference to spacing_position (as transversal_ref)

AIM element: compound_representation_item
Source: ISO 10303-43
Reference path: property_definition<=
/PROP_DEF_REP_HELP('local coordinate system with position reference')/
representation
{representation.name = 'local axis with position reference representation'}
representation.items [i] ->
representation_item
representation_item =>
compound_representation_item
{CLASS_HELP(compound_representation_item)
(group.name = 'transversal position')
(group.name = 'spacing position with offset')
class}

5.1.11.5.14 local_co_ordinate_system_with_position_reference to length_-measure (as vertical_ref)

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: property_definition<=
/PROP_DEF_REP_HELP('local coordinate system with position reference')/
representation
{representation.name = 'local axis with position reference representation'}
representation.items [i] ->
representation_item
representation_item =>
{representation_item.name = 'vertical ref'}
value_representation_item
{value_representation_item.value_component ->
measure_value = length_measure}

5.1.11.5.15 local_co_ordinate_system_with_position_reference to spacing_position (as vertical_ref)

AIM element: compound_representation_item
Source: ISO 10303-43
Reference path: property_definition<=
/PROP_DEF_REP_HELP('local coordinate system with position reference')/
representation
{representation.name = 'local axis with position reference representation'}
representation.items [i] ->

representation_item
representation_item =>
compound_representation_item
{/CLASS_HELP(compound_representation_item)/
(group.name = ‘vertical position’)
(group.name = ‘spacing position with offset’)
class}

5.1.11.6 LONGITUDINAL_POSITION

AIM element: compound_representation_item
Source: ISO 10303-43
Reference path: {[/CLASS(compound_representation_item, ‘longitudinal position’, ‘spacing position’)/]
[/ROOT_CLASS(compound_representation_item, ‘spacing position’)/]}
{compound_representation_item <-
representation.items[i]
representation
representation.context_of_items
representation_context =>
{representation_context.context_type = ‘global coordinate space’}
geometric_representation_context}

5.1.11.6.1 name

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: /SUPERTYPE(spacing_position)/5.1.11.8.1
Reference path: see supertype mappings

5.1.11.6.2 position

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: /SUPERTYPE(spacing_position)/5.1.11.8.2

5.1.11.6.3 position_number

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: /SUPERTYPE(spacing_position)/5.1.11.8.3

5.1.11.7 LONGITUDINAL_TABLE

AIM element: property_definition
Source: ISO 10303-41
Reference path: {[/CLASS(property_definition, ‘longitudinal table’, ‘spacing table’)/]
[/CLASS(property_definition, ‘spacing table’, ‘definition’)/]
[/CLASS(property_definition, ‘definition’, ‘versionable object’)/]

[/ROOT_CLASS(property_definition, ‘versionable object’)/]}

5.1.11.7.1 description

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.1

5.1.11.7.2 name

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.2

5.1.11.7.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.3

5.1.11.7.4 longitudinal_table to definable_object (as defined_for)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.4

5.1.11.7.5 longitudinal_table to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.5

5.1.11.7.6 longitudinal_table to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.6

5.1.11.7.7 longitudinal_table to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.7

5.1.11.7.8 longitudinal_table to longitudinal_position (as spacing_table_-representations)

AIM element: PATH
Reference path: /PROP_DEF_TO_REP/
representation.items [i] ->
representation_item =>
compound_representation_item
{/CLASS_ID(compound_representation_item, 'longitudinal position')/}

5.1.11.8 SPACING_POSITION

AIM element: compound_representation_item
Source: ISO 10303-43
Reference path: {/ROOT_CLASS(compound_representation_item, 'spacing position')/}
{compound_representation_item <-
representation.items[i]
representation
representation.context_of_items
representation_context =>
{representation_context.context_type = 'global coordinate space'}
geometric_representation_context}

5.1.11.8.1 name

AIM element: compound_representation_item.name
Source: ISO 10303-43

5.1.11.8.2 position

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.350
Reference path: /COMPOUND('position')/
value_representation_item
{value_representation_item.value_component ->
measure_value = length_measure}

5.1.11.8.3 position_number

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.350
Reference path: /COMPOUND('position number')/
value_representation_item
{value_representation_item.value_component ->
measure_value = count_measure}

5.1.11.9 SPACING_POSITION_WITH_OFFSET

AIM element: compound_representation_item
Source: ISO 10303-43
Reference path: {[/CLASS(compound_representation_item, ‘spacing position with offset’, ‘spacing position’)/]
[/ROOT_CLASS(compound_representation_item, ‘spacing position’)/]}
{compound_representation_item <-
representation.items[i]
representation
representation.context_of_items
representation_context =>
{representation_context.context_type = ‘global coordinate space’}
geometric_representation_context}

5.1.11.9.1 name

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: compound_representation_item.name
Source: ISO 10303-43

5.1.11.9.2 offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.352
Reference path: /COMPOUND(‘offset‘)/
value_representation_item
{value_representation_item.value_component ->
measure_value = length_measure}

5.1.11.9.3 position

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: /COMPOUND(‘position‘)/
value_representation_item
{value_representation_item.value_component ->
measure_value = length_measure}

5.1.11.9.4 position_number

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.350

Reference path: /COMPOUND(‘position number‘)/
value_representation_item
{value_representation_item.value_component ->
measure_value = count_measure}

5.1.11.9.5 spacing_position_with_offset to spacing_position (as relating_-spacing_position)

AIM element: PATH
Rules: 5.2.4.351
Reference path: /COMPOUND(‘relating spacing position‘)/
{/CLASS_HELP(compound_representation_item)/
(group.name = ‘longitudinal position’)
(group.name = ‘transversal position’)
(group.name = ‘vertical position’)
class}

5.1.11.10 SPACING_TABLE

AIM element: property_definition
Source: ISO 10303-41
Reference path: {[/CLASS(property_definition, ‘spacing table’, ‘definition’)/]
[/CLASS(property_definition, ‘definition’, ‘versionable object’)/]
[/ROOT_CLASS(property_definition, ‘versionable object’)/]}

5.1.11.10.1 description

AIM element: property_definition.description
Source: ISO 10303-43

5.1.11.10.2 name

AIM element: property_definition.name
Source: ISO 10303-43

5.1.11.10.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.11.10.4 spacing_table to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: property_definition
property_defintion.definition ->
characterized_definition
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
product_definition.formation ->
product_definition_formation
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')}

5.1.11.10.5 spacing_table to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: property_definition
identification_item = property_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.11.10.6 spacing_table to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.11.10.7 spacing_table to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.11.10.8 spacing_table to spacing_position (as spacing_table_-representations)

AIM element: PATH
Reference path: /PROP_DEF_TO_REP/
representation.items [i] ->
representation_item =>
compound_representation_item
{/CLASS_HELP(compound_representation_item)/
(group.name = 'longitudinal position')
(group.name = 'transversal position')
(group.name = 'vertical position')
(group.name = 'spacing position with offset')
class}

5.1.11.11 STATION_TABLE

AIM element: property_definition
Source: ISO 10303-41
Reference path: {[/CLASS(property_definition, 'station table', 'longitudinal table')/]
[/CLASS(property_definition, 'longitudinal table', 'spacing table')/]
[/CLASS(property_definition, 'spacing table', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.11.11.1 description

NOTE Attribute inherited from supertype Spacing_table (see 5.1.11.10).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.1

5.1.11.11.2 name

NOTE Attribute inherited from supertype Spacing_table (see 5.1.11.10).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.2

5.1.11.11.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.3

5.1.11.11.4 station_table to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.4

5.1.11.11.5 station_table to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.5

5.1.11.11.6 station_table to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.6

5.1.11.11.7 station_table to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.7.7

5.1.11.11.8 station_table to longitudinal_position (as spacing_table_-representations)

NOTE Attribute inherited from supertype Longitudinal_table (see 5.1.11.7).

AIM element: /SUPERTYPE(longitudinal_table)/5.1.11.7.8

5.1.11.12 TRANSVERSAL_POSITION

AIM element: compound_representation_item
Source: ISO 10303-43
Reference path: {[/CLASS(compound_representation_item, 'transversal position', 'spacing position')/]
[/ROOT_CLASS(compound_representation_item, 'spacing position')/]}
{compound_representation_item <-
representation.items[i]
representation
representation.context_of_items
representation_context =>
{representation_context.context_type = 'global coordinate space'}
geometric_representation_context}

5.1.11.12.1 name

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: /SUPERTYPE(spacing_position)/5.1.11.8.1

5.1.11.12.2 position

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: /SUPERTYPE(spacing_position)/5.1.11.8.2

5.1.11.12.3 position_number

NOTE Attribute inherited from supertype Spacing_position (see 5.1.11.8).

AIM element: /SUPERTYPE(spacing_position)/5.1.11.8.3

5.1.11.13 TRANSVERSAL_TABLE

AIM element: property_definition
Source: ISO 10303-41
Reference path: {[/CLASS(property_definition, 'transversal table', 'spacing table')/]
[/CLASS(property_definition, 'spacing table', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.11.13.1 description

NOTE Attribute inherited from supertype Spacing_table (see 5.1.11.10).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.1

5.1.11.13.2 name

NOTE Attribute inherited from supertype Spacing_table (see 5.1.11.10).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.2

5.1.11.13.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.3

5.1.11.13.4 transversal_table to definable_object (as defined_for)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.4

5.1.11.13.5 transversal_table to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.5

5.1.11.13.6 transversal_table to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.6

5.1.11.13.7 transversal_table to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.7

5.1.11.13.8 transversal_table to transversal_position (as spacing_table_-representations)

AIM element: PATH
Reference path: property_definition<=
/PROP_DEF_TO_REP/
representation.items [i] ->
representation_item =>
compound_representation_item
{/CLASS_ID(compound_representation_item, 'transversal position')/}

5.1.11.14 VERTICAL_POSITION

AIM element: compound_representation_item
Source: ISO 10303-43
Reference path: {[/CLASS(compound_representation_item, 'vertical position', 'spacing position')/]
[/ROOT_CLASS(compound_representation_item, 'spacing position')/]}
{compound_representation_item <-
representation.items[i]
representation
representation.context_of_items
representation_context =>
{representation_context.context_type = 'global coordinate space'}
geometric_representation_context}

5.1.11.14.1 name

NOTE Attribute inherited from supertype 5.1.11.8

AIM element: /SUPERTYPE(spacing_position)/5.1.11.8.1

5.1.11.14.2 position

NOTE Attribute inherited from supertype 5.1.11.8

AIM element: /SUPERTYPE(spacing_position)/5.1.11.8.2

5.1.11.14.3 position_number

NOTE Attribute inherited from supertype 5.1.11.8

AIM element: /SUPERTYPE(spacing_position)/5.1.11.8.3

5.1.11.15 VERTICAL_TABLE

AIM element: property_definition
Source: ISO 10303-41
Reference path: {[/CLASS(property_definition, 'vertical table', 'spacing table')/]
[/CLASS(property_definition, 'spacing table', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.11.15.1 description

NOTE Attribute inherited from supertype 5.1.11.10

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.1

5.1.11.15.2 name

NOTE Attribute inherited from supertype 5.1.11.10

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.2

5.1.11.15.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.3

5.1.11.15.4 vertical_table to definable_object (as defined_for)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.4

5.1.11.15.5 vertical_table to global_id (as id)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.5

5.1.11.15.6 vertical_table to derived_unit (as local_units)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.6

5.1.11.15.7 vertical_table to named_unit (as local_units)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.7

5.1.11.15.8 vertical_table to vertical_position (as spacing_table_-representations)

AIM element: PATH
Reference path: /PROP_DEF_TO_REP/
representation.items [i] ->
representation_item =>
compound_representation_item
{/CLASS_ID(compound_representation_item, 'vertical position')/}

5.1.11.16 WATERLINE_TABLE

AIM element: property_definition
Source: ISO 10303-41
Reference path: {[/CLASS(property_definition, 'waterline table', 'vertical table')/]
[/CLASS(property_definition, 'vertical table', 'spacing table')/]
[/CLASS(property_definition, 'spacing table', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.11.16.1 description

NOTE Attribute inherited from supertype 5.1.11.10

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.1

5.1.11.16.2 name

NOTE Attribute inherited from supertype 5.1.11.10

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.2

5.1.11.16.3 version_id

NOTE Attribute inherited from supertype Versionable_object (see 5.1.9.7).

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.3

5.1.11.16.4 waterline_table to definable_object (as defined_for)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.4

5.1.11.16.5 waterline_table to global_id (as id)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.5

5.1.11.16.6 waterline_table to derived_unit (as local_units)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.6

5.1.11.16.7 waterline_table to named_unit (as local_units)

NOTE Attribute inherited from supertype 5.1.4.1

AIM element: /SUPERTYPE(spacing_table)/5.1.11.10.7

5.1.11.16.8 waterline_table to vertical_position (as spacing_table_-representations)

NOTE Attribute inherited from supertype 5.1.11.15

AIM element: /SUPERTYPE(vertical_table)/5.1.11.15.8

5.1.12 product_structures UoF

5.1.12.1 ASSEMBLY

AIM element: [product_definition]
[group]
Source: ISO 10303-41
Reference path: {[/LINK_TO_GROUP(product_definition)/]
[/CLASS(product_definition, 'assembly', 'item structure')/]

[/CLASS(product_definition,'item structure', 'definable object')/]
[/CLASS(product_definition,'item structure', 'versionable object')/]
[/ROOT_CLASS(product_definition,'versionable object')/]
[/CLASS(product_definition,'assembly', 'item')/]
[/CLASS(product_definition,'item', 'definable object')/]
[/ROOT_CLASS(product_definition,'definable object')/]
[(/CLASS(product_definition, 'unit assembly', 'assembly')
(/CLASS(product_definition, 'production panel','assembly')/)
(/CLASS(product_definition, 'preconstructed group','assembly')/)
(/CLASS(product_definition, 'preconstructed section','assembly')/)
(/CLASS(product_definition, 'user defined', 'assembly')/]}

5.1.12.1.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.12.1.2 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.12.1.3 the_class

AIM element: group.name
Source: ISO 10303-41
Reference path: product_definition = classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'class membership'}
classification_assignment.assigned_classification ->
group
group.name
{(group.name)
(group.name = 'unit assembly')
(group.name = 'production panel')
(group.name ='preconstructed group')
(group.name ='preconstructed section')}

5.1.12.1.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369

Reference path: /VERSION_ID(product_definition)/

5.1.12.1.5 assembly to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
product_definition = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.12.1.6 assembly to external_instance_reference (as external_items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition,’item structure’)/
{(/EXT_INST_REF(product_definition,'ship structures schema', 'part')/)
(/EXT_INST_REF(product_definition,'ship structures schema', 'assembly')/)}

5.1.12.1.7 assembly to external_instance_reference (as external_-relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship,’item structure’)/
{(/EXT_INST_REF(product_definition_relationship, 'ship structures schema', 'part relationship')/)
(/EXT_INST_REF(product_definition,'ship structures schema', 'assembly relationship')/)}

5.1.12.1.8 assembly to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.12.1.9 assembly to item (as items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition,'item structure')/
{(/CLASS_ID(product_definition, 'part')/)
(/CLASS_ID(product_definition, 'assembly')/)}

5.1.12.1.10 assembly to item_relationship (as relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{(/CLASS_ID(product_definition_relationship, 'part relationship')/)
(/CLASS_ID(product_definition_relationship, 'assembly relationship')/)}

5.1.12.1.11 assembly to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation->
product_definition_formation
{product_definition_formation.id = 'ship structures'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product,'ship')/}

5.1.12.2 ASSEMBLY_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Rules: 5.2.4.16
Reference path: {[/CLASS(product_definition_relationship,'assembly_relationship', 'item relationship')/]
[/CLASS(product_definition_relationship,'item relationship', 'definable object')/]
[/ROOT_CLASS(product_definition_relationship,'definable object')/]
[/CLASS(product_definition_relationship,'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship,'versionable object')/]}

5.1.12.2.1 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.12.2.2 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.12.2.3 assembly_relationship to external_instance_reference (as external_item_1)

AIM element: PATH
Reference path: product_definition_relationship
{product_definition_relationship.relating_product_definition ->
product_definition
[/CLASS_ID(product_definition,'assembly')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'assembly')/]}

5.1.12.2.4 assembly_relationship to external_instance_reference (as external_item_2)

AIM element: PATH
Reference path: product_definition_relationship
{product_definition_relationship.related_product_definition->
product_definition
{(/EXT_INST_REF(product_definition, 'ship structures schema', 'assembly')/)
(/EXT_INST_REF(product_definition, 'ship structures schema', 'part')/)}

5.1.12.2.5 assembly_relationship to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.91
Reference path: product_definition_relationship
identification_item = product_definition_relationship <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.12.2.6 assembly_relationship to assembly (as item_1)

AIM element: PATH
Reference path: product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition,'assembly')/}

5.1.12.2.7 assembly_relationship to item (as item_2)

AIM element: PATH
Reference path: product_definition_relationship
product_definition_relationship.related_product_definition ->
product_definition
{(/CLASS_ID(product_definition,'assembly')/)
(/CLASS_ID(product_definition,'part')/)}

5.1.12.3 COMPOSITE_FEATURE

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.323
Reference path: {[/CLASS(shape_aspect,'composite feature', 'feature')/]
[/CLASS(shape_aspect,'feature', 'item')/]
[/CLASS(shape_aspect,'item', 'definable object')/]
[/ROOT_CLASS(shape_aspect, 'definable object')/]}

5.1.12.3.1 description

AIM element: shape_aspect.description
Source: ISO 10303-41

5.1.12.3.2 name

AIM element: shape_aspect.name
Source: ISO 10303-41

5.1.12.3.3 composite_feature to feature (as composed_of)

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.285, 5.2.4.324
Reference path: shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
{shape_aspect_relationship.name = 'composed of'}
shape_aspect_relationship.related_shape_aspect->
shape_aspect
{(/CLASS_ID(shape_aspect, 'corner cutout')/)

(/CLASS_ID(shape_aspect, ‘edge cutout‘)/)
(/CLASS_ID(shape_aspect, ‘interior cutout‘)/)
(/CLASS_ID(shape_aspect, ‘edge bevel‘)/)
(/CLASS_ID(shape_aspect, ‘seam‘)/)}

5.1.12.3.4 composite_feature to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (shape_aspect
shape_aspect = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (shape_aspect
/DOC_REF(shape_aspect,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.12.3.5 composite_feature to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.346
Reference path: shape_aspect
identification_item = shape_aspect <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.12.3.6 composite_feature to ship (as ship_context)

AIM element: PATH
Reference path: shape_aspect
shape_definition = shape_aspect
characterized_definition = shape_definition
characterized_product_definition = characterized_definition
characterized_product_definition = product_definition
product_definition
product_definition.formation ->

product_definition_formation
{product_definition_formation.id = 'ship structures'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product,'ship')/}

5.1.12.4 EXPLICIT_FEATURE_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Reference path: product_definition_shape
{[/CLASS(product_definition_shape, 'explicit feature design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition','versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]

5.1.12.4.1 description

AIM element: product_definition_shape.description
Source: ISO 10303-41

5.1.12.4.2 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.12.4.3 explicit_feature_design_definition to feature (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition = shape_definition
shape_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect,'corner cutout')/}
{/CLASS_ID(shape_aspect,'edge cutout')/}
{/CLASS_ID(shape_aspect,'interior cutout')/}
{/CLASS_ID(shape_aspect,'edge bevel')/}
{/CLASS_ID(shape_aspect,'seam')/}

5.1.12.4.4 explicit_feature_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.12.4.5 explicit_feature_design_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.12.4.6 explicit_feature_design_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.12.4.7 explicit_feature_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition = shape_aspect
represented_definition <-
/SDR_NAME('explicit feature design shape representation')/
property_definition_representation.used_representation ->
representation
{representation.name = 'representations'} =>
shape_representation =>
advanced_brep_shape_representation

5.1.12.4.8 explicit_feature_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition = shape_aspect
represented_definition <-
/SDR_NAME('explicit feature design shape representation')/
property_definition_representation.used_representation ->
representation
{representation.name = 'representations'} =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.12.4.9 explicit_feature_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition = shape_aspect
represented_definition <-
/SDR_NAME('explicit feature design shape representation')/
property_definition_representation.used_representation ->
representation
{representation.name = 'representations'} =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.12.4.10 explicit_feature_design_definition to non_manifold_surface_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition = shape_aspect
represented_definition <-
/SDR_NAME('explicit feature design shape representation')/
property_definition_representation.used_representation ->
representation
{representation.name = 'representations'} =>
shape_representation =>

non_manifold_surface_shape_representation

5.1.12.5 FEATURE

#1: feature is a composite_feature
#2: feature is a structural_feature

AIM element: #1: /SUBTYPE(composite_feature)/ (See 5.1.12.3)
#2: /SUBTYPE(structural_feature)/ (See 5.1.18.35)

5.1.12.5.1 description

AIM element: #1: /SUBTYPE(composite_feature)/ (See 5.1.12.3.1)
#2: /SUBTYPE(structural_feature)/ (See 5.1.18.35.1)

5.1.12.5.2 name

AIM element: #1: /SUBTYPE(composite_feature)/ (See 5.1.12.3.2)
#2: /SUBTYPE(structural_feature)/ (See 5.1.18.35.2)

5.1.12.5.3 feature to external_reference (as documentation)

AIM element: #1: /SUBTYPE(composite_feature)/ (See 5.1.12.3.4)
#2: /SUBTYPE(structural_feature)/ (See 5.1.18.35.3)

5.1.12.5.4 feature to global_id (as id)

AIM element: #1: /SUBTYPE(composite_feature)/ (See 5.1.12.3.5)
#2: /SUBTYPE(structural_feature)/ (See 5.1.18.35.4)

5.1.12.5.5 feature to ship (as ship_context)

AIM element: #1: /SUBTYPE(composite_feature)/ (See 5.1.12.3.6)
#2: /SUBTYPE(structural_feature)/ (See 5.1.18.35.6)

5.1.12.6 FEATURE_DESIGN_DEFINITION

#1: feature_design_definition is a corner_cutout_design_definition
#2: feature_design_definition is a part_edge_cutout_design_definition
#3: feature_design_definition is a drain_hole_cutout_design_definition
#4: feature_design_definition is a explicit_feature_design_definition
#5: feature_design_definition is a interior_cutout_design_definition
#6: feature_design_definition is a position_feature_design_definition
#7: feature_design_definition is a structural_added_material_feature_design_-definition

#8: feature_design_definition is a structural_weld_shrinkage_allowance_feature_-design_definition

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21)
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13)

5.1.12.6.1 description

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.1)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.1)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.1)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.1)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.1)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.1)
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.2)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.1)

5.1.12.6.2 version_id

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.3)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.4)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.7)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.2)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.2)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.2)
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.3)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.5)

5.1.12.6.3 feature_design_definition to feature (as defined_for)

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.4)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.11)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.8)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.3)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.3)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.3)

#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.4)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.6)

5.1.12.6.4 feature_design_definition to global_id (as id)

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.5)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.12)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.9)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.4)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.4)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.4)
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.9)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.11)

5.1.12.6.5 feature_design_definition to derived_unit (as local_units)

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.6)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.13)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.10)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.5)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.5)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.5)
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.10)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.12)

5.1.12.6.6 feature_design_definition to named_unit (as local_units)

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.7)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.14)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.11)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.6)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.6)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.6)
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.11)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.13)

5.1.12.6.7 feature_design_definition to advanced_brep_shape (as representations)

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.8)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.15)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.12)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.7)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.7)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.7)
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.5)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.7)

5.1.12.6.8 feature_design_definition to edge_based_wireframe_shape (as representations)

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.9)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.16)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.13)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.8)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.8)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.8
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.7)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.9)

5.1.12.6.9 feature_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.10)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.17)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.14)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.9)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.9)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.9)
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.6)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.8)

5.1.12.6.10 feature_design_definition to non_manifold_surface_shape (as representations)

AIM element: #1: /SUBTYPE(corner_cutout_design_definition)/ (See 5.1.18.5.11)
#2: /SUBTYPE(part_edge_cutout_design_definition)/ (See 5.1.18.19.18)
#3: /SUBTYPE(drain_hole_cutout_design_definition)/ (See 5.1.18.6.15)
#4: /SUBTYPE(explicit_feature_design_definition)/ (See 5.1.12.4.10)
#5: /SUBTYPE(interior_cutout_design_definition)/ (See 5.1.18.15.10)
#6: /SUBTYPE(position_feature_design_definition)/ (See 5.1.18.21.10)
#7: /SUBTYPE(structural_added_material_feature_design_definition)/ (See 5.1.15.9.8)
#8: /SUBTYPE(structural_weld_shrinkage_allowance_feature_design_definition)/ (See 5.1.15.13.10)

5.1.12.7 FEATURE_RELATIONSHIP

#1: feature_relationship is a structural_feature_relationship

AIM element: #1: /SUBTYPE(structural_feature_relationship)/ (See 5.1.18.36)

5.1.12.7.1 description

AIM element: #1: /SUBTYPE(structural_feature_relationship)/ (See 5.1.18.36.1)

5.1.12.7.2 version_id

AIM element: #1: /SUBTYPE(structural_feature_relationship)/ (See 5.1.18.36.2)

5.1.12.7.3 feature_relationship to external_instance_reference (as external_item_1)

AIM element: #1: /SUBTYPE(structural_feature_relationship)/ (See 5.1.18.36.3)

5.1.12.7.4 feature_relationship to external_instance_reference (as external_item_2)

AIM element: #1: /SUBTYPE(structural_feature_relationship)/ (See 5.1.18.36.4)

5.1.12.7.5 feature_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(structural_feature_relationship)/ (See 5.1.18.36.5)

5.1.12.7.6 feature_relationship to feature (as item_1)

AIM element: #1: /SUBTYPE(structural_feature_relationship)/ (See 5.1.18.36.6)

5.1.12.7.7 feature_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(structural_feature_relationship)/ (See 5.1.18.36.7)

5.1.12.8 PART

#1: part is a structural_part

AIM element: #1: /SUBTYPE(structural_part)/ (See 5.1.19.35)

5.1.12.8.1 description

AIM element: #1: /SUBTYPE(structural_part)/ (See 5.1.19.35.1)

5.1.12.8.2 name

AIM element: #1: /SUBTYPE(structural_part)/ (See 5.1.19.35.2)

5.1.12.8.3 part to external_reference (as documentation)

AIM element: #1: /SUBTYPE(structural_part)/ (See 5.1.19.35.3)

5.1.12.8.4 part to global_id (as id)

AIM element: #1: /SUBTYPE(structural_part)/ (See 5.1.19.35.4)

5.1.12.8.5 part to ship (as ship_context)

AIM element: #1: /SUBTYPE(structural_part)/ (See 5.1.19.35.5)

5.1.12.9 PART_RELATIONSHIP

#1: part_relationship is a structural_part_relationship

AIM element: #1: /SUBTYPE(structural_part_relationship)/ (See 5.1.19.38)

5.1.12.9.1 description

AIM element: #1: /SUBTYPE(structural_part_relationship)/ (See 5.1.19.38.1)

5.1.12.9.2 version_id

AIM element: #1: /SUBTYPE(structural_part_relationship)/ (See 5.1.19.38.2)

5.1.12.9.3 part_relationship to external_instance_reference (as external_-item_1)

AIM element: #1: /SUBTYPE(structural_part_relationship)/ (See 5.1.19.38.3)

5.1.12.9.4 part_relationship to external_instance_reference (as external_-item_2)

AIM element: #1: /SUBTYPE(structural_part_relationship)/ (See 5.1.19.38.4)

5.1.12.9.5 part_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(structural_part_relationship)/ (See 5.1.19.38.5)

5.1.12.9.6 part_relationship to part (as item_1)

AIM element: #1: /SUBTYPE(structural_part_relationship)/ (See 5.1.19.38.6)

5.1.12.9.7 part_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(structural_part_relationship)/ (See 5.1.19.38.7)

5.1.12.10 SYSTEM

#1: system is a structural_system

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16)

5.1.12.10.1 description

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.3)

5.1.12.10.2 name

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.4)

5.1.12.10.3 version_id

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.12)

5.1.12.10.4 system to external_reference (as documentation)

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.5)

5.1.12.10.5 system to external_instance_reference (as external_items)

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.7)

5.1.12.10.6 system to external_instance_reference (as external_-relationships)

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.8)

5.1.12.10.7 system to global_id (as id)

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.11)

5.1.12.10.8 system to item (as items)

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.9)

5.1.12.10.9 system to item_relationship (as relationships)

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.10)

5.1.12.10.10 system to ship (as ship_context)

AIM element: #1: /SUBTYPE(structural_system)/ (See 5.1.20.16.6)

5.1.12.11 SYSTEM_DESIGN_DEFINITION

#1: system_design_definition is a structural_system_design_definition

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ (See 5.1.20.18)

5.1.12.11.1 description

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ (See 5.1.20.18.1)

5.1.12.11.2 version_id

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ -- (See 5.1.20.18.12)

5.1.12.11.3 system_design_definition to system (as defined_for)

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ (See 5.1.20.18.8)

5.1.12.11.4 system_design_definition to global_id (as id)

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ --- (See 5.1.20.18.11)

5.1.12.11.5 system_design_definition to derived_unit (as local_units)

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ (See 5.1.20.18.9)

5.1.12.11.6 system_design_definition to named_unit (as local_units)

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ (See 5.1.20.18.10)

5.1.12.11.7 system_design_definition to advanced_brep_shape (as representations)

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ (See 5.1.20.18.7)

5.1.12.11.8 system_design_definition to edge_based_wireframe_shape (as representations)

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ (See 5.1.20.18.5)

5.1.12.11.9 system_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ (See 5.1.20.18.6)

5.1.12.11.10 system_design_definition to non_manifold_surface_shape (as representations)

AIM element: #1: /SUBTYPE(structural_system_design_definition)/ (See 5.1.20.18.4)

5.1.12.12 SYSTEM_RELATIONSHIP

#1: system_relationship is a structural_system_relationship

AIM element: #1: /SUBTYPE(structural_system_relationship)/ (See 5.1.20.20)

5.1.12.12.1 description

AIM element: #1: /SUBTYPE(structural_system_relationship)/ (See 5.1.20.20.1)

5.1.12.12.2 version_id

AIM element: #1: /SUBTYPE(structural_system_relationship)/ (See 5.1.20.20.7)

5.1.12.12.3 system_relationship to external_instance_reference (as external_item_1)

AIM element: #1: /SUBTYPE(structural_system_relationship)/ (See 5.1.20.20.3)

5.1.12.12.4 system_relationship to external_instance_reference (as external_item_2)

AIM element: #1: /SUBTYPE(structural_system_relationship)/ (See 5.1.20.20.5)

5.1.12.12.5 system_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(structural_system_relationship)/ (See 5.1.20.20.6)

5.1.12.12.6 system_relationship to system (as item_1)

AIM element: #1: /SUBTYPE(structural_system_relationship)/ (See 5.1.20.20.2)

5.1.12.12.7 system_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(structural_system_relationship)/ (See 5.1.20.20.4)

5.1.13 shapes UoF

5.1.13.1 ADVANCED_BREB_SHAPE

AIM element: advanced_breb_shape_representation
Source: 514

5.1.13.2 EDGE_BASED_WIREFRAME_SHAPE

AIM element: edge_based_wireframe_shape_representation
Source: 501

5.1.13.3 GEOMETRICALLY_BOUNDED_WIREFRAME_SHAPE

AIM element: geometrically_bounded_wireframe_shape_representation
Source: 510

5.1.13.4 NON_MANIFOLD_SURFACE_SHAPE

AIM element: non_manifold_surface_shape_representation
Source: 508

5.1.14 ship_general_characteristics UoF

5.1.14.1 CARRIER

AIM element: product_related_product_category
Source: ISO 10303-41
Reference path: {[/PROD_CAT_NAME(‘carrier‘)/]
[/CLASS(product_related_product_category, ‘Shiptype’, ‘functional definition’)/]
[/CLASS(product_related_product_category,‘functional definition’, ‘definition’)/]
[/CLASS(product_related_product_category,‘definition’, ‘versionable object’)/]
[/ROOT_CLASS(product_related_product_category,‘versionable object’)/]}

5.1.14.1.1 description

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.1

5.1.14.1.2 has_type

AIM element: product_category.name
Source: ISO 10303-41
Reference path: /PROD_CAT_NAME(‘carrier‘)/
product_category <-
product_category_relationship.category
product_category_relationship
{product_category_relationship.name = ‘carrier types‘}
product_category_relationship.sub_category ->
product_category
{(product_category.name =‘Container carrier‘)
(product_category.name = ‘Bulk carrier’)
(product_category.name = ‘Ore carrier’)
(product_category.name = ‘Oil tanker’)
(product_category.name = ‘Roro vessel’)
(product_category.name = ‘Ferry’)
(product_category.name = ‘Car ferry’)
(product_category.name = ‘Cruise liner’)
(product_category.name = ‘Passenger vessel’)
(product_category.name = ‘Cargo ship carrying passengers’)
(product_category.name = ‘Product tanker’)
(product_category.name = ‘Gas carrier’)
(product_category.name = ‘Liquefied gas tanker’)
(product_category.name = ‘Chemical tanker’)
(product_category.name = ‘Chemical tanker Type 1’)
(product_category.name = ‘Tanker for refrigerated fruit juice’)
(product_category.name = ‘General cargo carrier’)
(product_category.name = ‘Dry cargo vessel’)
(product_category.name = ‘Refrigerated cargo carrying ship’)
(product_category.name = ‘High speed craft passenger’)

(product_category.name = ‘High speed craft cargo’)
(product_category.name = ‘Hydrofoil’)
(product_category.name = ‘Car carrier’)
(product_category.name = ‘Barge’)
(product_category.name = ‘Barge for deck loading’)
(product_category.name = ‘Barge for liquefied gas’)
(product_category.name = ‘Barge for oil’)
(product_category.name = ‘Barge pontoon’)
(product_category.name = ‘LNG carrier’)
(product_category.name = ‘LPG carrier’)
(product_category.name = ‘user defined’)}

5.1.14.1.3 user_def_function

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.2

5.1.14.1.4 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.3

5.1.14.1.5 carrier to ship (as defined_for)

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.4

5.1.14.1.6 Shiptype to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16

5.1.14.2 CLASS_AND_STATUTORY_DESIGNATION

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.33
Reference path: {[/CLASS(product_definition, ‘class and statutory designation’, ‘general characteristics definition’)/]
[/CLASS(product_definition, ‘general characteristics definition’, ‘definition’)/]
[/CLASS(product_definition, ‘definition’, ‘versionable object’)/]
[/ROOT_CLASS(product_definition, ‘versionable object’)/]}

5.1.14.2.1 class_number

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.181
Reference path: /PROD_DEF_TO_DESC_REP_ITEM(‘class and statutory designation’, ‘class number‘)/

5.1.14.2.2 description

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: product_definition.description
Source: ISO 10303-41

5.1.14.2.3 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.14.2.4 class_and_statutory_designation to ship (as defined_for)

NOTE Attribute inherited from supertype General_characteristics_definition (see 5.1.4.4).

AIM element: /SUPERTYPE(general_characteristics_definition)/5.1.4.4.3

5.1.14.2.5 class_and_statutory_designation to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.14.2.6 class_and_statutory_designation to class_notation (as the_class)

AIM element: PATH
Rules: 5.2.4.82
Reference path: /PROD_DEF_PROP_DEF/
{/CLASS_ID(property_definition, ‘class notation’)/}

5.1.14.2.7 class_and_statutory_designation to regulation (as the_statutory)

AIM element: PATH
Rules: 5.2.4.86
Reference path: /PROD_DEF_PROP_DEF/
{/CLASS_ID(property_definition, 'regulation')/}

5.1.14.3 CLASS_NOTATION

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.150
Reference path: {/ROOT_CLASS(property_definition, 'class notation')/}

5.1.14.3.1 approval_required_for_heavy_cargo

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.244
Reference path: /PROP_DEF_TO_DESC_REP_ITEM('class notation', 'approval required for heavy cargo')/
{(descriptive_representation_item.description = 'HC')
(descriptive_representation_item.description = 'HC_E')
(descriptive_representation_item.description = 'HC_EA')}

5.1.14.3.2 approval_required_for_oil_cargo

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.266
Reference path: /PROP_DEF_TO_DESC_REP_ITEM('class notation', 'approval required for oil cargo')/
{(descriptive_representation_item.description = 'TRUE')
(descriptive_representation_item.description = 'FALSE')}

5.1.14.3.3 approval_required_loading_unloading_aground

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.266
Reference path: /PROP_DEF_TO_DESC_REP_ITEM('class notation', 'approval required for loading unloading aground')/
{(descriptive_representation_item.description = 'TRUE')
(descriptive_representation_item.description = 'FALSE')}

5.1.14.3.4 approval_required_loading_unloading_grabs

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.266
Reference path: /PROP_DEF_TO_DESC_REP_ITEM('class notation','approval required for unloading grabs')/
{(descriptive_representation_item.description = 'TRUE')
(descriptive_representation_item.description = 'FALSE')}

5.1.14.3.5 class_notations_hull

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.34
Reference path: /PROP_DEF_TO_DESC_REP_ITEM('class notation', 'class notations hull')/

5.1.14.3.6 class_notations_machinery

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.34
Reference path: /PROP_DEF_TO_DESC_REP_ITEM('class notation', 'class notations machinery') /

5.1.14.3.7 class_society

AIM element: organization_assignment.assigned_organization
Source: ISO 10303-41
Rules: 5.2.4.151
Reference path: /ORG_ASSGN(property_definition,'class society')/

5.1.14.3.8 ice_class_notation

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.244
Reference path: /PROP_DEF_TO_DESC_REP_ITEM('class notation', 'ice class notation')/

5.1.14.3.9 service_area

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.266
Reference path: /PROP_DEF_TO_DESC_REP_ITEM('class society', 'service area')/

5.1.14.3.10 service_factor

AIM element: value_representation_item.value_component
Rules: 5.2.4.215, 5.2.4.244
Reference path: /PROP_DEF_TO_VAL_REP_ITEM('class notation', 'service factor', count_measure)/

5.1.14.4 CLASS_PARAMETERS

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.35
Reference path: {[/CLASS(product_definition, 'class parameters', 'general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition', 'definition')/]
[/CLASS(product_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.14.4.1 block_coefficient_class

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.267
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('class parameters', 'block coefficient class', ratio_measure)/

5.1.14.4.2 description

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: product_definition.description
Source: ISO 10303-41

5.1.14.4.3 design_ahead

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.267
Reference path: /PROD_DEF_TO_SPECIAL_VAL_REP_ITEM('class parameters', 'design speed ahead', 'speed unit')/

5.1.14.4.4 design_astern

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.267
Reference path: /PROD_DEF_TO_SPECIAL_VAL_REP_ITEM('class parameters', 'design speed astern', 'speed unit')/

5.1.14.4.5 length_class

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.267
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('class parameters', 'length class',
positive_length_measure)/

5.1.14.4.6 length_solas

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.267
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('class parameters', 'length solas',
positive_length_measure)/

5.1.14.4.7 scantlings_draught

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.267
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('class parameters', 'scantlings draught',
positive_length_measure)/

5.1.14.4.8 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.14.4.9 class_parameters to ship (as defined_for)

NOTE Attribute inherited from supertype General_characteristics_definition (see 5.1.4.4).

AIM element: /SUPERTYPE(general_characteristics_definition)/5.1.4.4.3

5.1.14.4.10 class_parametersto global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]

applied_identification_assignment

5.1.14.4.11 class_parameters to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROD_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.14.4.12 class_parameters to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROD_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.14.5 FREEBOARD_CHARACTERISTICS

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.54
Reference path: {[/CLASS(product_definition, 'freeboard characteristics', 'general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition', 'definition')/]
[/CLASS(product_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.14.5.1 assigned_code

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.270
Reference path: /PROD_DEF_TO_DESC_REP_ITEM('freeboard characteristics', 'assigned code')/

5.1.14.5.2 date_freeboard_assigned

AIM element: date_and_time
Source: ISO 10303-41
Rules: 5.2.4.141
Reference path: /DAT_TIME_ASSGN(product_definition, 'date freeboard assigned')/

5.1.14.5.3 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.14.5.4 freeboard

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.270, 5.2.4.215
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('freeboard characteristics', 'freeboard', positive_length_measure)/

5.1.14.5.5 freeboard_assigned_by

AIM element: organization
Source: ISO 10303-41
Rules: 5.2.4.143
Reference path: /ORG_ASSGN(product_definition, 'freeboard assigned by')/

5.1.14.5.6 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.14.5.7 freeboard_characteristics to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROD_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.14.5.8 freeboard_characteristics to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.14.5.9 freeboard_characteristics to loadline (as applicable_loadline)

AIM element: PATH
Rules: 5.2.4.88
Reference path: /PROD_DEF_PROP_DEF/
{/CLASS_ID(property_definition, 'loadline')/}

5.1.14.5.10 freeboard_characteristics to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROD_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.14.5.11 freeboard_characteristics to ship (as defined_for)

NOTE Attribute inherited from supertype General_characteristics_definition (see 5.1.4.4).

AIM element: /SUPERTYPE(general_characteristics_definition)/5.1.4.4.3

5.1.14.6 LIGHTSHIP_DEFINITION

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.69
Reference path: {[/CLASS(product_definition, 'lightship definition', 'definition')/]
[/CLASS(product_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.14.6.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.14.6.2 lightship_weight

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.271, 5.2.4.215
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('lightship definition','lightship weight',
mass_measure)/

5.1.14.6.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.14.6.4 lightship_definition to centre_location (as lightship_center_-of_gravity)

AIM element: PATH
Rules: 5.2.4.271, 5.2.4.215
Reference path: /PROD_DEF_PROP_DEF_HELP/
/PROP_DEF_TO_REP/
representation.items [i] ->
representation_item =>
{representation_item.name = 'lightship centre of gravity'}
compound_representation_item
{/CLASS_ID(compound_representation_item, 'centre location')/

5.1.14.6.5 lightship_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: /PROD_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.14.6.6 lightship_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.14.6.7 lightship_definition to lightship_weight_item (as lightship_items)

AIM element: PATH
Reference path: /PROD_DEF_PROP_DEF/
{/CLASS_ID(property_definition, 'lightship weight item')/}

5.1.14.6.8 lightship_definition to named_unit (as local_units)

AIM element: PATH
Reference path: /PROD_DEF_PRODUCT/
unit
unit = named_unit
named_unit

5.1.14.6.9 lightship_definition to ship (as defined_for)

AIM element: PATH
Reference path: /PROD_DEF_PROP_DEF/
{/CLASS_ID(product, 'ship')/}

5.1.14.7 LIGHTSHIP_WEIGHT_ITEM

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.171
Reference path: {[/CLASS(property_definition, 'lightship weight item',
'weight_and_centre_of_gravity')/[
[/ROOT_CLASS(property_definition, 'weight_and_centre_of_gravity')/]}

5.1.14.7.1 aft_weight_extent

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.272, 5.2.4.215
Reference path: /PROP_DEF_TO_VAL_REP_ITEM('lightship weight item', 'aft weight extent',
length_measure)/

5.1.14.7.2 lightship_item_description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.14.7.3 fwd_weight_extent

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.272, 5.2.4.215
Reference path: /PROP_DEF_TO_VAL_REP_ITEM('lightship weight item', 'fwd weight extent',
length_measure)/

5.1.14.7.4 mass

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.282, 5.2.4.215
Reference path: /PROP_DEF_TO_VAL_REP_ITEM('weight and centre of gravity', 'mass',
mass_measure)/

5.1.14.7.5 lightship_weight_item to centre_location (as centre_of_gravity)

AIM element: PATH
Rules: 5.2.4.282
Reference path: /PROP_DEF_TO_REP/
representation.items [i] ->
representation_item =>
{representation_item.name = 'centre of gravity'}
compound_representation_item
{/CLASS_ID(compound_representation_item, 'centre location')/

5.1.14.7.6 lightship_weight_item to moment_3d (as moment)

AIM element: PATH
Rules: 5.2.4.216
Reference path: /PROP_DEF_TO_REP/
{representation
representation.name= 'moment 3d'}

5.1.14.8 LOADLINE

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.166
Reference path: {/CLASS_ID(property_definition, 'loadline')/}

5.1.14.8.1 load_line_block_coefficient

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.273, 5.2.4.215
Reference path: /PROP_DEF_TO_VAL_REP_ITEM('loadline', 'load line block coefficient',
ratio_measure)/

5.1.14.8.2 load_line_depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.273, 5.2.4.215
Reference path: /PROP_DEF_TO_VAL_REP_ITEM('loadline', 'load line depth',

positive_length_measure)/

5.1.14.8.3 load_line_displacement

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.273, 5.2.4.215
Reference path: /PROP_DEF_TO_VAL_REP_ITEM('loadline', 'load line displacement',
volume_measure)/

5.1.14.8.4 load_line_length

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.273, 5.2.4.215
Reference path: /PROP_DEF_TO_VAL_REP_ITEM('loadline', 'load line length',
positive_length_measure)/

5.1.14.8.5 load_line_regulation

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.273
Reference path: /PROP_DEF_TO_DESC_REP_ITEM('loadline', 'load line regulation')/
{(descriptive_representation_item.description = 'ILLC_1930')
(descriptive_representation_item.description = 'ILLC_1966')
(descriptive_representation_item.description = 'other')}

5.1.14.9 MOMENT_3D

AIM element: representation
Source: ISO 10303-43
Reference path: {representation
representation.name= 'moment 3d'}

5.1.14.9.1 longitudinal_moment

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.311, 5.2.4.215
Reference path: /REP_TO_SPECIAL_VAL_REP_ITEM('longitudinal moment', moment_unit)/

5.1.14.9.2 transverse_moment

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.311, 5.2.4.215
Reference path: /REP_TO_SPECIAL_VAL_REP_ITEM('transverse moment', moment_unit)/

5.1.14.9.3 vertical_moment

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.311, 5.2.4.215
Reference path: /REP_TO_SPECIAL_VAL_REP_ITEM(‘vertical moment‘, moment_unit)/

5.1.14.9.4 moment_3d to centre_location (as origin)

AIM element: PATH
Rules: 5.2.4.311, 5.2.4.215
Reference path: representation
representation.items [i] ->
representation_item =>
{representation_item.name = ‘origin’}
compound_representation_item
{/CLASS_ID(compound_representation_item, ‘centre location’)/

5.1.14.10 NAVY_SHIP

AIM element: product_related_product_category
Source: ISO 10303-41
Reference path: {[/PROD_CAT_NAME(‘navy ship‘)/]
[/CLASS(product_related_product_category, ‘Shiptype’, ‘functional definition’)/]
[/CLASS(product_related_product_category,‘functional definition’, ‘definition’)/]
[/CLASS(product_related_product_category,‘definition’, ‘versionable object’)/]
[/ROOT_CLASS(product_related_product_category,‘versionable object’)/]}

5.1.14.10.1 description

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.1

5.1.14.10.2 has_type

AIM element: product_category.name
Source: ISO 10303-41
Reference path: /PROD_CAT_NAME(‘navy ship‘)/
product_category <-
product_category_relationship.category
product_category_relationship
{product_category_relationship.name = 'navy ship types‘}
product_category_relationship.category ->
product_category
product_category_relationship
product_category_relationship.sub_category ->
product_category

{(product_category.name = ‘Aircraft carrier’)
(product_category.name = ‘Corvette’)
(product_category.name = ‘Cruiser’)
(product_category.name = ‘Destroyer’)
(product_category.name = ‘Fleet auxiliary vessel’)
(product_category.name = ‘Frigate’)
(product_category.name = ‘Mine warfare ship’)
(product_category.name = ‘Patrol force vessel’)
(product_category.name = ‘Service craft’)
(product_category.name = ‘Submarine’)
(product_category.name = ‘Auxiliary oiler’)
(product_category.name = ‘Landing platform dock’)
(product_category.name = ‘Landing platform helicopter’)
(product_category.name = ‘user defined’)}

5.1.14.10.3 user_def_function

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16)).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.2

5.1.14.10.4 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.3

5.1.14.10.5 navy_ship to ship (as defined_for)

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.4

5.1.14.10.6 navy_ship to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.5

5.1.14.11 OWNER_DESIGNATION

AIM element: product_definition
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition, ‘owner designation’, ‘general characteristics definition’)/]
[/CLASS(product_definition, ‘general characteristics definition’, ‘definition’)/]
[/CLASS(product_definition, ‘definition’, ‘versionable object’)/]
/ROOT_CLASS(product_definition, ‘versionable object’)/]}

5.1.14.11.1 description

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: product_definition.description
Source: ISO 10303-41

5.1.14.11.2 managing_company

AIM element: organization_assignment.assigned_organization
Source: ISO 10303-41
Reference path: /ORG_ASSGN(product_definition,‘managing company‘)/

5.1.14.11.3 ordering_company

AIM element: organization_assignment.assigned_organization
Source: ISO 10303-41
Reference path: /ORG_ASSGN(product_definition,‘ordering company‘)/

5.1.14.11.4 owning_company

AIM element: organization_assignment.assigned_organization
Source: ISO 10303-41
Reference path: /ORG_ASSGN(product_definition,‘owning company‘)/

5.1.14.11.5 owner_approval

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.247
Reference path: /PROD_DEF_TO_DESC_REP_ITEM(‘owner designation’, ‘owner approval‘)/

5.1.14.11.6 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.14.11.7 owner_designation to ship (as defined_for)

NOTE Attribute inherited from supertype General_characteristics_definition (see 5.1.4.4).

AIM element: /SUPERTYPE(general_characteristics_definition)/5.1.4.4.3

5.1.14.11.8 owner_designation to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.14.12 PRINCIPAL_CHARACTERISTICS

AIM element: product_definition
Source: ISO 10303-41
Rules: 5.2.4.80
Reference path: {[/CLASS(product_definition, 'principal characteristics','general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition', 'definition')/]
[/CLASS(product_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.14.12.1 block_coefficient

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.252
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'block coefficient', ratio_measure)/

5.1.14.12.2 description

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: product_definition.description
Source: ISO 10303-41

5.1.14.12.3 design_deadweight

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.252
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'design deadweight', mass_measure)/

5.1.14.12.4 design_draught

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.252
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'design draught', positive_length_measure)/

5.1.14.12.5 length_between_perpendiculars

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.276
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'length between perpendiculars', positive_length_measure)/

5.1.14.12.6 max_draught_at_ap

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.252
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'max draught at ap', positive_length_measure)/

5.1.14.12.7 max_draught_at_fp

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.252
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'max draught at fp', positive_length_measure)/

5.1.14.12.8 min_draught_at_ap

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.252
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'min draught at ap', positive_length_measure)/

5.1.14.12.9 min_draught_at_fp

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.252
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'min draught at fp', positive_length_measure)/

5.1.14.12.10 moulded_breadth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.276
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'moulded breadth', positive_length_measure)/

5.1.14.12.11 moulded_depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.276
Reference path: /PROD_DEF_TO_VAL_REP_ITEM('principal characteristics', 'moulded depth', positive_length_measure)/

5.1.14.12.12 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.14.12.13 principal_characteristics to ship (as defined_for)

NOTE Attribute inherited from supertype General_characteristics_definition (see 5.1.4.4).

AIM element: /SUPERTYPE(general_characteristics_definition)/5.1.4.4.3

5.1.14.12.14 principal_characteristics to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.14.12.15 principal_characteristics to derived_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROD_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.14.12.16 principal_characteristics to named_unit (as local_units)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Reference path: /PROD_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.14.13 REGULATION

AIM element: property_definition
Source: ISO 10303-41
Reference path: {/ROOT_CLASS(property_definition, 'regulation')/}

5.1.14.13.1 regulation to external_reference (as international_regulations)

#1: If as international_regulations is an External_reference

AIM element: PATH
Reference path: property_definition
external_identification_item = property_definition
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.role ->
identificaton_role
{[identification_role.name = 'external reference']
[identification_role.description = 'international regulations']}

#2: If as international_regulations is a Document_reference_with_address

AIM element: PATH
Reference path: property_definition
/DOC_REF(property_definition,'international regulations')/
document

{/CLASS_ID(document, ‘document reference with address’)/}

5.1.14.13.2 regulation to external_reference (as national_regulations)

AIM element: PATH
Reference path: (property_definition
external_identification_item = property_definition
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.role ->
identificaton_role
{[identification_role.name = 'national regulations']
[identification_role.description = 'external reference']})
(property_definition
document_reference_item = property_definition
document_reference_item <-
applied_document_reference.items[i]
applied_document_reference <=
document_reference
{{/ROLE_ASSGN(document_reference)/
object_role.name = 'national regulations'}
document_reference.assigned_document ->
document
{/CLASS_ID(document, ‘document reference with address’)/})

5.1.14.13.3 regulation to external_reference (as standards)

#1: If as standards is an External_reference

AIM element: PATH
Reference path: property_definition
external_identification_item = property_definition
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
identification_assignment.role ->
identificaton_role
{[identification_role.name = 'external reference']
[identification_role.description = 'standards']}

#2: If as standards is a Document_reference_with_address

AIM element: PATH
Reference path: property_definition

/DOC_REF(property_definition,'standards')/
document
{/CLASS_ID(document, 'document reference with address')/}

5.1.14.14 RESEARCH_SHIP

AIM element: product_related_product_category
Source: ISO 10303-41
Reference path: {[/PROD_CAT_NAME('research ship')/]
[/CLASS(product_related_product_category, 'Shiptype', 'functional definition')/]
[/CLASS(product_related_product_category,'functional definition', 'definition')/]
[/CLASS(product_related_product_category,'definition', 'versionable object')/]
[/ROOT_CLASS(product_related_product_category,'versionable object')/]}

5.1.14.14.1 description

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.1

5.1.14.14.2 has_type

AIM element: product_category.name
Source: ISO 10303-41
Reference path: /PROD_CAT_NAME('research ship')/
product_category <-
product_category_relationship.category
product_category_relationship
{product_category_relationship.name = 'research ship types'}
product_category_relationship.sub_category ->
product_category
{product_category.name = 'user defined'}

5.1.14.14.3 user_def_function

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.2

5.1.14.14.4 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.3

5.1.14.14.5 research_ship to ship (as defined_for)

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.4

5.1.14.14.6 research_ship to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.5

5.1.14.15 SHIP_DESIGNATION

AIM element: product_definition
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition, 'ship designation', 'general characteristics definition')/]
[/CLASS(product_definition, 'general characteristics definition', 'definition')/]
[/CLASS(product_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition,'versionable object')/]}

5.1.14.15.1 call_sign

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.81
Reference path: /HAS_ID_1_ROLE(product_definition, 'call sign')/

5.1.14.15.2 description

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: product_definition.description
Source: ISO 10303-41

5.1.14.15.3 flag_state

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.84
Reference path: /HAS_ID_1_ROLE(product_definition, 'flag state')/

5.1.14.15.4 port_of_registration

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.85
Reference path: /HAS_ID_1_ROLE(product_definition, 'port of registration')/

5.1.14.15.5 ship_identification

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.347
Reference path: /HAS_ID_2_ROLES(product_definition, 'IMO number', 'pennant hull number')/

5.1.14.15.6 ship_name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.14.15.7 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.14.15.8 ship_designation to ship (as defined_for)

NOTE Attribute inherited from supertype General_characteristics_definition (see 5.1.4.4).

AIM element: /SUPERTYPE(general_characteristics_definition)/5.1.4.4.3

5.1.14.15.9 ship_designation to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.14.16 Shiptype

#1: Shiptype is a carrier
#2: Shiptype is a navy ship
#3: Shiptype is a research ship
#4: Shiptype is a working ship

AIM element: #1: /SUBTYPE(carrier)/5.1.14.1
#2: /SUBTYPE(navy_ship)/5.1.14.8, 5.1.14.9, 5.1.14.10

#3: /SUBTYPE(research_ship)/5.1.14.14
#4: /SUBTYPE(working_ship)/5.1.14.19

5.1.14.16.1 description

AIM element: product_category.description
Source: ISO 10303-41
Reference path: product_related_product_category <=
product_category
product_category.description

5.1.14.16.2 user_def_function

NOTE Attribute inherited from supertype Functional_definition (see 5.1.4.3).

AIM element: PATH
Reference path: product_related_product_category <=
product_category
{product_category.name = 'user defined'}
product_category<-
product_category_relationship.category
product_category_relationship
product_category_relationship.sub_category ->
product_category
product_category.name

5.1.14.16.3 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_related_product_category)/

5.1.14.16.4 Shiptype to ship (as defined_for)

AIM element: PATH
Reference path: product_related_product_category
product_related_product_category.products[i] ->
product
{/CLASS_ID(product,'ship')/}

5.1.14.16.5 Shiptype to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.145

Reference path: product_related_product_category
identification_item = product_related_product_category <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.14.17 SHIPYARD_DESIGNATION

AIM element: product_definition
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition, ‘shipyard designation’,‘general characteristics definition’)/]
[/CLASS(product_definition, ‘general characteristics definition’, ‘definition’)/]
[/CLASS(product_definition, ‘definition’, ‘versionable object’)/]
[/ROOT_CLASS(product_definition, ‘versionable object’)/]}

5.1.14.17.1 description

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: product_definition.description
Source: ISO 10303-41

5.1.14.17.2 role

AIM element: organization_assignment.role
Source: ISO 10303-41
Reference path: /ORG_ASSGN_PART(product_definition)/
{organization_assignment.role ->
organization_role
{(organization_role.name = ‘prime design’)
(organization_role.name =‘prime build’)
(organization_role.name =‘prime repair’)
(organization_role.name =‘prime’)
(organization_role.name =‘subcontractor’)}

5.1.14.17.3 shipyard

AIM element: organization_assignment.assigned_organization
Source: ISO 10303-41
Rules: 5.2.4.87
Reference path: /ORG_ASSGN(product_definition,‘shipyard‘)/

5.1.14.17.4 shipyard_new_building_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Reference path: /HAS_ID_1_ROLE(product_definition, ‘shipyard new building id’)/

5.1.14.17.5 shipyard_project_name

AIM element: organizational_project.name
Source: ISO 10303-41
Reference path: /ORG_ASSGN(product_definition,‘shipyard‘)/ <-
organizational_project.responsible_organizations[i]
organizational_project
{organizational_project.description = ‘shipyard project name‘}
organizational_project
organizational_project.name

5.1.14.17.6 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.14.17.7 shipyard_designation to ship (as defined_for)

NOTE Attribute inherited from supertype General_characteristics_definition (see 5.1.4.4).

AIM element: /SUPERTYPE(general_characteristics_definition)/5.1.4.4.3

5.1.14.17.8 shipyard_designation to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.14.18 WEIGHT_AND_CENTRE_OF_GRAVITY

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.173
Reference path: {[/ROOT_CLASS(property_definition, ‘weight_and_centre_of_gravity’)/]}

5.1.14.18.1 mass

AIM element: value_representation_item.value_component
Source: ISO 10303-43

Rules: 5.2.4.282, 5.2.4.215
Reference path: /PROP_DEF_TO_VAL_REP_ITEM('weight and centre of gravity', 'mass',
mass_measure)/

5.1.14.18.2 weight_and_centre_of_gravity to centre_location (as centre_of_gravity)

AIM element: PATH
Rules: 5.2.4.282, 5.2.4.215
Reference path: /PROP_DEF_TO_REP/
representation.items [i] ->
representation_item =>
{representation_item.name = 'centre of gravity'}
compound_representation_item
{/CLASS_ID(compound_representation_item, 'centre location')/

5.1.14.18.3 weight_and_centre_of_gravity to moment_3d (as moment)

AIM element: PATH
Rules: 5.2.4.216
Reference path: /PROP_DEF_TO_REP/
{representation
representation.name= 'moment 3d'}

5.1.14.19 WORKING_SHIP

AIM element: product_related_product_category
Source: ISO 10303-41
Reference path: {[/PROD_CAT_NAME('working ship')/]
[/CLASS(product_related_product_category, 'Shiptype', 'functional definition')/]
[/CLASS(product_related_product_category,'functional definition', 'definition')/]
[/CLASS(product_related_product_category,'definition', 'versionable object')/]
/ROOT_CLASS(product_related_product_category,'versionable object')/]}

5.1.14.19.1 description

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.1

5.1.14.19.2 has_type

AIM element: product_category.name
Source: ISO 10303-41
Reference path: /PROD_CAT_NAME('working ship')/
product_category <-
product_category_relationship.category
product_category_relationship

{product_category_relationship.name = ‘working ship types‘}
product_category_relationship.sub_category ->
product_category
{(product_category.name = ‘Tug’)
(product_category.name = ‘Sealer’)
(product_category.name = ‘Fire fighter’)
(product_category.name = ‘Drilling vessel’)
(product_category.name = ‘Pipe laying vessel’)
(product_category.name = ‘Crane vessel’)
(product_category.name = ‘Dredger’)
(product_category.name = ‘Supply vessel’)
(product_category.name = ‘Ice breaker’)
(product_category.name = ‘Fishing vessel’)
(product_category.name = ‘Floating dock’)
(product_category.name = ‘Pilot boat’)
(product_category.name = ‘Floating hotel’)
(product_category.name = ‘Well stimulation vessel’)
(product_category.name = ‘Pusher’)
(product_category.name = ‘Stern trawler’)
(product_category.name = ‘Reefer’)
(product_category.name = ‘Offshore supply vessel’)
(product_category.name = ‘Oil production vessel’)
(product_category.name = ‘Oil storage vessel’)
(product_category.name = ‘Oil production and storage vessel’)
(product_category.name = ‘Shuttle tanker’)
(product_category.name = ‘FPSO’)
(product_category.name = ‘FPGO’)
(product_category.name = ‘user defined’)}

5.1.14.19.3 user_def_function

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.2

5.1.14.19.4 version_id

NOTE Attribute inherited from supertype Versionable_object (See 5.1.9.7).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.3

5.1.14.19.5 working_ship to ship (as defined_for)

NOTE Attribute inherited from supertype Shiptype (see 5.1.14.16).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.4

5.1.14.19.6 working_ship to global_id (as id)

NOTE Attribute inherited from supertype Definition (see 5.1.4.1).

AIM element: /SUPERTYPE(Shiptype)/5.1.14.16.5

5.1.15 ship_manufacturing_definitions UoF

5.1.15.1 ANNOTATION

AIM element: compound_representation_item
Source: ISO 10303-43
Rules: 5.2.4.214, 5.2.4.x (new rule added below:
annotation_compound_representation_has_specified_name)
Reference path: /ROOT_CLASS(compound_representation_item, 'annotation')/

5.1.15.1.1 location

AIM element: cartesian_point
Source: ISO 10303-42
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition = list_representation_item
list_representation_item [i] ->
representation_item =>
{representation_item.name = 'location'}
geometric_representation_item =>
point =>
cartesian_point

5.1.15.1.2 text

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition = list_representation_item
list_representation_item [i] ->
/DSC_REP_ITEM('text')/

5.1.15.2 ASSEMBLY_BOUNDING_BOX

AIM element: representation
Source: ISO 10303-43
Reference path: representation
{representation.name = 'bounding box representation'}

5.1.15.2.1 assembly_bounding_box to centre_location (as point_min)

AIM element: PATH
Rules: 5.2.4.215
Reference path: representation
{representation.name = 'bounding box representation'}
representation.items[i] ->
{representation_item.name = ‘point min’}
compound_representation_item
{/CLASS_ID(compound_representation_item, ‘centre location’)/}

5.1.15.2.2 assembly_bounding_box to centre_location (as point_max)

AIM element: PATH
Rules: 5.2.4.215
Reference path: representation
{representation.name = 'bounding box representation'}
representation.items[i] ->
{representation_item.name = ‘point max’}
compound_representation_item
{/CLASS_ID(compound_representation_item, ‘centre location’)/}

5.1.15.3 ASSEMBLY_MANUFACTURING_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.92
Reference path: {[/CLASS(product_definition_shape,'assembly manufacturing definition',
'manufacturing definition')/]
[/CLASS(product_definition_shape,'manufacturing definition', 'definition')/]
[/CLASS(product_definition_shape,'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape,'versionable object')]}

5.1.15.3.1 assembly_method

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.92
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('assembly manufacturing definition parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'assembly method'}
representation_item =>
descriptive_representation_item

descriptive_representation_item.description

5.1.15.3.2 assembly_stage

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.92
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('assembly manufacturing definition parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'assembly stage'}
representation_item =>
descriptive_representation_item
descriptive_representation_item.description

5.1.15.3.3 centre_of_gravity

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.176, 5.2.4.133
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect,'centre_of_gravity')/]
[shape_aspect = represented_definition <-
/PDR_NAME('centre of gravity representation')/->
representation
representation.items[i] ->
{representation_item.name = ‘centre of gravity’}
compound_representation_item
{/CLASS_ID(compound_representation_item, ‘centre location’)/}

5.1.15.3.4 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.15.3.5 mass

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.176

Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('assembly manufacturing definition parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('mass', mass_measure)/

5.1.15.3.6 production_date

AIM element: date_and_time
Source: ISO 10303-41
Reference path: /DAT_TIME_ASSGN(product_definition_shape,'production date')/

5.1.15.3.7 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.15.3.8 assembly_manufacturing_definition to assembly (as defined_-for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'assembly')/}

5.1.15.3.9 assembly_manufacturing_definition to assembly_-manufacturing_position (as assembly_positions)

AIM element: PATH
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{/CLASS_ID(shape_aspect,'assembly manufacturing position')/}

5.1.15.3.10 assembly_manufacturing_definition to document_reference (as assembly_drawing)

AIM element: PATH
Reference path: document_reference_item <-
applied_document_reference.items[i]
applied_document_reference <=
document_reference

{/ROLE_ASSGN(document_reference)/
object_role.name = 'assembly drawing'}

5.1.15.3.11 assembly_manufacturing_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.15.3.12 assembly_manufacturing_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.15.3.13 assembly_manufacturing_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.15.3.14 assembly_manufacturing_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=

property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.15.3.15 assembly_manufacturing_definition to global_id (as id)

AIM element: PATH
Source: ISO 10303-218
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.15.3.16 assembly_manufacturing_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.15.3.17 assembly_manufacturing_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.15.4 ASSEMBLY_MANUFACTURING_POSITION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.318
Reference path: {/ROOT_CLASS(shape_aspect, 'assembly manufacturing position')/}

5.1.15.4.1 assembly_footprint

AIM element: bounded_curve
Source: ISO 10303-42
Rules: 5.2.4.215, 5.2.4.281
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('assembly manufacturing position parameters')/
representation
{representation.name = 'assembly footprint representation'}
representation.items [i] ->
representation_item
{representation_item.name = 'assembly footprint'}
representation_item =>
geometric_representation_item =>
bounded_curve

5.1.15.4.2 orientation

AIM element: direction
Source: ISO 10303-42
Rules: 5.2.4.215, 5.2.4.281
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('assembly manufacturing position parameters')/ ->
representation
{representation.name = 'orientation representation'}
representation.items [i] ->
representation_item
{representation_item.name = 'orientation'} =>
geometric_representation_item =>
direction

5.1.15.4.3 assembly_manufacturing_position to assembly_bounding_box (as bounding_box)

AIM element: PATH
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('assembly manufacturing position parameters')/ ->
representation
{representation.name = 'bounding box representation'}

5.1.15.5 PLATE_MANUFACTURING_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.116
Reference path: {[/CLASS(product_definition_shape,'plate manufacturing definition', 'structural part manufacturing definition')/]
[/CLASS(product_definition_shape, 'structural part manufacturing definition', 'manufacturing definition')/]
[/CLASS(product_definition_shape,'manufacturing definition', 'definition')/]
[/CLASS(product_definition_shape,'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')]}

5.1.15.5.1 annotation

AIM element: /SUPERTYPE (structural_part_manufacturing_definition)/ -- (see 5.1.15.11.1)

5.1.15.5.2 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.15.5.3 inner_contours

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.4)

5.1.15.5.4 layout_marks

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.5, 5.1.15.11.6, 5.1.15.11.6)

5.1.15.5.5 outer_contour

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.79
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'outer contour')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('outer contour representation')/->
representation
representation.items[i] ->

/GEO_REP_ITEM('.UNUSED.', edge_based_wireframe_model)/]}

5.1.15.5.6 version_id

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.8, 5.1.15.11.9, 5.1.15.11.9)

5.1.15.5.7 plate_manufacturing_definition to structural_part (as defined_-for)

#1: structural_part is a Plate
#2: structural_part is a Corrugated_part

AIM element: #1: PATH
#2: PATH
Reference path: #1: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'plate')/}

#2: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition,'corrugated part')/}

5.1.15.5.8 plate_manufacturing_definition to advanced_brep_shape (as representations)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.11)

5.1.15.5.9 plate_manufacturing_definition to edge_based_wireframe_-shape (as representations)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.13)

5.1.15.5.10 plate_manufacturing_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.12)

5.1.15.5.11 plate_manufacturing_definition to non_manifold_surface_-shape (as representations)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.14)

5.1.15.5.12 plate_manufacturing_definition to global_id (as id)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.15)

5.1.15.5.13 plate_manufacturing_definition to derived_unit (as local_-units)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.16)

5.1.15.5.14 plate_manufacturing_definition to named_unit (as local_units)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.17)

5.1.15.6 PROFILE_MANUFACTURING_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition_shape,'profile manufacturing definition',
'structural part manufacturing definition')/]
[/CLASS(product_definition_shape,'structural part manufacturing definition',
'manufacturing definition')/]
[/CLASS(product_definition_shape,'manufacturing definition', 'definition')/]
[/CLASS(product_definition_shape,'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape,'versionable object')]}

5.1.15.6.1 annotation

AIM element: /SUPERTYPE (structural_part_manufacturing_definition)/ – (see 5.1.15.11.1)

5.1.15.6.2 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.15.6.3 inner_contours

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.4)

5.1.15.6.4 layout_marks

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.5, 5.1.15.11.6, 5.1.15.11.6)

5.1.15.6.5 inverse_bend_trace

AIM element: PATH
Rules: 5.2.4.215
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'inverse bend trace')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('inverse bend trace representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', bounded_curve)/]}

5.1.15.6.6 outer_flange_countour

AIM element: PATH
Rules: 5.2.4.215
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'outer flange contour')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('outer flange contour representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', bounded_curve)/]}

5.1.15.6.7 outer_web_countour

AIM element: PATH
Rules: 5.2.4.215
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'outer web contour')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('outer web contour representation')/->

representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', bounded_curve)/]}

5.1.15.6.8 version_id

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.8, 5.1.15.11.9, 5.1.15.11.9)

5.1.15.6.9 profile_manufacturing_definition to profile (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition,'profile')/}

5.1.15.6.10 profile_manufacturing_definition to global_id (as id)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.15)

5.1.15.6.11 profile_manufacturing_definition to derived_unit (as local_-units)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.16)

5.1.15.6.12 profile_manufacturing_definition to named_unit (as local_-units)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.17)

5.1.15.6.13 profile_manufacturing_definition to advanced_brep_shape (as representations)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.11)

5.1.15.6.14 profile_manufacturing_definition to edge_based_wireframe_-shape (as representations)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.13)

5.1.15.6.15 profile_manufacturing_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.12)

5.1.15.6.16 profile_manufacturing_definition to non_manifold_surface_-shape (as representations)

AIM element: /SUPERTYPE(Structural_part_manufacturing_definition)/ (See 5.1.15.11.14)

5.1.15.7 STRUCTURAL_ADDED_MATERIAL_BOUNDARY_-RELATIONSHIP

AIM element: shape_aspect_relationship
Source: ISO 10303-41
Reference path: {[/CLASS(shape_aspect_relationship,'structural added material boundary relationship', 'structural feature relationship')/]
[/CLASS(shape_aspect_relationship,'structural feature relationship', 'feature relationship')/]
[/CLASS(shape_aspect_relationship,'feature relationship', 'item relationship')/]
[/CLASS(shape_aspect_relationship,'item relationship', 'definable object')/]
[/CLASS(shape_aspect_relationship,'item relationship', 'versionable object')/]
[/ROOT_CLASS(shape_aspect_relationship,'definable object')]
[/ROOT_CLASS(shape_aspect_relationship,'versionable object')]}

5.1.15.7.1 boundary_index

AIM element: PATH
Reference path: shape_aspect_relationship
shape_aspect_relationship.related_shape_aspect->
shape_aspect
[{/CLASS_ID(shape_aspect,'border')/}]
[{shape_aspect.of_shape->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{(/CLASS_ID(product_definition,'profile')/)
(/CLASS_ID(product_definition,'plate')/)
(/CLASS_ID(product_definition,'corrugated part')/)}]

5.1.15.7.2 description

AIM element: shape_aspect_relationship.description
Source: ISO 10303-41

5.1.15.7.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(shape_aspect_relationship)/

5.1.15.7.4 structural_added_material_boundary_relationship to structural_added_material_feature (as item_1)

AIM element: PATH
Reference path: shape_aspect_relationship
shape_aspect_relationship.relating_shape_aspect ->
shape_aspect
(/CLASS_ID(shape_aspect,'structural added material feature')/)

5.1.15.7.5 structural_added_material_boundary_relationship to external_-instance_reference (as external_item_1)

AIM element: PATH
Reference path: shape_aspect_relationship
/EXT_INST_REF(shape_aspect_relationship, 'ship structures schema', 'shape aspect')/

5.1.15.7.6 structural_added_material_boundary_relationship to external_-instance_reference (as external_item_2)

AIM element: PATH
Reference path: shape_aspect_relationship
/EXT_INST_REF(shape_aspect_relationship,'ship structures schema', 'product definition')/
{(/CLASS_ID(product_definition,'profile')/)
(/CLASS_ID(product_definition,'plate')/)
(/CLASS_ID(product_definition, 'corrugated part')/)}

5.1.15.7.7 structural_added_material_boundary_relationship to item (as item_2)

#1: item is a Profile
#2: item is a Plate
#3: item is a Corrugated_part

AIM element: PATH
Reference path: #1: shape_aspect_relationship <-
shape_aspect_relationship.related_shape_aspect ->
shape_aspect
[{/CLASS_ID(shape_aspect,'border')/}]
[shape_aspect.of_shape ->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition,'profile')/}]

#2: shape_aspect_relationship <-
shape_aspect_relationship.related_shape_aspect ->
shape_aspect

[{/CLASS_ID(shape_aspect,'border')/}]
[shape_aspect.of_shape ->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition,'plate')/}]

#3: shape_aspect_relationship <-
shape_aspect_relationship.related_shape_aspect ->
shape_aspect
[{/CLASS_ID(shape_aspect,'border')/}]
[shape_aspect.of_shape ->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'corrugated part')/}]

5.1.15.7.8 structural_added_material_boundary_relationship to global_id (as id)

AIM element: ISO 10303-218
Rules: 5.2.4.56, 5.2.4.137
Reference path: shape_aspect_relationship
identification_item = shape_aspect_relationship
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.15.8 STRUCTURAL_ADDED_MATERIAL_FEATURE

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: {[/CLASS(shape_aspect, 'structural added material feature', 'structural manufacturing feature')/]
[/CLASS(shape_aspect,'structural manufacturing feature', 'structural feature')/]
[/CLASS(shape_aspect,'structural feature', 'feature')/]
[/CLASS(shape_aspect,'feature', 'item')/]
[/ROOT_CLASS(shape_aspect, 'item', 'definable object')/]

5.1.15.8.1 description

AIM element: shape_aspect.description
Source: ISO 10303-41

5.1.15.8.2 name

AIM element: shape_aspect.name
Source: ISO 10303-41

5.1.15.8.3 structural_added_material_feature to structural_part (as parent)

#1: Plate
#2: Profile
#3: Corrugated_part

AIM element: PATH
Reference path: #1: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'parent'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition,'plate')/}

#2: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'parent'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition,'profile')/}

#3: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'parent'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'corrugated part')/}

5.1.15.8.4 structural_added_material_feature to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (shape_aspect
shape_aspect = external_identification_item

external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (shape_aspect
/DOC_REF(shape_aspect,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.15.8.5 structural_added_material_feature to ship (as ship_context)

AIM element: PATH
Reference path: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'ship context'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')/}

5.1.15.8.6 structural_added_material_feature to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: shape_aspect
identification_item = shape_aspect <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.15.9 STRUCTURAL_ADDED_MATERIAL_FEATURE_DESIGN_-DEFINITION

AIM element: property_definition
Source: ISO 10303-43
Rules: 5.2.4.168
Reference path: {[/CLASS(property_definition, 'structural added material feature design definition', 'feature design definition')/]
[/CLASS(property_definition,'feature design definition', 'design definition')/]
[/CLASS(property_definition,'design definition', 'definition')/]
[/CLASS(property_definition,'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition,'versionable object')]}

5.1.15.9.1 added_material_length

AIM element: PATH
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.208
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('structural added material feature design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('added material length', positive_length_measure)/

5.1.15.9.2 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.15.9.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.15.9.4 structural_added_material_feature_design_definition to structural_added_material_feature (as defined_for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'structural added material feature')/}

5.1.15.9.5 structural_added_material_feature_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.15.9.6 structural_added_material_feature_design_definition to non_-manifold_surface_shape (as representations)

AIM element: PATH
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.15.9.7 structural_added_material_feature_design_definition to edge_-based_wireframe_shape (as representations)

AIM element: PATH
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.15.9.8 structural_added_material_feature_design_definition to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/

property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.15.9.9 structural_added_material_feature_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: property_definition
identification_item = property_definition
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.15.9.10 structural_added_material_feature_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.15.9.11 structural_added_material_feature_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.15.10 STRUCTURAL_MANUFACTURING_FEATURE

#1: structural_manufacturing_feature is a structural_added_material_feature
#2: structural_manufacturing_feature is a structural_weld_shrinkage_allowance_-
feature

AIM element: #1: /SUBTYPE(structural_added_material_feature)/ (See 5.1.15.8)
#2: /SUBTYPE(structural_weld_shrinkage_allowance_feature)/ (See 5.1.15.12)

5.1.15.10.1 description

AIM element: #1: /SUBTYPE(structural_added_material_feature)/ (See 5.1.15.8.1)
#2: /SUBTYPE(structural_weld_shrinkage_allowance_feature)/ (See 5.1.15.12.1)

5.1.15.10.2 name

AIM element: #1: /SUBTYPE(structural_added_material_feature)/ (See 5.1.15.8.2)
#2: /SUBTYPE(structural_weld_shrinkage_allowance_feature)/ (See 5.1.15.12.2)

5.1.15.10.3 structural_manufacturing_feature to structural_part_item (as parent)

AIM element: #1: /SUBTYPE(structural_added_material_feature)/ (See 5.1.15.8.3)
#2: /SUBTYPE(structural_weld_shrinkage_allowance_feature)/ (See 5.1.15.12.3)

5.1.15.10.4 structural_manufacturing_feature to external_reference (as documentation)

AIM element: #1: /SUBTYPE(structural_added_material_feature)/ (See 5.1.15.8.4)
#2: /SUBTYPE(structural_weld_shrinkage_allowance_feature)/ (See 5.1.15.12.4)

5.1.15.10.5 structural_manufacturing_feature to ship (as ship_context)

AIM element: #1: /SUBTYPE(structural_added_material_feature)/ (See 5.1.15.8.5)
#2: /SUBTYPE(structural_weld_shrinkage_allowance_feature)/ (See 5.1.15.12.5)

5.1.15.10.6 structural_manufacturing_feature to global_id (as id)

AIM element: #1: /SUBTYPE(structural_added_material_feature)/ (See 5.1.15.8.6)
#2: /SUBTYPE(structural_weld_shrinkage_allowance_feature)/ (See 5.1.15.12.6)

5.1.15.11 STRUCTURAL_PART_MANUFACTURING_DEFINITION

#1: structural_part_manufacturing_definition is a plate_manufacturing_definition
#2: structural_part_manufacturing_definition is a profile_manufacturing_definition

AIM element: #1: /SUBTYPE(plate_manufacturing_definition)/ (See 5.1.15.5)
#2: /SUBTYPE(profile_manufacturing_definition)/ (See 5.1.15.6)

5.1.15.11.1 structural_part_manufacturing_definition to annotation (as bottom_annotation)

AIM element: PATH
Rules: 5.2.4.214
Reference path: product_definition_shape <-
shape_aspect.of_shape

shape_aspect
{[/ROOT_CLASS(shape_aspect, 'bottom annotation')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('annotation representation')/ ->
representation
representation.items[i] ->
{representation_item.name = 'bottom annotation'}
compound_representation_item
{/CLASS_ID(compound_representation_item,'annotation')/}]}

5.1.15.11.2 structural_part_manufacturing_definition to annotation (as top_annotation)

AIM element: PATH
Rules: 5.2.4.214
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'top annotation')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('annotation representation')/ ->
representation
representation.items[i] ->
{representation_item.name = 'top annotation'}
compound_representation_item
{/CLASS_ID(compound_representation_item,'annotation')/}]}

5.1.15.11.3 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.15.11.4 inner_contours

AIM element: PATH
Rules: 5.2.4.215
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'inner contour')/]
[shape_aspect
represented_definition = shape_aspect

represented_definition <-
/PDR_NAME('inner contour representation')/->
representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', bounded_curve)/]}

5.1.15.11.5 bottom_layout_marks

AIM element: PATH
Rules: 5.2.4.214
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'bottom layout mark')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('layout mark representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('bottom layout mark', bounded_curve)/]}

5.1.15.11.6 top_layout_marks

AIM element: PATH
Rules: 5.2.4.214
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'top layout mark')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('layout mark representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('top layout mark', bounded_curve)/]}

5.1.15.11.7 bottom_punch_points

AIM element: PATH
Rules: 5.2.4.214
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'bottom punch point')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('punch point representation')/ ->

representation
representation.items[i] ->
/GEO_REP_ITEM('bottom punch point', cartesian_point)/]}

5.1.15.11.8 top_punch_points

AIM element: PATH
Rules: 5.2.4.214
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'top punch point')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('punch point representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('top punch point', cartesian_point)/]}

5.1.15.11.9 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.15.11.10 structural_part_manufacturing_definition to structural_part (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'structural part')/}

5.1.15.11.11 structural_part_manufacturing_definition to advanced_-brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.15.11.12 structural_part_manufacturing_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.15.11.13 structural_part_manufacturing_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.15.11.14 structural_part_manufacturing_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.15.11.15 structural_part_manufacturing_definition to global_id (as id)

AIM element: PATH
Source: ISO 10303-218
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape

identification_item = product_definition_shape
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.15.11.16 structural_part_manufacturing_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.15.11.17 structural_part_manufacturing_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.15.12 STRUCTURAL_WELD_SHRINKAGE_ALLOWANCE_-FEATURE

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: shape_aspect
{[/CLASS(shape_aspect, 'structural weld shrinkage allowance feature', 'structural manufacturing feature')/]
[/CLASS(shape_aspect, 'structural manufacturing feature', 'structural feature')/]
[/CLASS(shape_aspect, 'structural feature', 'feature')/]
[/CLASS(shape_aspect, 'feature', 'item')/]
[/ROOT_CLASS(shape_aspect, 'item', 'definable object')/]}

5.1.15.12.1 description

AIM element: shape_aspect.description
Source: ISO 10303-41

5.1.15.12.2 name

AIM element: shape_aspect.name
Source: ISO 10303-41

5.1.15.12.3 structural_weld_shrinkage_allowance_feature to structural_part (as parent)

#1: Structural_part is a Plate
#2: Structural_part is a Profile

AIM element: #1: PATH
#2: PATH
Reference path:
#1: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'parent'}
property_definition.definition->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition,'plate')/}

#2: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'parent'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition,'profile')/}

5.1.15.12.4 structural_weld_shrinkage_allowance_feature to external_-reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (shape_aspect
shape_aspect = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment

{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (shape_aspect
/DOC_REF(shape_aspect,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.15.12.5 structural_weld_shrinkage_allowance_feature to ship (as ship_context)

AIM element: PATH
Reference path: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'ship context'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')/}

5.1.15.12.6 structural_weld_shrinkage_allowance_feature to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: shape_aspect
identification_item = shape_aspect <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.15.13 STRUCTURAL_WELD_SHRINKAGE_ALLOWANCE_-FEATURE_DESIGN_DEFINITION

AIM element: property_definition
Source: ISO 10303-43
Rules: 5.2.4.167
Reference path: {[/CLASS(property_definition, 'structural weld shrinkage allowance feature design definition', 'feature design definition')/]
[/CLASS(property_definition, 'feature design definition', 'design definition')/]
[/CLASS(property_definition, 'design definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')]}

5.1.15.13.1 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.15.13.2 longitudinal_allowance

AIM element: PATH
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.219
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('structural weld shrinkage allowance feature design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('longitudinal allowance', ratio_measure)/

5.1.15.13.3 transverse_allowance

AIM element: PATH
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.219
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('structural weld shrinkage allowance feature design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('transverse allowance', ratio_measure)/

5.1.15.13.4 vertical_allowance

AIM element: PATH
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.219
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('structural weld shrinkage allowance feature design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('verticalallowance', ratio_measure)/

5.1.15.13.5 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.15.13.6 structural_weld_shrinkage_allowance_feature_design_-definition to structural_weld_shrinkage_allowance_feature (as defined_-for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'structural weldshrinkage allowance feature')/}

5.1.15.13.7 structural_weld_shrinkage_allowance_feature_design_-definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.15.13.8 structural_weld_shrinkage_allowance_feature_design_-definition to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.15.13.9 structural_weld_shrinkage_allowance_feature_design_-definition to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.15.13.10 structural_weld_shrinkage_allowance_feature_design_-definition to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.15.13.11 structural_weld_shrinkage_allowance_feature_design_-definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: property_definition
identification_item = property_definition
identification_item<-

applied_identification_assignment.items[i]
applied_identification_assignment

5.1.15.13.12 structural_weld_shrinkage_allowance_feature_design_-definition to derived_unit (as local_units)

AIM element: PATH
Reference path: property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.15.13.13 structural_weld_shrinkage_allowance_feature_design_-definition to named_unit (as local_units)

AIM element: PATH
Reference path: property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.15.14 WELD_MANUFACTURING_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition_shape,'weld manufacturing definition','manufacturing definition')/]
[/CLASS(product_definition_shape,'manufacturing definition', 'definition')/]
[/CLASS(product_definition_shape,'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape,'versionable object')]}

5.1.15.14.1 degree_of_automations

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.307
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('weld manufacturing definition parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'degree of automations'}
representation_item =>

descriptive_representation_item
{(descriptive_representation_item.description= 'manual')
(descriptive_representation_item.description = 'self run')
(descriptive_representation_item.description = 'automatic')
(descriptive_representation_item.description = 'robotic')
(descriptive_representation_item.description= 'mechanized')
(descriptive_representation_item.description= 'user defined')}

5.1.15.14.2 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.15.14.3 environment

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.307
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('weld manufacturing definition parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'environment'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description= 'field')
(descriptive_representation_item.description = 'shop')}

5.1.15.14.4 number_of_weld_passes

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.307
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('weld manufacturing definition parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('number of weld passes', count_measure)/

5.1.15.14.5 position

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.307
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('weld manufacturing definition parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'position'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description= 'flat')
(descriptive_representation_item.description = 'horizontal')
(descriptive_representation_item.description = 'vertical')
(descriptive_representation_item.description = 'overhead')
(descriptive_representation_item.description = 'other')}

5.1.15.14.6 process

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.307
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('weld manufacturing definition parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'process'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description= 'SMAW')
(descriptive_representation_item.description = 'SAW')
(descriptive_representation_item.description = 'GMAW')
(descriptive_representation_item.description = 'FCAW')
(descriptive_representation_item.description = 'user defined')}

5.1.15.14.7 tack_weld_used

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.307
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('weld manufacturing definition parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'tack weld used'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description= 'TRUE')
(descriptive_representation_item.description = 'FALSE')}

5.1.15.14.8 torch_vector

AIM element: vector
Source: ISO 10303-42
Rules: 5.2.4.215, 5.2.4.97
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'torch vector')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('torch vector representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', vector)/]}

5.1.15.14.9 welding_deposition_sequences

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.307
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('weld manufacturing definition parameters')/ ->
representation
representation.items[i]->
representation_item

{representation_item.name = 'welding deposition sequences'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description= 'progressive')
(descriptive_representation_item.description = 'backstep')
(descriptive_representation_item.description = 'wandering')
(descriptive_representation_item.description = 'user defined')}

5.1.15.14.10 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.15.14.11 weld_manufacturing_definition to weld_testing (as weld_test)

AIM element: PATH
Reference path: action_assignment_item = product_definition_shape <-
applied_action_assignment.items[i]
applied_action_assignment <=
action_assignment
{/ROLE_ASSGN(action_assignment)/
object_role.name = 'weld test'}
action_assignment.assigned_action ->
executed_action <=
action

5.1.15.14.12 weld_manufacturing_definition to welding_sequence (as sequences)

AIM element: PATH
Source: ISO 10303-45
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('welding sequence representation')/ ->
representation
(/CLASS_ID(representation, ‘welding sequence’)/}

5.1.15.14.13 weld_manufacturing_definition to document (as welding_-procedures)

AIM element: PATH
Reference path: document_reference_item = product_definition_shape<-
applied_document_reference.items[i]
applied_document_reference
{applied_document_reference <=
document_reference

{/ROLE_ASSGN(document_reference)/
object_role.name = 'welding procedures'}

5.1.15.14.14 weld_manufacturing_definition to weld (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'weld')/}

5.1.15.14.15 weld_manufacturing_definition to advanced_brep_shape (as representation)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.15.14.16 weld_manufacturing_definition to geometrically_bounded_-wireframe_shape (as representation)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.15.14.17 weld_manufacturing_definition to edge_based_wireframe_-shape (as representation)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('representation')/
property_definition_representation.used_representation ->

representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.15.14.18 weld_manufacturing_definition to non_manifold_surface_-shape (as representation)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.15.14.19 weld_manufacturing_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.15.14.20 weld_manufacturing_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.15.14.21 weld_manufacturing_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.15.15 WELD_TESTING

AIM element: executed_action
Source: ISO 10303-41

5.1.15.15.1 test_methods

AIM element: PATH
Reference path: executed_action <=
action
action.chosen_method ->
action_method
{(action_method.name = 'visual examination')
(action_method.name = 'dye penetrant')
(action_method.name = 'magnetic particle')
(action_method.name = 'radiographic')
(action_method.name = 'ultrasonic')}
action_method <-
action_method_relationship.related_method
action_method_relationship
action_method_relationship.relating_method->
action_method
{(action_method.name = 'visual examination')
(action_method.name = 'dye penetrant')
(action_method.name = 'magnetic particle')
(action_method.name = 'radiographic')
(action_method.name = 'ultrasonic')}

5.1.15.15.2 test_results

AIM element: PATH
Reference path: executed_action <-
action_status.assigned_action
action_status
{(action_status.status = 'pre qualified')
(action_status.status = 'rejected')
(action_status.status = 'pending')
(action_status.status = 'other')}

5.1.15.15.3 weld_testing to document (as test_sequence)

AIM element: PATH
Reference path: executed_action =
document_reference_item <-
applied_document_reference.items[i]
applied_document_reference
{applied_document_reference <=
document_reference
{/ROLE_ASSGN(document_reference)/

object_role.name = 'test sequence'}

5.1.15.16 WELDING_SEQUENCE

AIM element: representation
Source: ISO 10303-41
Reference path: {/ROOT_CLASS(representation,'welding sequence')/}

5.1.15.16.1 direction

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.215, 5.2.4.309
Reference path: representation
representation.items[i] ->
representation_item
{representation_item.name = 'tack weld used'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description= 'TRUE')
(descriptive_representation_item.description = 'FALSE')}

5.1.15.16.2 end_offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.309
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('end offset', length_measure)/

5.1.15.16.3 name

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.309
Reference path: representation
representation.items[i] ->
representation_item
{representation_item.name = 'name'}
representation_item =>
descriptive_representation_item

5.1.15.16.4 start_offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.309
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('start offset', length_measure)/

5.1.16 ship_materials UoF

5.1.16.1 HOMOGENEOUS_SHIP_MATERIAL_PROPERTY

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.155
Reference path: {[/CLASS(property_definition, 'homogeneous ship material property',
'ship material property')/]
[/CLASS(property_definition, 'ship material property', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.16.1.1 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.16.1.2 density

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.291
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('homogeneous ship material property parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD('density', ‘density unit’)/

5.1.16.1.3 poisson_ratio

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.291
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('homogeneous ship material property parameters')/ ->
representation

representation.items[i] ->
/VAL_REP_ITEM('poisson ratio', ratio_measure)/

5.1.16.1.4 stress_of_fracture

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.291
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('homogeneous ship material property parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD('stress of fracture', ‘stress unit’)/

5.1.16.1.5 thermal_expansion_coefficient

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.291
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('homogeneous ship material property parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD('thermal expansion coefficient', ‘dilatation unit’)/

5.1.16.1.6 yield_point

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.291
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('homogeneous ship material property parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM_CD('yield point', ‘stress unit’)/

5.1.16.1.7 youngs_module

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.291
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('homogeneous ship material property parameters')/ ->
representation
representation.items[i] ->

/VAL_REP_ITEM_CD('youngs module', ‘stress unit’)/

5.1.16.1.8 homogeneous_ship_material_property to part (as defined_for)

AIM element: PATH

Reference path: property_definition
property_definition.definition ->
characterized_definition
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS(product_definition, 'part')/}

5.1.16.1.9 homogeneous_ship_material_property to external_reference (as material_reference)

#1: If as documentation refers to an External_reference

AIM element: PATH
Rules: 5.2.4.58
Reference path: product_definition
product_definition = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = ‘material reference']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Rules: 5.2.4.58
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.16.1.10 homogeneous_ship_material_property to global_id (as id)

AIM element: PATH
Rules: 5.2.4.169, 5.2.4.56
Reference path: property_definition
identification_item = property_definition <-
applied_identification_assignment.items[i]

applied_identification_assignment

5.1.16.1.11 homogeneous_ship_material_property to derived_unit (as local_units)

AIM element: PATH
Reference path: property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.16.1.12 homogeneous_ship_material_property to named_unit (as local_units)

AIM element: PATH
Reference path: property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.16.1.13 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.16.2 SHIP_MATERIAL_PROPERTY

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.163
Reference path: {[/CLASS(property_definition, 'ship material property', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.16.2.1 density

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.300
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('ship material property parameters')/ ->
representation
representation.items[i] ->

/VAL_REP_ITEM_CD('density', ‘density unit’)/

5.1.16.2.2 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.16.2.3 ship_material_property to external_reference (as material_-reference)

#1: If as documentation refers to an External_reference

AIM element: PATH
Rules: 5.2.4.348
Reference path: (product_definition
product_definition = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = ‘material reference']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Rules: 5.2.4.348
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.16.2.4 ship_material_property to global_id (as id)

AIM element: PATH
Rules: 5.2.4.169, 5.2.4.56
Reference path: property_definition
identification_item = property_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.16.2.5 ship_material_property to derived_unit (as local_units)

AIM element: PATH
Reference path: property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.16.2.6 ship_material_property to named_unit (as local_units)

AIM element: PATH
Reference path: property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.16.2.7 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.17 ship_measures UoF

5.1.17.1 CENTRE_LOCATION

AIM element: compound_representation_item
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.30
Reference path: /ROOT_CLASS(compound_representation_item, 'centre location')/

5.1.17.1.1 longitudinal_location

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition = list_representation_item
list_representation_item [i] ->
representation_item =>
{representation_item. name = 'longitudinal location'}
value_representation_item
{value_representation_item. value_component ->
measure_value = length_measure}

5.1.17.1.2 transversal_location

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition = list_representation_item
list_representation_item [i] ->
representation_item =>
{representation_item. name = 'transversal location'}
value_representation_item
{value_representation_item. value_component ->
measure_value = length_measure}

5.1.17.1.3 vertical_location

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Reference path: compound_representation_item
compound_representation_item.item_element ->
compound_item_definition = list_representation_item
list_representation_item [i] ->
representation_item =>
{representation_item. name = 'vertical location'}
value_representation_item
{value_representation_item. value_component ->
measure_value = length_measure}

5.1.17.2 DERIVED_UNIT

AIM element: derived_unit
Source: ISO 10303-41
Reference path: (derived_unit
/NAME_ASSGN_WITH_VAL(derived_unit, ‘density unit’)/
derived_unit.elements[i] ->
[derived_unit_element
{[derived_unit_element.unit ->
([mass_unit][si_unit])
([mass_unit][conversion_based_unit])
([mass_unit][context_dependent_unit])]
[derived_unit_element.exponent = 1]}]
[derived_unit_element
{[derived_unit_element.unit ->
([length_unit][si_unit])
([length_unit][conversion_based_unit])
([length_unit][context_dependent_unit])]
[derived_unit_element.exponent = -3]}])
(derived_unit
/NAME_ASSGN_WITH_VAL(derived_unit, ‘dilatation unit’)/

```
derived_unit.elements[i] ->
[derived_unit_element
{[derived_unit_element.unit ->
([length_unit][si_unit])
([length_unit][conversion_based_unit])
([length_unit][context_dependent_unit])]
[derived_unit_element.exponent = 1]}]
[derived_unit_element
{[derived_unit_element.unit ->
([thermodynamic_temperature_unit][si_unit])
([thermodynamic_temperature_unit][conversion_based_unit])
([thermodynamic_temperature_unit][context_dependent_unit])]
[derived_unit_element.exponent = 1]}])
(derived_unit
/NAME_ASSGN_WITH_VAL(derived_unit, 'force unit')/
derived_unit.elements[i] ->
[derived_unit_element
{[derived_unit_element.unit ->
([mass_unit][si_unit])
([mass_unit][conversion_based_unit])
([mass_unit][context_dependent_unit])]
[derived_unit_element.exponent = 1]}]
[derived_unit_element
{[derived_unit_element.unit ->
([length_unit][si_unit])
([length_unit][conversion_based_unit])
([length_unit][context_dependent_unit])]
[derived_unit_element.exponent = 1]}]
[derived_unit_element
{[derived_unit_element.unit ->
([time_unit][si_unit])
([time_unit][conversion_based_unit])
([time_unit][context_dependent_unit])]
[derived_unit_element.exponent = -2]}])
(derived_unit
/NAME_ASSGN_WITH_VAL(derived_unit, 'inertia moment unit')/
derived_unit.elements[i] ->
derived_unit_element
{[derived_unit_element.unit ->
 ([length_unit][si_unit])
([length_unit][conversion_based_unit])
([length_unit][context_dependent_unit])]
[derived_unit_element.exponent = 4]})
(derived_unit
/NAME_ASSGN_WITH_VAL(derived_unit, 'moment unit')/
derived_unit.elements[i] ->
[derived_unit_element
{[derived_unit_element.unit ->
([mass_unit][si_unit])
([mass_unit][conversion_based_unit])
```



```
([mass_unit][context_dependent_unit])]
[derived_unit_element.exponent = 1]}]
[derived_unit_element
{[derived_unit_element.unit ->
([length_unit][si_unit])
([length_unit][conversion_based_unit])
([length_unit][context_dependent_unit])]
[derived_unit_element.exponent = 2]}]
[derived_unit_element
{[derived_unit_element.unit ->
([time_unit][si_unit])
([time_unit][conversion_based_unit])
([time_unit][context_dependent_unit])]
[derived_unit_element.exponent = -2]}])
(derived_unit
/NAME_ASSGN_WITH_VAL(derived_unit, 'section modulus unit')/
derived_unit.elements[i] ->
derived_unit_element
{[derived_unit_element.unit ->
 ([length_unit][si_unit])
([length_unit][conversion_based_unit])
([length_unit][context_dependent_unit])]
[derived_unit_element.exponent = 3]})
(derived_unit
/NAME_ASSGN_WITH_VAL(derived_unit, 'speed unit')/
derived_unit.elements[i] ->
[derived_unit_element
{[derived_unit_element.unit ->
([length_unit][si_unit])
([length_unit][conversion_based_unit])
([length_unit][context_dependent_unit])]
[derived_unit_element.exponent = 1]}]
[derived_unit_element
{[derived_unit_element.unit ->
([time_unit][si_unit])
([time_unit][conversion_based_unit])
([time_unit][context_dependent_unit])]
[derived_unit_element.exponent = -1]}])
(derived_unit
/NAME_ASSGN_WITH_VAL(derived_unit, 'stress unit')/
derived_unit.elements[i] ->
[derived_unit_element
{[derived_unit_element.unit ->
([length_unit][si_unit])
([length_unit][conversion_based_unit])
([length_unit][context_dependent_unit])]
[derived_unit_element.exponent = 1]}]
[derived_unit_element
{[derived_unit_element.unit ->
([mass_unit][si_unit])
```

([mass_unit][conversion_based_unit])
([mass_unit][context_dependent_unit])]
[derived_unit_element.exponent = 1]}]
[derived_unit_element
{[derived_unit_element.unit ->
([time_unit][si_unit])
([time_unit][conversion_based_unit])
([time_unit][context_dependent_unit])]
[derived_unit_element.exponent = -2]}])

5.1.17.3 NAMED_UNIT

AIM element: named_unit
Source: ISO 10303-41
Reference path: named_unit =>
[(si_unit)
(conversion_based_unit)
(context_dependent_unit)]
[(area_unit)
(length_unit)
(mass_unit)
(plane_angle_unit)
(ratio_unit)
(volume_unit)]

5.1.17.4 PRECISION

AIM element: uncertainty_measure_with_unit
Source: ISO 10303-43
Reference path:

5.1.17.4.1 point_coincidence

AIM element: measure_with_unit.value_component
Source: ISO 10303-43
Rules: 5.2.4.56
Reference path: uncertainty_measure_with_unit
{uncertainty_measure_with_unit.name = 'point coincidence'}
uncertainty_measure_with_unit <=
measure_with_unit
{[measure_with_unit.unit_component ->
unit
unit = named_unit
named_unit =>
([length_unit][si_unit])
([length_unit][conversion_based_unit])
([length_unit][expression_conversion_based_unit])
([length_unit][context_dependent_unit])]
[measure_with_unit.value_component >

measure_value = positive_length_measure]}

5.1.18 structural_features UoF

5.1.18.1 BEVEL_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.93
Reference path: {[/CLASS(product_definition_shape, 'bevel design definition', 'corner cutout design definition')/]
[/CLASS(product_definition_shape, 'corner cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.1.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.1.2 offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.265
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('bevel design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('offset', positive_length_measure)/

5.1.18.1.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.1.4 x_y_angle

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.265
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('bevel design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('x y angle', plane_angle_measure)/

5.1.18.1.5 z_depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.241
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('bevel design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('z depth', positive_length_measure)/

5.1.18.1.6 bevel_design_definition to feature (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{(/CLASS_ID(shape_aspect, 'corner cutout')/)
(/CLASS_ID(shape_aspect, 'edge feature')/)}

5.1.18.1.7 bevel_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.1.8 bevel_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.18.1.9 bevel_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.18.1.10 bevel_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.1.11 bevel_design_definition to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.1.12 bevel_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.1.13 bevel_design_definition to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.2 CIRCULAR_CUTOUT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.96
Reference path: {[/CLASS(product_definition_shape, 'circular cutout design definition', 'interior cutout design definition')/]
[/CLASS(product_definition_shape, 'interior cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.2.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.2.2 radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.179
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('circular cutout design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('radius', positive_length_measure)/

5.1.18.2.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.2.4 circular_cutout_design_definition to interior_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'interior cutout')/}

5.1.18.2.5 circular_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.2.6 circular_cutout_design_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.18.2.7 circular_cutout_design_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.18.2.8 circular_cutout_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.2.9 circular_cutout_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>

shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.2.10 circular_cutout_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.2.11 circular_cutout_design_definition to non_manifold_surface_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.3 CORNER_CUTOUT

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: {[/CLASS(shape_aspect, 'corner cutout', 'structural cutout')/]
[/CLASS(shape_aspect, 'structural cutout', 'structural feature')/]
[/CLASS(shape_aspect, 'structural feature', 'feature')/]
[/CLASS(shape_aspect, 'feature', 'item')/]
[/CLASS(shape_aspect, 'item', 'definable object')/]
[/ROOT_CLASS(shape_aspect, 'definable object')/]}

5.1.18.3.1 description

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.2)

5.1.18.3.2 name

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.1)

5.1.18.3.3 corner_cutout to external_reference (as documentation)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.3, 5.1.18.35.3, 5.1.18.35.3)

5.1.18.3.4 corner_cutout to global_id (as id)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.4)

5.1.18.3.5 corner_cutout to item (as parent)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.5)

5.1.18.3.6 corner_cutout to ship (as ship_context)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.6)

5.1.18.4 CORNER_CUTOUT_BOUNDARY_RELATIONSHIP

AIM element: group
Source: 218
Rules: 5.2.4.319, 5.2.4.326
Reference path: {[/CLASS(group, 'corner cutout boundary relationship', 'structural cutout boundary relationship')/]
[/CLASS(group, 'structural cutout boundary relationship', 'structural feature relationship')/]
[/CLASS(group, 'structural feature relationship', 'feature relationship')/]
[/CLASS(group, 'feature relationship', 'item relationship')/]
[/CLASS(group, 'item relationship', 'definable object')/]
[/CLASS(group, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(group, 'versionable object')/]}

5.1.18.4.1 boundary_index

AIM element: /SUPERTYPE(structural_cutout_boundary_relationship)/ (See 5.1.18.33)

5.1.18.4.2 boundary_index_2

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: group <-
/GROUPS(shape_aspect, 'boundary index 2')/
shape_aspect
[{/CLASS_ID(shape_aspect, 'border')/}]

[{shape_aspect.of_shape ->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{(/CLASS_ID(product_definition, 'panel system')/)
(/CLASS_ID(product_definition, 'plate')/)}]

5.1.18.4.3 description

AIM element: /SUPERTYPE(structural_cutout_boundary_relationship)/ (See 5.1.18.33)

5.1.18.4.4 version_id

AIM element: /SUPERTYPE(structural_cutout_boundary_relationship)/ (See 5.1.18.33)

5.1.18.4.5 corner_cutout_boundary_relationship to external_instance_-reference (as external_item_1)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect
{[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'corner cutout')/]
[/CLASS_ID(shape_aspect, 'corner cutout')/]}

5.1.18.4.6 corner_cutout_boundary_relationship to external_instance_-reference (as external_item_2)

AIM element: /SUPERTYPE(structural_cutout_boundary_relationship)/ (See 5.1.18.33)

5.1.18.4.7 corner_cutout_boundary_relationship to global_id (as id)

AIM element: /SUPERTYPE(structural_cutout_boundary_relationship)/ (See 5.1.18.33)

5.1.18.4.8 corner_cutout_boundary_relationship to corner_cutout (as item_1)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect
(/CLASS_ID(shape_aspect, 'corner cutout')/)

5.1.18.4.9 corner_cutout_boundary_relationship to item (as item_2)

AIM element: /SUPERTYPE(structural_cutout_boundary_relationship)/ (See 5.1.18.33)

5.1.18.5 CORNER_CUTOUT_DESIGN_DEFINITION

#1: Corner_cutout_design_definition is a Bevel_design_definition
#2: Corner_cutout_design_definition is a Rectangular_cutback_corner_-design_definition
#3: Corner_cutout_design_definition is a Round_corner_design_definition

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25)

5.1.18.5.1 description

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.1)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.1)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.1)

5.1.18.5.2 z_depth

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.5)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.6)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.4)

5.1.18.5.3 version_id

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.3)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.3)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.3)

5.1.18.5.4 corner_cutout_design_definition to feature (as defined_for)

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.6)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.7)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.5)

5.1.18.5.5 corner_cutout_design_definition to global_id (as id)

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.7)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.8)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.6)

5.1.18.5.6 corner_cutout_design_definition to derived_unit (as local_units)

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.8)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.9)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.7)

5.1.18.5.7 corner_cutout_design_definition to named_unit (as local_units)

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.9)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.10)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.8)

5.1.18.5.8 corner_cutout_design_definition to advanced_breb_shape (as representations)

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.10)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.11)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.9)

5.1.18.5.9 corner_cutout_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.11)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.12)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.10)

5.1.18.5.10 corner_cutout_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.12)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.13)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.11)

5.1.18.5.11 corner_cutout_design_definition to non_manifold_surface_-shape (as representations)

AIM element: #1: /SUBTYPE(Bevel_design_definition)/ (See 5.1.18.1.13)
#2: /SUBTYPE(Rectangular_cutback_corner_design_definition)/ (See 5.1.18.23.14)
#3: /SUBTYPE(Round_corner_design_definition)/ (See 5.1.18.25.12)

5.1.18.6 DRAIN_HOLE_CUTOUT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.102
Reference path: {[/CLASS(product_definition_shape, 'drain hole cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.6.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.6.2 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.184
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('drain hole cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('depth', positive_length_measure)/

5.1.18.6.3 drain_hole_radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.184
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('drain hole cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('drain hole radius', positive_length_measure)/

5.1.18.6.4 gap

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.184
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('drain hole cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('gap', positive_length_measure)/

5.1.18.6.5 gap_radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.184
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('drain hole cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('gap radius', positive_length_measure)/

5.1.18.6.6 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.184
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('drain hole cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.18.6.7 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.6.8 drain_hole_cutout_design_definition to edge_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'edge cutout')/}

5.1.18.6.9 drain_hole_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.6.10 drain_hole_cutout_design_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.18.6.11 drain_hole_cutout_design_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.6.12 drain_hole_cutout_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.6.13 drain_hole_cutout_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.6.14 drain_hole_cutout_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.6.15 drain_hole_cutout_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.7 EDGE_CUTOUT

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: {[/CLASS(shape_aspect, 'edge cutout', 'structural cutout')/]
[/CLASS(shape_aspect, 'structural cutout', 'structural feature')/]
[/CLASS(shape_aspect, 'structural feature', 'feature')/]
[/CLASS(shape_aspect, 'feature', 'item')/]
[/CLASS(shape_aspect, 'item', 'definable object')/]
[/ROOT_CLASS(shape_aspect, 'definable object')/]}

5.1.18.7.1 description

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.2)

5.1.18.7.2 name

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.1)

5.1.18.7.3 edge_cutout to external_reference (as documentation)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.3, 5.1.18.35.3, 5.1.18.35.3)

5.1.18.7.4 edge_cutout to global_id (as id)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.4)

5.1.18.7.5 edge_cutout to item (as parent)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.5)

5.1.18.7.6 edge_cutout to ship (as ship_context)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.6)

5.1.18.8 EDGE_CUTOUT_FUNCTIONAL_DEFINITION

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.153
Reference path: {[/CLASS(property_definition, 'edge cutout functional definition', 'functional definition')/]
[/CLASS(property_definition, 'functional definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.18.8.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: property_definition.description

5.1.18.8.2 the_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.357, 5.2.4.185
Reference path: property_definition = represented_definition <-
/PDR_NAME('edge cutout function parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('function')/
{(descriptive_representation_item.description = 'access hole')
(descriptive_representation_item.description = 'air escape')
(descriptive_representation_item.description = 'air liquid escape')
(descriptive_representation_item.description = 'liquid escape')
(descriptive_representation_item.description = 'penetration')
(descriptive_representation_item.description = 'user defined')}

5.1.18.8.3 user_def_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.225
Reference path: property_definition = represented_definition <-
/PDR_NAME('edge cutout function parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('user def function')/

5.1.18.8.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.18.8.5 edge_cutout_functional_definition to edge_cutout (as defined_-for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'edge cutout')/}

5.1.18.8.6 edge_cutout_functional_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: property_definition
identification_item = property_definition
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.9 EDGE_FEATURE

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: {[/CLASS(shape_aspect, 'edge feature', 'structural cutout')/]
[/CLASS(shape_aspect, 'structural cutout', 'structural feature')/]
[/CLASS(shape_aspect, 'structural feature', 'feature')/]
[/CLASS(shape_aspect, 'feature', 'item')/]
[/CLASS(shape_aspect, 'item', 'definable object')/]
[/ROOT_CLASS(shape_aspect, 'definable object')/]}

5.1.18.9.1 description

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.2)

5.1.18.9.2 name

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.1)

5.1.18.9.3 edge_feature to external_reference (as documentation)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.3, 5.1.18.35.3, 5.1.18.35.3)

5.1.18.9.4 edge_feature to global_id (as id)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.4)

5.1.18.9.5 edge_feature to item (as parent)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.5)

5.1.18.9.6 edge_feature to ship (as ship_context)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.6)

5.1.18.10 EDGE_FEATURE_FUNCTIONAL_DEFINITION

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.154
Reference path: {[/CLASS(property_definition, 'edge feature functional definition', 'functional definition')/]
[/CLASS(property_definition, 'functional definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.18.10.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path:
property_definition.description

5.1.18.10.2 the_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.357, 5.2.4.186
Reference path: property_definition = represented_definition <-
/PDR_NAME('edge feature function parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('function')/
{(descriptive_representation_item.description = 'bevel')
(descriptive_representation_item.description = 'chamfer')
(descriptive_representation_item.description = 'user defined')}

5.1.18.10.3 user_def_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.226
Reference path: property_definition = represented_definition <-
/PDR_NAME('edge feature function parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('user def function')/

5.1.18.10.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.18.10.5 edge_feature_functional_definition to edge_feature (as defined_for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'edge cutout')/}

5.1.18.10.6 edge_feature_functional_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: property_definition
identification_item = property_definition
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.11 ELLIPTICAL_CUTOUT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.103
Reference path: {[/CLASS(product_definition_shape, 'elliptical cutout design definition', 'interior cutout design definition')/]
[/CLASS(product_definition_shape, 'interior cutout design definition', 'feature design definition')/]

[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.11.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.11.2 half_axis_a

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.188
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('elliptical cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('half axis a', positive_length_measure)/

5.1.18.11.3 half_axis_b

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.188
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('elliptical cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('half axis b', positive_length_measure)/

5.1.18.11.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.11.5 elliptical_cutout_design_definition to interior_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'interior cutout')/}

5.1.18.11.6 elliptical_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.11.7 elliptical_cutout_design_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.18.11.8 elliptical_cutout_design_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.11.9 elliptical_cutout_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.11.10 elliptical_cutout_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.11.11 elliptical_cutout_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.11.12 elliptical_cutout_design_definition to non_manifold_surface_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=

property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.12 ELONGATED_OVAL_CUTOUT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-43
Rules: 5.2.4.104
Reference path: {[/CLASS(product_definition_shape, 'elongated oval cutout design definition', 'interior cutout design definition')/]
[/CLASS(product_definition_shape, 'interior cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.12.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.12.2 distance

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.189
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('elongated oval cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('distance', positive_length_measure)/

5.1.18.12.3 r1

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.189
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('elongated oval cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('r1', positive_length_measure)/

5.1.18.12.4 r2

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.189
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('elongated oval cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('r2', positive_length_measure)/

5.1.18.12.5 r3

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.227
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('elongated oval cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('r3', positive_length_measure)/

5.1.18.12.6 r3_x

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.227
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('elongated oval cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('r3 x', length_measure)/

5.1.18.12.7 r3_y

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.227
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('elongated oval cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('r3 y', length_measure)/

5.1.18.12.8 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.12.9 elongated_oval_cutout_design_definition to interior_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'interior cutout')/}

5.1.18.12.10 elongated_oval_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.12.11 elongated_oval_cutout_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.18.12.12 elongated_oval_cutout_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.12.13 elongated_oval_cutout_design_definition to advanced_brep_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.12.14 elongated_oval_cutout_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>

shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.12.15 elongated_oval_cutout_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.12.16 elongated_oval_cutout_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.13 FREE_FORM_INTERIOR_CUTOUT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.106
Reference path: {[/CLASS(product_definition_shape, 'free form interior cutout design definition', 'interior cutout design definition')/]
[/CLASS(product_definition_shape, 'interior cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.13.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.13.2 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.13.3 free_form_interior_cutout_design_definition to curve (as bounding_curve)

AIM element: curve
Source: ISO 10303-42
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('free form interior cutout design parameters')/ ->
representation
representation.items [i] ->
representation_item
{representation_item.name = 'bounding curve'} =>
geometric_representation_item =>
curve

5.1.18.13.4 free_form_interior_cutout_design_definition to interior_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'interior cutout')/}

5.1.18.13.5 free_form_interior_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.13.6 free_form_interior_cutout_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.18.13.7 free_form_interior_cutout_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.13.8 free_form_interior_cutout_design_definition to advanced_-brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.13.9 free_form_interior_cutout_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.13.10 free_form_interior_cutout_design_definition to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.13.11 free_form_interior_cutout_design_definition to non_-manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.14 INTERIOR_CUTOUT

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: {[/CLASS(shape_aspect, 'interior cutout', 'structural cutout')/]
[/CLASS(shape_aspect, 'structural cutout', 'structural feature')/]
[/CLASS(shape_aspect, 'structural feature', 'feature')/]
[/CLASS(shape_aspect, 'feature', 'item')/]
[/CLASS(shape_aspect, 'item', 'definable object')/]
[/ROOT_CLASS(shape_aspect, 'definable object')/]}

5.1.18.14.1 description

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.2)

5.1.18.14.2 name

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.1)

5.1.18.14.3 interior_cutout to external_reference (as documentation)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.3, 5.1.18.35.3, 5.1.18.35.3)

5.1.18.14.4 interior_cutout to global_id (as id)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.4)

5.1.18.14.5 interior_cutout to item (as parent)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.5)

5.1.18.14.6 interior_cutout to ship (as ship_context)

AIM element: /SUPERTYPE(Structural_cutout)/ (See 5.1.18.32.6)

5.1.18.15 INTERIOR_CUTOUT_DESIGN_DEFINITION

#1: Interior_cutout_design_definition is a Circular_cutout_design_definition
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_design_-definition
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39)

5.1.18.15.1 description

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2.1)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.1)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.1)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.1)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.1)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.1)

5.1.18.15.2 version_id

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2.3)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.4)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.8)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.2)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.4)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.8)

5.1.18.15.3 interior_cutout_design_definition to interior_cutout (as defined_for)

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2.4)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.5)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.9)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.4)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.5)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.9)

5.1.18.15.4 interior_cutout_design_definition to global_id (as id)

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2.5)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.6)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.10)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.5)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.6)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.10)

5.1.18.15.5 interior_cutout_design_definition to derived_unit (as local_-units)

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2.6)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.7)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.11)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.6)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.7)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.11)

5.1.18.15.6 interior_cutout_design_definition to named_unit (as local_-units)

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2.7)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.8)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.12)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.7)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.8)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.12)

5.1.18.15.7 interior_cutout_design_definition to advanced_brep_shape (as representations)

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.9)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.13)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.8)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.12)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.13)

5.1.18.15.8 interior_cutout_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2.9)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.10)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.14)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.9)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.9)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.14)

5.1.18.15.9 interior_cutout_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2.10)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.11)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.15)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.10)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.10)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.15)

5.1.18.15.10 interior_cutout_design_definition to non_manifold_surface_-shape (as representations)

AIM element: #1: Interior_cutout_design_definition is a Circular_cutout_design_definition -- (See 5.1.18.2.11)
#2: Interior_cutout_design_definition is an Elliptical_cutout_design_definition -- (See 5.1.18.11.12)
#3: Interior_cutout_design_definition is an Elongated_oval_cutout_design_definition -- (See 5.1.18.12.16)
#4: Interior_cutout_design_definition is a Free_form_interior_cutout_-design_definition -- (See 5.1.18.13.11)
#5: Interior_cutout_design_definition is a Rectangular_cutout_design_definition -- (See 5.1.18.24.11)
#6: Interior_cutout_design_definition is a Triangular_cutout_design_definition -- (See 5.1.18.39.16)

5.1.18.16 INTERIOR_CUTOUT_FUNCTIONAL_DEFINITION

AIM element: property_definition
Source: ISO 10303-43
Rules: 5.2.4.156
Reference path: {[/CLASS(property_definition, 'interior cutout functional definition', 'functional definition')/]
[/CLASS(property_definition, 'functional definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.18.16.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: property_definition.description

5.1.18.16.2 the_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.357, 5.2.4.194
Reference path: property_definition = represented_definition <-
/PDR_NAME('interior cutout function parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('function')/
{[(descriptive_representation_item.description = 'access hole')
(descriptive_representation_item.description = 'air escape')
(descriptive_representation_item.description = 'air liquid escape')
(descriptive_representation_item.description = 'foothold')
(descriptive_representation_item.description = 'lightening hole')
(descriptive_representation_item.description = 'liquid escape')
(descriptive_representation_item.description = 'penetration')
(descriptive_representation_item.description = 'user defined')]

5.1.18.16.3 user_def_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.228
Reference path: representation
representation.items[i] ->
/DSC_REP_ITEM('user def function') /

5.1.18.16.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.18.16.5 interior_cutout_functional_definition to interior_cutout (as defined_for)

AIM element: PATH
Reference path: property_definition.definition ->
characterized_definition =
shape_definition
shape_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'interior cutout')/}

5.1.18.16.6 interior_cutout_functional_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: property_definition
identification_item = property_definition
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.17 INWARD_ROUND_CORNER_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.110
Reference path: {[/CLASS(product_definition_shape, 'inward round corner design definition', 'round corner design definition')/]
[/CLASS(product_definition_shape, 'round corner cutout design definition', 'corner cutout design definition')/]
[/CLASS(product_definition_shape, 'corner cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.17.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.17.2 radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.195
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('inward round corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('radius', positive_length_measure)/

5.1.18.17.3 x_offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.195
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('inward round corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('x offset', length_measure)/

5.1.18.17.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.17.5 y_offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.195
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('inward round corner design parameters')/ ->
representation

representation.items [i] ->
/VAL_REP_ITEM('y offset', length_measure)/

5.1.18.17.6 z_depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.229
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('inward round corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('z depth', length_measure)/

5.1.18.17.7 inward_round_corner_design_definition to feature (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{(/CLASS_ID(shape_aspect, 'corner cutout')/)
(/CLASS_ID(shape_aspect, 'edge feature')/)}

5.1.18.17.8 inward_round_corner_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.17.9 inward_round_corner_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.18.17.10 inward_round_corner_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.17.11 inward_round_corner_design_definition to advanced_brep_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.17.12 inward_round_corner_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.17.13 inward_round_corner_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-

/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.17.14 inward_round_corner_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.18 OUTWARD_ROUND_CORNER_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.117
Reference path: {[/CLASS(product_definition_shape, 'outward round corner design definition', 'round corner design definition')/]
[/CLASS(product_definition_shape, 'round corner cutout design definition', 'corner cutout design definition')/]
[/CLASS(product_definition_shape, 'corner cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.18.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.18.2 radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.197
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('outward round corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('radius', positive_length_measure)/

5.1.18.18.3 x_offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.197
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('outward round corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('x offset', length_measure)/

5.1.18.18.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.18.5 y_offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.197
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('outward round corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('y offset', length_measure)/

5.1.18.18.6 z_depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.232
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('outward round corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('z depth', length_measure)/

5.1.18.18.7 outward_round_corner_design_definition to feature (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{(/CLASS_ID(shape_aspect, 'corner cutout')/)
(/CLASS_ID(shape_aspect, 'edge feature')/)}

5.1.18.18.8 outward_round_corner_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.18.9 outward_round_corner_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.18.18.10 outward_round_corner_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.18.11 outward_round_corner_design_definition to advanced_-brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.18.12 outward_round_corner_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.18.13 outward_round_corner_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-

/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.18.14 outward_round_corner_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.19 PART_EDGE_CUTOUT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.119
Reference path: {[/CLASS(product_definition_shape, 'part edge cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.19.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.19.2 r_left

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.198
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('part edge cutout design parameters')/ ->

representation
representation.items [i] ->
/VAL_REP_ITEM('r left', positive_length_measure)/

5.1.18.19.3 r_right

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.198
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('part edge cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('r right', positive_length_measure)/

5.1.18.19.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.19.5 x_a

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.198
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('part edge cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('x a', positive_length_measure)/

5.1.18.19.6 x_b

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.198
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('part edge cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('x b', length_measure)/

5.1.18.19.7 x_c

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.198
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('part edge cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('x c', length_measure)/

5.1.18.19.8 x_d

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.198
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('part edge cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('x d', positive_length_measure)/

5.1.18.19.9 y_b

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.198
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('part edge cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('y b', length_measure)/

5.1.18.19.10 y_c

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.198
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('part edge cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('y c', length_measure)/

5.1.18.19.11 part_edge_cutout_design_definition to edge_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'edge cutout')/}

5.1.18.19.12 part_edge_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.19.13 part_edge_cutout_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.18.19.14 part_edge_cutout_design_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.18.19.15 part_edge_cutout_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.19.16 part_edge_cutout_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.19.17 part_edge_cutout_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.19.18 part_edge_cutout_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.20 POSITION_FEATURE

#1: Position_feature is a Seam

AIM element: #1: /SUBTYPE(Seam)/ (See 5.1.18.28)

5.1.18.20.1 description

AIM element: #1: /SUBTYPE(Seam)/ (See 5.1.18.28.2)

5.1.18.20.2 name

AIM element: #1: /SUBTYPE(Seam)/ (See 5.1.18.28.1)

5.1.18.20.3 position_feature to external_reference (as documentation)

AIM element: #1: /SUBTYPE(Seam)/ (See 5.1.18.28.3)

5.1.18.20.4 position_feature to global_id (as id)

AIM element: #1: /SUBTYPE(Seam)/ (See 5.1.18.28.4)

5.1.18.20.5 position_feature to item (as parent)

AIM element: #1: /SUBTYPE(Seam)/ (See 5.1.18.28.5)

5.1.18.20.6 position_feature to ship (as ship_context)

AIM element: #1: /SUBTYPE(Seam)/ (See 5.1.18.28.6)

5.1.18.21 POSITION_FEATURE_DESIGN_DEFINITION

#1: Position_feature_design_definition is a Seam_design_definition

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30)

5.1.18.21.1 description

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.1)

5.1.18.21.2 version_id

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.3)

5.1.18.21.3 position_feature_design_definition to position_feature (as defined_for)

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.9)

5.1.18.21.4 position_feature_design_definition to global_id (as id)

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.5)

5.1.18.21.5 position_feature_design_definition to derived_unit (as local_-units)

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.6)

5.1.18.21.6 position_feature_design_definition to named_unit (as local_-units)

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.7)

5.1.18.21.7 position_feature_design_definition to advanced_breb_shape (as representations)

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.8)

5.1.18.21.8 position_feature_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.10)

5.1.18.21.9 position_feature_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.11)

5.1.18.21.10 position_feature_design_definition to non_manifold_surface_-shape (as representations)

AIM element: #1: /SUBTYPE(Seam_design_definition)/ (See 5.1.18.30.12)

5.1.18.22 POSITION_FEATURE_RELATIONSHIP

#1: Position_feature_relationship is a Seam_curve_relationship

AIM element: #1: /SUBTYPE(seam_curve_relationship)/ (See 5.1.18.29)

5.1.18.22.1 description

AIM element: #1: /SUBTYPE(Seam_curve_relationship)/ (See 5.1.18.29.2)

5.1.18.22.2 version_id

AIM element: #1: /SUBTYPE(Seam_curve_relationship)/ (See 5.1.18.29.4)

5.1.18.22.3 position_feature_relationship to external_instance_reference (as external_item_1)

AIM element: #1: /SUBTYPE(Seam_curve_relationship)/ (See 5.1.18.29.5)

5.1.18.22.4 position_feature_relationship to external_instance_reference (as external_item_2)

AIM element: #1: /SUBTYPE(Seam_curve_relationship)/ (See 5.1.18.29.6)

5.1.18.22.5 position_feature_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(Seam_curve_relationship)/ (See 5.1.18.29.7)

5.1.18.22.6 position_feature_relationship to structural_feature (as item_1)

AIM element: #1: /SUBTYPE(Seam_curve_relationship)/ (See 5.1.18.29.8)

5.1.18.22.7 position_feature_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(Seam_curve_relationship)/ (See 5.1.18.29.9)

5.1.18.23 RECTANGULAR_CUTBACK_CORNER_DESIGN_-DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.136
Reference path: {[/CLASS(product_definition_shape, 'rectangular cutback corner design definition', 'corner cutout design definition')/]
[/CLASS(product_definition_shape, 'corner cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.23.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.23.2 radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.201
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('rectangular cutback corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('radius', positive_length_measure)/

5.1.18.23.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.23.4 x_depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.201
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('rectangular cutback corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('x depth', positive_length_measure)/

5.1.18.23.5 y_depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.201
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('rectangular cutback corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('y depth', positive_length_measure)/

5.1.18.23.6 z_depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.238
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('rectangular cutback corner design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('y depth', positive_length_measure)/

5.1.18.23.7 rectangular_cutback_corner_design_definition to feature (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{(/CLASS_ID(shape_aspect, 'corner cutout')/)
(/CLASS_ID(shape_aspect, 'edge feature')/)}

5.1.18.23.8 rectangular_cutback_corner_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.23.9 rectangular_cutback_corner_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.18.23.10 rectangular_cutback_corner_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.23.11 rectangular_cutback_corner_design_definition to advanced_-brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.23.12 rectangular_cutback_corner_design_definition to edge_-based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.23.13 rectangular_cutback_corner_design_definition to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.23.14 rectangular_cutback_corner_design_definition to non_-manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.24 RECTANGULAR_CUTOUT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.123
Reference path: {[/CLASS(product_definition_shape, 'rectangular cutout design definition', 'interior cutout design definition')/]
[/CLASS(product_definition_shape, 'interior cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.24.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.24.2 length_of

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.202
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('rectangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('length of', positive_length_measure)/

5.1.18.24.3 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.202
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('rectangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.18.24.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.24.5 rectangular_cutout_design_definition to interior_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'interior cutout')/}

5.1.18.24.6 rectangular_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.24.7 rectangular_cutout_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.18.24.8 rectangular_cutout_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/

unit
unit = named_unit
named_unit

5.1.18.24.9 rectangular_cutout_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.24.10 rectangular_cutout_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.24.11 rectangular_cutout_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.24.12 rectangular_cutout_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.25 ROUND_CORNER_DESIGN_DEFINITION

#1: Round_corner_design_definition is an Inward_round_corner_design_definition
#2: Round_corner_design_definition is an Outward_round_corner_design_definition

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18)

5.1.18.25.1 description

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.1)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.1)

5.1.18.25.2 radius

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.2)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.2)

5.1.18.25.3 version_id

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.4)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.4)

5.1.18.25.4 z_depth

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.6)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.6)

5.1.18.25.5 round_corner_design_definition to feature (as defined_for)

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.7)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.7)

5.1.18.25.6 round_corner_design_definition to global_id (as id)

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.8)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.8)

5.1.18.25.7 round_corner_design_definition to derived_unit (as local_-units)

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.9)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.9)

5.1.18.25.8 round_corner_design_definition to named_unit (as local_units)

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.10)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.10)

5.1.18.25.9 round_corner_design_definition to advanced_brep_shape (as representations)

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.11)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.11)

5.1.18.25.10 round_corner_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.12)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.12)

5.1.18.25.11 round_corner_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.13)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.13)

5.1.18.25.12 round_corner_design_definition to non_manifold_surface_-shape (as representations)

AIM element: #1: /SUBTYPE(Inward_round_corner_design_definition)/ (See 5.1.18.17.14)
#2: /SUBTYPE(Outward_round_corner_design_definition)/ (See 5.1.18.18.14)

5.1.18.26 ROUND_CORNER_RECTANGULAR_CUTOUT_DESIGN_-DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.124
Reference path: {[/CLASS(product_definition_shape, 'round corner rectangular cutout design definition', 'rectangular cutout design definition')/]
[/CLASS(product_definition_shape, 'rectangular cutout design definition', 'interior cutout design definition')/]
[/CLASS(product_definition_shape, 'interior cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.26.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.26.2 corner_radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.204
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('round corner rectangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('corner radius', positive_length_measure)/

5.1.18.26.3 length_of

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.204
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('round corner rectangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('length of', positive_length_measure)/

5.1.18.26.4 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.204
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('round corner rectangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.18.26.5 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.26.6 round_corner_rectangular_cutout_design_definition to interior_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'interior cutout')/}

5.1.18.26.7 round_corner_rectangular_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.26.8 round_corner_rectangular_cutout_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=

property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.18.26.9 round_corner_rectangular_cutout_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.18.26.10 round_corner_rectangular_cutout_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.26.11 round_corner_rectangular_cutout_design_definition to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.26.12 round_corner_rectangular_cutout_design_definition to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.26.13 round_corner_rectangular_cutout_design_definition to non_-manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.27 ROUND_EDGE_RECTANGULAR_CUTOUT_DESIGN_-DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.125
Reference path: {[/CLASS(product_definition_shape, 'round edge rectangular cutout design definition', 'rectangular cutout design definition')/]
[/CLASS(product_definition_shape, 'rectangular cutout design definition', 'interior cutout design definition')/]
[/CLASS(product_definition_shape, 'interior cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.27.1 description

AIM element: property_definition.description

Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.27.2 distance

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.205
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('round edge rectangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('distance', length_measure)/

5.1.18.27.3 edge_radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.205
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('round edge rectangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('edge radius', positive_length_measure)/

5.1.18.27.4 length_of

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.205
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('round edge rectangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('length of', positive_length_measure)/

5.1.18.27.5 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.27.6 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.205
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('round edge rectangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.18.27.7 round_edge_rectangular_cutout_design_definition to interior_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'interior cutout')/}

5.1.18.27.8 round_edge_rectangular_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.27.9 round_edge_rectangular_cutout_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.18.27.10 round_edge_rectangular_cutout_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.27.11 round_edge_rectangular_cutout_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.27.12 round_edge_rectangular_cutout_design_definition to edge_-based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.27.13 round_edge_rectangular_cutout_design_definition to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-

/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.27.14 round_edge_rectangular_cutout_design_definition to non_-manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.28 SEAM

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: {[/CLASS(shape_aspect, 'seam', 'position feature')/]
[/CLASS(shape_aspect, 'position feature', 'structural feature')/]
[/CLASS(shape_aspect, 'structural feature', 'feature')/]
[/CLASS(shape_aspect, 'feature', 'item')/]
[/CLASS(shape_aspect, 'item', 'definable object')/]
[/ROOT_CLASS(shape_aspect, 'definable object')/]}

5.1.18.28.1 name

AIM element: shape_aspect.name
Source: ISO 10303-41

5.1.18.28.2 description

AIM element: shape_aspect.description
Source: ISO 10303-41

5.1.18.28.3 seam to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (shape_aspect
shape_aspect = external_identification_item

external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (shape_aspect
/DOC_REF(shape_aspect,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.18.28.4 seam to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.346
Reference path: shape_aspect
identification_item = shape_aspect <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.28.5 seam to panel_system (as parent)

AIM element: PATH
Reference path: shape_aspect <=
shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'parent'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'panel system')/}

5.1.18.28.6 seam to ship (as ship_context)

AIM element: PATH
Reference path: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition

{property_definition.name = 'ship context'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')/}

5.1.18.29 SEAM_CURVE_RELATIONSHIP

AIM element: shape_aspect_relationship
Source: ISO 10303-41
Rules: 5.2.4.341, 5.2.4.316
Reference path: {[/CLASS(shape_aspect_relationship, 'seam curve relationship', 'position feature relationship')/]
[/CLASS(shape_aspect_relationship, 'position feature relationship', 'structural feature relationship')/]
[/CLASS(shape_aspect_relationship, 'structural feature relationship', 'feature relationship')/]
[/CLASS(shape_aspect_relationship, 'feature relationship', 'item relationship')/]
[/CLASS(shape_aspect_relationship, 'item relationship', 'definable object')/]
[/CLASS(shape_aspect_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(shape_aspect_relationship, 'versionable object')/]
[/ROOT_CLASS(shape_aspect_relationship, 'definable object')/]}

5.1.18.29.1 boundary_index

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: shape_aspect_relationship
shape_aspect_relationship.related_shape_aspect ->
shape_aspect
{[/CLASS_ID(shape_aspect, 'border')/]
[shape_aspect.of_shape ->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'panel system')/}]}

5.1.18.29.2 description

AIM element: shape_aspect_relationship.description
Source: ISO 10303-41

5.1.18.29.3 displacement

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.239
Reference path: shape_aspect_relationship
represented_definition = shape_aspect_relationship
represented_definition <-
/PDR_NAME('seam curve relationship parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('displacement', length_measure)/

5.1.18.29.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(shape_aspect_relationship)/

5.1.18.29.5 seam_curve_relationship to external_instance_reference (as external_item_1)

AIM element: PATH
Reference path: shape_aspect_relationship
shape_aspect_relationship.relating_shape_aspect ->
shape_aspect
{[/CLASS_ID(shape_aspect, 'seam')/]
[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'seam')/]}

5.1.18.29.6 seam_curve_relationship to external_instance_reference (as external_item_2)

#1: external_item_2 is a seam
#2: external_item_2 is the border of a panel_system

AIM element: PATH
Reference path: #1: shape_aspect_relationship
shape_aspect_relationship.related_shape_aspect ->
shape_aspect
{[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'seam')/]
[/CLASS_ID(shape_aspect, 'seam')/]}

#2: shape_aspect_relationship
shape_aspect_relationship.related_shape_aspect ->
shape_aspect
{[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'border')/]
[/CLASS_ID(shape_aspect, 'border')/]}

5.1.18.29.7 seam_curve_relationship to global_id (as id)

AIM element: PATH
Source: ISO 10303-218
Rules: 5.2.4.56, 5.2.4.344
Reference path: shape_aspect_relationship
identification_item= shape_aspect_relationship <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.29.8 seam_curve_relationship to position_feature (as item_1)

AIM element: PATH
Reference path: shape_aspect_relationship
shape_aspect_relationship.relating_shape_aspect ->
shape_aspect
{/CLASS_ID(shape_aspect, 'seam')/}

5.1.18.29.9 seam_curve_relationship to seam or panel_system (as item_2)

#1: item_2 is a seam
#2: item_2 is the border of a panel_system

AIM element: PATH
Reference path: #1: shape_aspect_relationship
shape_aspect_relationship.related_shape_aspect ->
shape_aspect
{/CLASS_ID(shape_aspect, 'seam')/}

#2: shape_aspect_relationship
shape_aspect_relationship.related_shape_aspect ->
shape_aspect
{/CLASS_ID(shape_aspect, 'border')/}
shape_aspect.of_shape ->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'panel system')/}

5.1.18.30 SEAM_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition_shape, 'seam design definition', 'position feature design definition')/]
[/CLASS(product_definition_shape, 'position feature design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]

[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.30.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.30.2 seam_curve

#1: if the seam curve is a curve
#2: if the seam curve is a seam curve relationship

AIM element: #1: (curve)
#2: (shape_aspect_relationship)
Source: ISO 10303-42
ISO 10303-43
Rules: #1: 5.2.4.255
#2:
Reference path: #1: (product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('seam design parameters')/ ->
representation
representation.items [i] ->
representation_item
{representation_item.name = 'seam curve'} =>
geometric_representation_item =>
curve)

#2: (product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_definition
shape_definition = shape_aspect
shape_aspect
{({/CLASS_ID(shape_aspect, 'seam')/})
({/CLASS_ID(shape_aspect, 'border')}
shape_aspect.of_shape ->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'panel system')/})} <-
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'seam curve relationship')/}

5.1.18.30.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.30.4 seam_design_definition to seam (as border)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_definition
shape_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'seam')/}

5.1.18.30.5 seam_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.30.6 seam_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.18.30.7 seam_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.18.30.8 seam_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.30.9 seam_design_definition to seam (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'seam')/}

5.1.18.30.10 seam_design_definition to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.30.11 seam_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/

property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.30.12 seam_design_definition to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.18.31 SHEAR_BEVEL_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.126
Reference path: {[/CLASS(product_definition_shape, 'shear bevel design definition', 'bevel design definition')/]
[/CLASS(product_definition_shape, 'bevel design definition', 'corner cutout design definition')/]
[/CLASS(product_definition_shape, 'corner cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.31.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.31.2 offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.206

Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('shear bevel design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('offset', positive_length_measure)/

5.1.18.31.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.31.4 x_y_angle

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.206
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('shear bevel design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('x y angle', plane_angle_measure)/

5.1.18.31.5 x_z_angle

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.206
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('shear bevel design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('x y angle', plane_angle_measure)/

5.1.18.31.6 z_depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.240
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('shear bevel design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('z depth', positive_length_measure)/

5.1.18.31.7 shear_bevel_design_definition to feature (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{(/CLASS_ID(shape_aspect, 'corner cutout')/)
(/CLASS_ID(shape_aspect, 'edge feature')/)}

5.1.18.31.8 shear_bevel_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.31.9 shear_bevel_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.18.31.10 shear_bevel_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.31.11 shear_bevel_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition

represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.31.12 shear_bevel_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.31.13 shear_bevel_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.31.14 shear_bevel_design_definition to non_manifold_surface_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>

non_manifold_surface_shape_representation

5.1.18.32 STRUCTURAL_CUTOUT

#1: Structural_cutout is a Corner_cutout
#2: Structural_cutout is a Edge_cutout
#3: Structural_cutout is an Edge_feature
#4: Structural_cutout is an Interior_cutout

AIM element: #1: /SUBTYPE(corner_cutout)/ (See 5.1.18.3)
#2: /SUBTYPE(edge_cutout)/ (See 5.1.18.7)
#3: /SUBTYPE(edge_feature)/ (See 5.1.18.9)
#4: /SUBTYPE(interior_cutout)/ (See 5.1.18.14)

5.1.18.32.1 name

AIM element: shape_aspect.name
Source: ISO 10303-41

5.1.18.32.2 description

AIM element: shape_aspect.description
Source: ISO 10303-41

5.1.18.32.3 structural_cutout to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (shape_aspect
shape_aspect = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (shape_aspect
/DOC_REF(shape_aspect,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.18.32.4 structural_cutout to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.346
Reference path: shape_aspect
identification_item = shape_aspect <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.32.5 structural_cutout to item (as parent)

Item is a Structural_part or Structural_system

AIM element: PATH
Reference path: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'parent'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{(/CLASS_ID(product_definition, 'structural system')/)
(/CLASS_ID(product_definition, 'structural part')/)}
(/CLASS_ID(product_definition_relationship, 'structural part penetration
relationship')/)}

5.1.18.32.6 structural_cutout to ship (as ship_context)

AIM element: PATH
Reference path: shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'parent'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')/}

5.1.18.33 STRUCTURAL_CUTOUT_BOUNDARY_RELATIONSHIP

AIM element: group
Source: ISO 10303-41
Rules: 5.2.4.325, 5.2.4.337
Reference path: {[/CLASS(group, 'structural cutout boundary relationship', 'structural feature relationship')/]
[/CLASS(group, 'structural feature relationship', 'feature relationship')/]
[/CLASS(group, 'feature relationship', 'item relationship')/]
[/CLASS(group, 'item relationship', 'definable object')/]
[/CLASS(group, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(group, 'versionable object')/]
[/ROOT_CLASS(group, 'definable object')/]}

5.1.18.33.1 boundary_index

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: group <-
/GROUPS(shape_aspect, 'boundary index')/
shape_aspect
[{/CLASS_ID(shape_aspect, 'border')/}]
[shape_aspect.of_shape ->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{(/CLASS_ID(product_definition, 'panel system')/)
(/CLASS_ID(product_definition, 'plate')/)}]

5.1.18.33.2 description

AIM element: group.description
Source: ISO 10303-41

5.1.18.33.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(group)

5.1.18.33.4 structural_cutout_boundary_relationship to external_-instance_reference (as external_item_1)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect

{[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'structural cutout')/]
[/CLASS_ID(shape_aspect, 'structural cutout')/]}

5.1.18.33.5 structural_cutout_boundary_relationship to external_-instance_reference (as external_item_2)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect
shape_aspect.of_shape
product_definition_shape <=
property_definition
{property_definition.name = 'item 2'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{([/EXT_INST_REF(product_definition, 'ship structures schema', 'plate')/]
[/CLASS_ID(product_definition, 'plate')/])
([/EXT_INST_REF(product_definition, 'ship structures schema', 'panel system')/]
[/CLASS_ID(product_definition, 'panel system')/])}

5.1.18.33.6 structural_cutout_boundary_relationship to global_id (as id)

AIM element: PATH
Source: ISO 10303-218
Rules: 5.2.4.56, 5.2.4.57
Reference path: group
identification_item = group <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.33.7 structural_cutout_boundary_relationship to structural_cutout (as item_1)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect
{/CLASS_ID(shape_aspect, 'structural cutout')/}

5.1.18.33.8 structural_cutout_boundary_relationship to item (as item_2)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect

shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'item 2'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{(/CLASS_ID(product_definition, 'plate')/)
(/CLASS_ID(product_definition, 'panel system')/)}

5.1.18.34 STRUCTURAL_CUTOUT_MANUFACTURING_RELATIONSHIP

AIM element: group
Source: ISO 10303-41
Rules:
Reference path: {[/CLASS(group, 'structural cutout manufacturing relationship', 'structural feature relationship')/]
[/CLASS(group, 'structural feature relationship', 'feature relationship')/]
[/CLASS(group, 'feature relationship', 'item relationship')/]
[/CLASS(group, 'item relationship', 'definable object')/]
[/CLASS(group, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(group, 'versionable object')/]
[/ROOT_CLASS(group, 'definable object')/]}

5.1.18.34.1 outer_contour_index

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: group <-
/GROUPS(shape_aspect, 'boundary index')/
shape_aspect
[{/CLASS_ID(shape_aspect, 'border')/}]
[shape_aspect.of_shape ->
product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'plate')/}]

5.1.18.34.2 structural_cutout_manufacturing_relationship to structural_cutout (as item_1)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect
{/CLASS_ID(shape_aspect, 'structural cutout')/}

5.1.18.34.3 description

AIM element: group.description
Source: ISO 10303-41

5.1.18.34.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.368
Reference path: /VERSION_ID(group)/

5.1.18.34.5 structural_cutout_manufacturing_relationship to external_instance_reference (as external_item_1)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect
{[/EXT_INST_REF(shape_aspect, 'ship_structures_schema', 'structural cutout')/]
[/CLASS_ID(shape_aspect, 'structural cutout')/]}

5.1.18.34.6 structural_cutout_manufacturing_relationship to external_instance_reference (as external_item_2)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect
shape_aspect.of_shape
product_definition_shape <=
property_definition
{property_definition.name = 'item 2'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{[/EXT_INST_REF(product_definition, 'ship_structures_schema', 'plate')/]
[/CLASS_ID(product_definition, 'plate')/]}

5.1.18.34.7 structural_cutout_manufacturing_relationship to global_id (as id)

AIM element: PATH
Source: ISO 10303-218
Rules: 5.2.4.55, 5.2.4.56
Reference path: group

identification_item = group <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.34.8 structural_cutout_manufacturing_relationship to item (as item_2)

AIM element: PATH
Reference path: group <-
/GROUPS(shape_aspect, 'item 1')/
shape_aspect
shape_aspect.of_shape ->
product_definition_shape <=
property_definition
{property_definition.name = 'item 2'}
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'plate')/)

5.1.18.35 STRUCTURAL_FEATURE

#1: Structural_feature is a Position_feature
#2: Structural_feature is a Structural_cutout
#3: Structural_feature is a Structural_manufacturing_feature

AIM element: #1: /SUBTYPE(Position_feature)/ (See 5.1.18.20)
#2: /SUBTYPE(Structural_cutout)/ (See 5.1.18.32)
#3: /SUBTYPE(Structural_manufacturing_feature)/ (See 5.1.15.10)

5.1.18.35.1 description

AIM element: #1: /SUBTYPE(Position_feature)/ (See 5.1.18.20.1)
#2: /SUBTYPE(Structural_cutout)/ (See 5.1.18.32.2)
#3: /SUBTYPE(Structural_manufacturing_feature)/ (See 5.1.15.10.1)

5.1.18.35.2 name

AIM element: #1: /SUBTYPE(Position_feature)/ (See 5.1.18.20.2)
#2: /SUBTYPE(Structural_cutout)/ (See 5.1.18.32.1)
#3: /SUBTYPE(Structural_manufacturing_feature)/ (See 5.1.15.10.2)

5.1.18.35.3 structural_feature to external_reference (as documentation)

AIM element: #1: /SUBTYPE(Position_feature)/ (See 5.1.18.20.3)
#2: /SUBTYPE(Structural_cutout)/ (See 5.1.18.32.3, 5.1.18.35.3, 5.1.18.35.3)
#3: /SUBTYPE(Structural_manufacturing_feature)/ (See 5.1.15.10.4)

5.1.18.35.4 structural_feature to global_id (as id)

AIM element: #1: /SUBTYPE(Position_feature)/ (See 5.1.18.20.4)
#2: /SUBTYPE(Structural_cutout)/ (See 5.1.18.32.4)
#3: /SUBTYPE(Structural_manufacturing_feature)/ (See 5.1.15.10.6)

5.1.18.35.5 structural_feature to item (as parent)

AIM element: #1: /SUBTYPE(Position_feature)/ (See 5.1.18.20.5)
#2: /SUBTYPE(Structural_cutout)/ (See 5.1.18.32.5)
#3: /SUBTYPE(Structural_manufacturing_feature)/ (See 5.1.15.10.3)

5.1.18.35.6 structural_feature to ship (as ship_context)

AIM element: #1: /SUBTYPE(Position_feature)/ (See 5.1.18.20.6)
#2: /SUBTYPE(Structural_cutout)/ (See 5.1.18.32.6)
#3: /SUBTYPE(Structural_manufacturing_feature)/ (See 5.1.15.10.5)

5.1.18.36 STRUCTURAL_FEATURE_RELATIONSHIP

#1: Structural_feature_relationship is a Position_feature_relationship
#2: Structural_feature_relationship is a Structural_cutout_boundary_relationship
#3: Structural_feature_relationship is a Structural_added_material_boundary_-relationship

AIM element: #1: /SUBTYPE(Position_feature_relationship)/ (See 5.1.18.22)
#2: /SUBTYPE(Structural_cutout_boundary_relationship)/ (See 5.1.18.33)
#3: /SUBTYPE(Structural_added_material_boundary_relationship)/ (See 5.1.15.7)

5.1.18.36.1 description

AIM element: #1: /SUBTYPE(Position_feature_relationship)/ (See 5.1.18.22.1)
#2: /SUBTYPE(Structural_cutout_boundary_relationship)/ (See 5.1.18.33.3)
#3: /SUBTYPE(Structural_added_material_boundary_relationship)/ (See 5.1.15.7.2)

5.1.18.36.2 version_id

AIM element: #1: /SUBTYPE(Position_feature_relationship)/ (See 5.1.18.22.2)
#2: /SUBTYPE(Structural_cutout_boundary_relationship)/ (See 5.1.18.33.3)
#3: /SUBTYPE(Structural_added_material_boundary_relationship)/ (See 5.1.15.7.3)

5.1.18.36.3 structural_feature_relationship to external_instance_reference (as external_item_1)

AIM element: #1: /SUBTYPE(Position_feature_relationship)/ (See 5.1.18.22.3)
#2: /SUBTYPE(Structural_cutout_boundary_relationship)/ (See 5.1.18.33.4)
#3: /SUBTYPE(Structural_added_material_boundary_relationship)/ (See 5.1.15.7.5)

5.1.18.36.4 structural_feature_relationship to external_instance_reference (as external_item_2)

AIM element: #1: /SUBTYPE(Position_feature_relationship)/ (See 5.1.18.22.4)
#2: /SUBTYPE(Structural_cutout_boundary_relationship)/ (See 5.1.18.33.5)
#3: /SUBTYPE(Structural_added_material_boundary_relationship)/ (See 5.1.15.7.6)

5.1.18.36.5 structural_feature_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(Position_feature_relationship)/ (See 5.1.18.22.5)
#2: /SUBTYPE(Structural_cutout_boundary_relationship)/ (See 5.1.18.33.6)
#3: /SUBTYPE(Structural_added_material_boundary_relationship)/ (See 5.1.18.33.6)

5.1.18.36.6 structural_feature_relationship to structural_feature (as item_1)

AIM element: #1: /SUBTYPE(Position_feature_relationship)/ (See 5.1.18.22.6)
#2: /SUBTYPE(Structural_cutout_boundary_relationship)/ (See 5.1.18.33.7)
#3: /SUBTYPE(Structural_added_material_boundary_relationship)/ (See 5.1.15.7.4)

5.1.18.36.7 structural_feature_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(Position_feature_relationship)/ (See 5.1.18.22.7)
#2: /SUBTYPE(Structural_cutout_boundary_relationship)/ (See 5.1.18.33.8)
#3: /SUBTYPE(Structural_added_material_boundary_relationship)/ (See 5.1.15.7.7)

5.1.18.37 STRUCTURAL_PART_PENETRATION_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Reference path: product_definition_relationship
{[/CLASS(product_definition_relationship, 'structural part penetration relationship', 'item relationship')/]
[/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'definable object')/]}

5.1.18.37.1 description

AIM element: product_definition_relationship.description

5.1.18.37.2 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41

Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.18.37.3 structural_part_penetration_relationship to external_-instance_reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
{[/CLASS(product_definition, 'structural part')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'structural part')/]}}

5.1.18.37.4 structural_part_penetration_relationship to external_-instance_reference (as external_item_2)

AIM element: product_definition_relationship.related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.related_product_definition ->
product_definition
{([/CLASS(product_definition, 'part')/] [/EXT_INST_REF(product_definition,
'ship structures schema', 'part')/])
([/CLASS(product_definition, 'system')/] [/EXT_INST_REF(product_definition,
'ship structures schema', 'system')/])}}

5.1.18.37.5 structural_part_penetration_relationship to global_id (as id)

AIM element: PATH
Source: ISO 10303-41
Rules: 5.2.4.56, 5.2.4.91
Reference path: product_definition_relationship
identification_item = product_definition_relationship <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.37.6 structural_part_penetration_relationship to structural_part (as item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, 'structural part')/}}

5.1.18.37.7 structural_part_penetration_relationship to item (as item_2)

AIM element: product_definition_relationship.related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.related_product_definition ->
product_definition
{(/CLASS_ID(product_definition, 'part')/)
(/CLASS_ID(product_definition, 'system')/)}}

5.1.18.37.8 structural_part_penetration_relationship to structural_cutout (as penetration_result)

AIM element: PATH
Reference path: product_definition_relationship
characterized_product_definition = product_definition_relationship
characterized_product_definition
characterized_definition = characterized_product_definition
characterized_definition <-
property_definition.definition
property_definition
{property_definition.name = 'penetration result'} =>
product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{/CLASS_ID(shape_aspect, 'structural cutout')/}

5.1.18.38 STRUCTURAL_SYSTEM_PENETRATION_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Reference path: product_definition_relationship
{[/CLASS(product_definition_relationship, 'structural system penetration relationship', 'item relationship')/]
[/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]
[/ROOT_CLASS(shape_aspect_relationship, 'definable object')/]}

5.1.18.38.1 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.18.38.2 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369

Reference path: /VERSION_ID(product_definition_relationship)

5.1.18.38.3 structural_system_penetration_relationship to external_-instance_reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
{[/CLASS(product_definition, 'structural system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'structural system')/]}}

5.1.18.38.4 structural_system_penetration_relationship to external_-instance_reference (as external_item_2)

AIM element: product_definition_relationship.related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.related_product_definition ->
product_definition
{([/CLASS(product_definition, 'part')/] [/EXT_INST_REF(product_definition, 'ship structures schema', 'part')/])
([/CLASS(product_definition, 'system')/] [/EXT_INST_REF(product_definition, 'ship structures schema', 'system')/])}}

5.1.18.38.5 structural_system_penetration_relationship to global_id (as id)

AIM element: PATH
Source: ISO 10303-41
Rules: 5.2.4.56, 5.2.4.91
Reference path: product_definition_relationship
identification_item = product_definition_relationship <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.38.6 structural_system_penetration_relationship to structural_-system (as item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, 'structural system')/}}

5.1.18.38.7 structural_system_penetration_relationship to item (as item_2)

AIM element: product_definition_relationship.related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.related_product_definition ->
product_definition
{(/CLASS_ID(product_definition, 'part')/)
(/CLASS_ID(product_definition, 'system')/)}}

5.1.18.38.8 structural_system_penetration_relationship to structural_-cutout (as penetration_result)

AIM element: PATH
Reference path: product_definition_relationship
characterized_product_definition = product_definition_relationship
characterized_product_definition
characterized_definition = characterized_product_definition
characterized_definition <-
property_definition.definition
property_definition
{property_definition.name = 'penetration result'} =>
product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{/CLASS_ID(shape_aspect, 'structural cutout')/}

5.1.18.39 TRIANGULAR_CUTOUT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.131
Reference path: {[/CLASS(product_definition_shape, 'triangular cutout design definition', 'interior cutout design definition')/]
[/CLASS(product_definition_shape, 'interior cutout design definition', 'feature design definition')/]
[/CLASS(product_definition_shape, 'feature design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.18.39.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.18.39.2 c1_radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.213
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('triangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('c1 radius', positive_length_measure)/

5.1.18.39.3 c2_radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.213
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('triangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('c2 radius', positive_length_measure)/

5.1.18.39.4 c3_radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.213
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('triangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('c3 radius', positive_length_measure)/

5.1.18.39.5 c2_x

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.213
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('triangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('c2 x', positive_length_measure)/

5.1.18.39.6 c3_x

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.213
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('triangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('c3 x', length_measure)/

5.1.18.39.7 c3_y

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.213
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('triangular cutout design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('c3 y', positive_length_measure)/

5.1.18.39.8 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.18.39.9 triangular_cutout_design_definition to interior_cutout (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = shape_aspect
shape_aspect
{/CLASS_ID(shape_aspect, 'interior cutout')/}

5.1.18.39.10 triangular_cutout_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape

identification_item = product_definition_shape
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.18.39.11 triangular_cutout_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.18.39.12 triangular_cutout_design_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.18.39.13 triangular_cutout_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.18.39.14 triangular_cutout_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition

represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.18.39.15 triangular_cutout_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.18.39.16 triangular_cutout_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.19 structural_parts UoF

5.1.19.1 ANGLE_BAR_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.317
Reference path: {[/CLASS(shape_aspect, 'angle bar cross section', 'angle profile cross section')/]
[/CLASS(shape_aspect, 'angle profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.1.1 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.175
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('angle bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('depth', positive_length_measure)/

5.1.19.1.2 designation

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.1)

5.1.19.1.3 radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.175
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('angle bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('radius', positive_length_measure)/

5.1.19.1.4 thk

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.175
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('angle bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('thk', positive_length_measure)/

5.1.19.1.5 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.175

Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('angle bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.19.1.6 angle_bar_cross_section to section_properties (as section_-properties)

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.2)

5.1.19.2 ANGLE_PROFILE_CROSS_SECTION

#1: angle_profile_cross_section is an angle_bar_cross_section

AIM element: #1: /SUBTYPE(angle_bar_cross_section)/ (See 5.1.19.1)

5.1.19.2.1 designation

AIM element: #1: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.2.2 angle_profile_cross_section to section_properties (as section_-properties)

AIM element: #1: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.3 BAR_PROFILE_CROSS_SECTION

#1: bar_profile_cross_section is a bulbflat_cross_section
#2: bar_profile_cross_section is a flat_bar_cross_section
#3: bar_profile_cross_section is a round_bar_cross_section

AIM element: #1: /SUBTYPE(bulbflat_cross_section)/ (See 5.1.19.4)
#2: /SUBTYPE(flat_bar_cross_section)/ (See 5.1.19.14)
#3: /SUBTYPE(round_bar_cross_section)/ (See) 5.1.19.32)

5.1.19.3.1 designation

AIM element: #1, #2, #3: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.3.2 bar_profile_cross_section to section_properties (as section_-properties)

AIM element: #1, #2, #3: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.4 BULBFLAT_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.320
Reference path: {[/CLASS(shape_aspect, 'bulbflat cross section', 'bar profile cross section')/]
[/CLASS(shape_aspect, 'bar profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.4.1 bulb_radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.177
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('bulbflat cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('bulb radius', positive_length_measure)/

5.1.19.4.2 bulb_width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.177
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('bulbflat cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('bulb width', positive_length_measure)/

5.1.19.4.3 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.177
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('bulbflat cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('depth', positive_length_measure)/

5.1.19.4.4 designation

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.4.5 tw

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.177
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('bulbflat cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('tw', positive_length_measure)/

5.1.19.4.6 bulbflat_profile_cross_section to section_properties (as section_-properties)

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.2)

5.1.19.5 CHANNEL_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.321
Reference path: {[/CLASS(shape_aspect, 'channel cross section', 'channel profile cross section')/]
[/CLASS(shape_aspect, 'channel profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.5.1 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.178
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('channel cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('depth', positive_length_measure)/

5.1.19.5.2 designation

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.1)

5.1.19.5.3 flange_thk

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.178
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('channel cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('flange thk', positive_length_measure)/

5.1.19.5.4 k

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.178
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('channel cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('k', positive_length_measure)/

5.1.19.5.5 radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.178
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('channel cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('radius', positive_length_measure)/

5.1.19.5.6 web_thk

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.178
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('channel cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('web thk', positive_length_measure)/

5.1.19.5.7 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.178
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('channel cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.19.5.8 channel_cross_section to section_properties (as section_-properties)

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.2)

5.1.19.6 CHANNEL_PROFILE_CROSS_SECTION

#1: channel_profile_cross_section is a channel_cross_section

AIM element: #1: /SUBTYPE(channel_cross_section)/ (See 5.1.19.5)

5.1.19.6.1 designation

AIM element: #1: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.6.2 channel_profile_cross_section to section_properties (as section_-properties)

AIM element: #1: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.2)

5.1.19.7 CIRCULAR_HOLLOW_PROFILE_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.322
Reference path: {[/CLASS(shape_aspect, 'circular hollow profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.7.1 designation

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.1)

5.1.19.7.2 inner_diameter

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.180
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('circular hollow profile cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('inner diameter', positive_length_measure)/

5.1.19.7.3 outer_diameter

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.180
Reference path: shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('circular hollow profile cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('outer diameter', positive_length_measure)/

5.1.19.7.4 circular_hollow_profile_cross_section to section_properties (as section_properties)

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.8 CORRUGATED_PART

AIM element: product_definition
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition, 'corrugated_part', 'structural part')/]
[/CLASS(product_definition, 'structural part', 'part')/]
[/CLASS(product_definition, 'part', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]}

5.1.19.8.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.19.8.2 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.19.8.3 corrugated_part to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
product_definition = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.19.8.4 corrugated_part to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.8.5 corrugated_part to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')/}

5.1.19.9 CORRUGATED_PART_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.99
Reference path: {[/CLASS(product_definition_shape, 'corrugated part design definition', 'structural part design definition')/]
[/CLASS(product_definition_shape, 'structural part design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.19.9.1 corrugated_part_design_definition to corrugated_structure_boundary_relationship (as border)

NOTE Shape_aspect.description should specify the index of the corrugated_structure_boundary_relationship in the border list

AIM element: PATH
Rules: 5.2.4.134
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship

{/CLASS_ID(shape_aspect_relationship, 'corrugated structure boundary relationship')/}]}

5.1.19.9.2 corrugated_part_design_definition to structural_system_adjacency_relationship (as border)

NOTE Shape_aspect.description should specify the index of the structural_system_adjacency_relationship in the border list

AIM element: PATH
Rules: 5.2.4.134
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'structural system adjacency relationship')/}]}

5.1.19.9.3 corrugated_part_design_definition to surface (as border)

NOTE Shape_aspect.description should specify the index of the surface in the border list

AIM element: PATH
Rules: 5.2.4.134
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect
represented_definition = shape_aspect
represented_definition <-
/PDR_NAME('border representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', surface)/]}

5.1.19.9.4 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.19.9.5 mass

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.222
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('corrugated part design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('mass', mass_measure)/

5.1.19.9.6 mirrored

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.182
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('corrugated part design parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('mirrored')/
{(descriptive_representation_item.description = 'TRUE')
(descriptive_representation_item.description = 'FALSE')}

5.1.19.9.7 offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.182
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('corrugated part design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('offset', length_measure)/

5.1.19.9.8 repetition

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.182
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('corrugated part design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('repetition', count_measure)/

5.1.19.9.9 thickness

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.182
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/PDR_NAME('corrugated part design parameters')/ ->
representation
representation.items[I] ->
/VAL_REP_ITEM('thickness', positive_length_measure)/

5.1.19.9.10 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.19.9.11 corrugated_part_design_definition to corrugated_part (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'corrugated part')/}

5.1.19.9.12 corrugated_part_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.9.13 corrugated_part_design_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.19.9.14 corrugated_part_design_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.19.9.15 corrugated_part_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.19.9.16 corrugated_part_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.19.9.17 corrugated_part_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.19.9.18 corrugated_part_design_definition to non_manifold_surface_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.19.9.19 corrugated_part_design_definition to corrugation (as shape_-aspect)

AIM element: PATH
Rules: 5.2.4.101
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect

{[shape_aspect.name = 'shape aspect']
[/CLASS_ID(shape_aspect, 'corrugation')/]}

5.1.19.10 CORRUGATION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.327
Reference path: {/ROOT_CLASS(shape_aspect, 'corrugation')/}

5.1.19.10.1 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.268
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('corrugation dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('depth', positive_length_measure)/

5.1.19.10.2 flat_width_1

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.268
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('corrugation dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('flat width 1', positive_length_measure)/

5.1.19.10.3 flat_width_2

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.268
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('corrugation dimensions')/->
representation
representation.items[i] ->
/VAL_REP_ITEM('flat width 2', positive_length_measure)/

5.1.19.10.4 radius_1

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.268
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('corrugation dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('radius 1', positive_length_measure)/

5.1.19.10.5 radius_2

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.245
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('corrugation dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('radius 2', positive_length_measure)/

5.1.19.10.6 slope_width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.268
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('corrugation dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('slope width', positive_length_measure)/

5.1.19.11 EXPLICIT_PROFILE_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.328
Reference path: {[/CLASS(shape_aspect, 'explicit profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.11.1 cross_section_geometry

AIM element: bounded_curve
Source: ISO 10303-42
Rules: 5.2.4.215, 5.2.4.190
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('explicit profile cross section shape')/ ->
representation =>
representation.items[i] ->
/GEO_REP_ITEM('cross section geometry', bounded_curve)/

5.1.19.11.2 designation

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.11.3 local_co_ordinate_system

AIM element: axis2_placement_2d
Source: ISO 10303-42
Rules: 5.2.4.215, 5.2.4.190
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('explicit profile cross section shape')/ ->
representation =>
representation.items[i] ->
/GEO_REP_ITEM('local coordinate system', axis2_placement_2d)/

5.1.19.11.4 expilicit_profile_cross_section to section_properties (as section_properties)

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.2)

5.1.19.12 FLANGED_PLATE_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.329
Reference path: {[/CLASS(shape_aspect, 'flanged profile cross section', 'proprietary profile cross section')/]
[/CLASS(shape_aspect, 'proprietary profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.12.1 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.191
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('flanged plate cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('depth', positive_legth_measure)/

5.1.19.12.2 designation

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.1)

5.1.19.12.3 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.191
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('flanged plate cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.19.12.4 radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.191
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('flanged plate cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('radius', positive_length_measure)/

5.1.19.12.5 thk

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.191
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('flanged plate cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('thk', positive_length_measure)/

5.1.19.12.6 flanged_plate_cross_section to section_properties (as section_-properties)

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.2)

5.1.19.13 FLANGED_PROFILE_CROSS_SECTION

#1: flanged_profile_cross_section is a t_bar_cross_section
#2: flanged_profile_cross_section is a w_shape_cross_section

AIM element: #1: /SUBTYPE(t_bar_cross_section)/ (See 5.1.19.41)
#2: /SUBTYPE(w_shape_cross_section)/ (See 5.1.19.42)

5.1.19.13.1 designation

AIM element: #1, #2: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.13.2 flanged_profile_cross_section to section_properties (as section_properties)

AIM element: #1, #2: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.14 FLAT_BAR_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.331
Reference path: {[/CLASS(shape_aspect, 'flat bar cross section', 'bar profile cross section')/]
[/CLASS(shape_aspect, 'bar profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.14.1 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.192
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('flat bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('depth', positive_length_measure)/

5.1.19.14.2 designation

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.14.3 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.192
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('flat bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.19.14.4 flat_bar_cross_section to section_properties (as section_-properties)

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.15 LIBRARY_PROFILE_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: {[/CLASS(shape_aspect, 'library profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.15.1 designation

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.1)

5.1.19.15.2 library_profile_cross_section to class_bsu (as library_identifier)

AIM element: PATH
Reference path: shape_aspect
classification_item = shape_aspect
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'definitional class membership'}
classification_assignment.assigned_classification ->
group =>
{group.name = 'library identifier'}
class =>
externally_defined_class

5.1.19.15.3 library_profile_cross_section to property_value (as property_-value_pairs)

AIM element: PATH
Reference path: shape_aspect
classification_item = shape_aspect
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'definitional class membership'}
classification_assignment.assigned_classification ->
group =>
{group.name = 'library identifier'}
class =>
externally_defined_class <=
externally_defined_item <-
externally_defined_item_relationship.related_item
{externally_defined_item_relationship.name=‘name scope’}
externally_defined_item_relationship
externally_defined_item_relationship.relating_item
externally_defined_item =>
externally_defined_general_property <=
general_property <-
general_property_association.base_definition
general_property_association
{ general_property_association.name ='definitional' }
general_property_association.derived_definition->
property_definition <-
property_definition_representation.definition
property_definition_representation
property_definition_representation.used_representation->
representation
{representation.name='property value'}

5.1.19.15.4 library_profile_cross_section to section_properties (as section_properties)

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.16 NON_CIRCULAR_HOLLOW_PROFILE_CROSS_SECTION

#1: non_circular_hollow_profile_cross_section is a square_tube_cross_section

AIM element: #1: /SUBTYPE(square_tube_cross_section)/ (See 5.1.19.34)

5.1.19.16.1 designation

AIM element: #1: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.16.2 non_circular_hollow_profile_cross_section to section_-properties (as section_properties)

AIM element: #1: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.17 PARAMETRIC_PROFILE_CROSS_SECTION

#1: angle_profile_cross_section
#2: bar_profile_cross_section
#3: channel_profile_cross_section
#4: circular_hollow_profile_cross_section
#5: flanged_profile_cross_section
#6: non_circular_hollow_profile_cross_section
#7: proprietary_profile_cross_section

AIM element: #1: /SUBTYPE(angle_profile_cross_section)/ (See 5.1.19.2)
#2: /SUBTYPE(bar_profile_cross_section)/ (See 5.1.19.3)
#3: /SUBTYPE(channel_profile_cross_section)/ (See 5.1.19.6)
#4: /SUBTYPE(circular_hollow_profile_cross_section)/ (See 5.1.19.7)
#5: /SUBTYPE(flanged_profile_cross_section)/ (See 5.1.19.13)
#6: /SUBTYPE(non_circular_hollow_profile_cross_section)/ (See 5.1.19.16)
#7: /SUBTYPE(proprietary_profile_cross_section)/ (See 5.1.19.31)

5.1.19.17.1 designation

AIM element: #1, #2, #3, #4, #5, #6, #7: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.1)

5.1.19.17.2 parametric_profile_cross_section to section_properties (as section_properties)

AIM element: #1, #2, #3, #4, #5, #6, #7: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.18 PLATE

AIM element: product_definition
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition, 'plate', 'structural part')/]
[/CLASS(product_definition, 'structural part', 'part')/]
[/CLASS(product_definition, 'part', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]}

5.1.19.18.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.19.18.2 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.19.18.3 plate to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
product_definition = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.19.18.4 plate to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition
identification_item = product_definition
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.18.5 plate to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')/}

5.1.19.19 PLATE_BOUNDARY_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition_relationship, 'plate boundary relationship', 'plate relationship')/]
[/CLASS(product_definition_relationship, 'plate relationship', 'structural part relationship')/]
[/CLASS(product_definition_relationship, 'structural part relationship', 'part relationship')/]
[/CLASS(product_definition_relationship, 'part relationship', 'item relationship')/]
[/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/ROOT_CLASS(product_definition_relationship, 'definable object')/]
[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]}

5.1.19.19.1 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.19.19.2 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path:
/VERSION_ID(product_definition_relationship)/

5.1.19.19.3 plate_boundary_relationship to external_instance_reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition

[/CLASS_ID(product_definition, 'plate')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'plate')/]}

5.1.19.19.4 plate_boundary_relationship to external_instance_reference (as external_item_2)

AIM element: PATH
Reference path: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/CLASS_ID(shape_aspect, 'seam')/]
[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'seam')/]}

5.1.19.19.5 plate_boundary_relationship to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition_relationship
identification_item = product_definition_relationship
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.19.6 plate_boundary_relationship to plate (as item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, 'plate')/}}

5.1.19.19.7 plate_boundary_relationship to seam (as item_2)

AIM element: PATH
Reference path: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect.of_shape
shape_aspect

{(/CLASS_ID(shape_aspect, 'seam')/)
(shape_aspect <-
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'plate boundary relationship')/})}

5.1.19.20 PLATE_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.120
Reference path: {[/CLASS(product_definition_shape, 'plate design definition', 'structural part design definition')/]
[/CLASS(product_definition_shape, 'structural part design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.19.20.1 plate_design_definition to bounded_curve (as border)

NOTE Shape_aspect.description should specify the index of the bounded_curve in the border list

AIM element: PATH
Rules: 5.2.4.76
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect =
represented_definition <-
/PDR_NAME('border representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', bounded_curve)/]}

5.1.19.20.2 plate_design_definition to panel_system_boundary (as border)

NOTE Shape_aspect.description should specify the index of the panel_system_boundary in the border list

AIM element: PATH
Rules: 5.2.4.76
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect <-

shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
shape_aspect_relationship.related_shape_aspect ->
shape_aspect
{/CLASS_ID(shape_aspect, 'boundary')/}
shape_aspect.of_shape ->
product_definition_shape
{/CLASS_ID(product_definition_shape, 'panel system design definition')/}]}

5.1.19.20.3 plate_design_definition to plate_boundary_relationship (as border)

NOTE Shape_aspect.description should specify the index of the plate_boundary_relationship in the border list

AIM element: PATH
Rules: 5.2.4.76
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'plate boundary relationship')/}]}

5.1.19.20.4 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.19.20.5 interior_point

AIM element: cartesian_point
Source: ISO 10303-42
Rules: 5.2.4.215, 5.2.4.249
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('plate design parameters')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('interior point', point)/]}

5.1.19.20.6 mass

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.249
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('plate design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('mass', mass_measure)/

5.1.19.20.7 material_offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.274
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('plate design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('material offset', ratio_measure)/

5.1.19.20.8 moulded_surface

#1: moulded surface is a surface
#2: moulded surface is an external_instance_reference
#3: moulded surface is a non_manifold_surface_shape

AIM element: #1: PATH
#2: PATH
#3: PATH

Source:
Rules: 5.2.4.78
Reference path: #1: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.name = 'moulded surface']
[/CLASS_ID(shape_aspect, 'moulded surface')/]
[shape_aspect = represented_definition <-
/PDR_NAME('moulded surface representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', surface)/}

#2: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.name = 'moulded surface']
[/CLASS_ID(shape_aspect, 'moulded surface')/]
[/EXT_INST_REF(shape_aspect, 'ship moulded form schema', 'moulded form')/]}

#3: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.name = 'moulded surface']
[/CLASS_ID(shape_aspect, 'moulded surface')/]
[shape_aspect = represented_definition <-
/PDR_NAME('moulded surface representation')/ ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation]}

5.1.19.20.9 thickness

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.274
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/PDR_NAME('plate design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('thickness', positive_length_measure)/

5.1.19.20.10 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.19.20.11 plate_design_definition to plate (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'plate')/}

5.1.19.20.12 plate_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.20.13 plate_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.19.20.14 plate_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.19.20.15 plate_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.19.20.16 plate_design_definition to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=

property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.19.20.17 plate_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.19.20.18 plate_design_definition to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.19.21 PLATE_FUNCTIONAL_DEFINITION

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.160
Reference path: {[/CLASS(property_definition, 'plate functional definition', 'structural part functional definition')/]
[/CLASS(property_definition, 'structural part functional definition', 'functional definition')/]
[/CLASS(property_definition, 'functional definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.19.21.1 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.19.21.2 the_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.275, 5.2.4.357
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('plate function parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'function'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description = 'standard plate')
(descriptive_representation_item.description = 'doubling plate')
(descriptive_representation_item.description = 'shedding plate')
(descriptive_representation_item.description = 'clip')
(descriptive_representation_item.description = 'watertight clip')
(descriptive_representation_item.description = 'lug')
(descriptive_representation_item.description = 'bracket')
(descriptive_representation_item.description = 'face plate')
(descriptive_representation_item.description = 'bilge plate')
(descriptive_representation_item.description = 'user defined')}

5.1.19.21.3 user_def_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.250
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('plate function parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'user def function'}
representation_item =>
descriptive_representation_item
descriptive_representation_item.description

5.1.19.21.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.19.21.5 plate_functional_definition to plate (as defined_for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition =
characterized_product_definition =
product_definition
{/CLASS_ID(product_definition, 'plate')/}

5.1.19.21.6 plate_functional_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: property_definition
identification_item = property_definition
identification_item <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.22 PLATE_RELATIONSHIP

#1: plate_relationship is a plate_boundary_relationship

AIM element: #1: /SUBTYPE(plate_boundary_relationship)/ (See 5.1.19.19)

5.1.19.22.1 description

AIM element: #1: /SUBTYPE(plate_boundary_relationship)/ (See 5.1.19.19)

5.1.19.22.2 version_id

AIM element: #1: /SUBTYPE(plate_boundary_relationship)/ --(See 5.1.19.19.2)

5.1.19.22.3 plate_relationship to external_instance_reference (as external_item_1)

AIM element: #1: /SUBTYPE(plate_boundary_relationship)/ (See 5.1.19.19)

5.1.19.22.4 plate_relationship to external_instance_reference (as external_item_2)

AIM element: #1: /SUBTYPE(plate_boundary_relationship)/ (See 5.1.19.19.4)

5.1.19.22.5 plate_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(plate_boundary_relationship)/ --(See 5.1.19.19.5)

5.1.19.22.6 plate_relationship to plate (as item_1)

AIM element: #1: /SUBTYPE(plate_boundary_relationship)/ (See 5.1.19.19)

5.1.19.22.7 plate_relationship to item (as item_2)

NOTE Attribute inherited from supertype item_relationship (see 5.1.9.4).

AIM element: #1: /SUBTYPE(plate_boundary_relationship)/ (See 5.1.19.19.7)

5.1.19.23 PROFILE

AIM element: product_definition
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition, 'profile', 'structural part')/]
[/CLASS(product_definition, 'structural part', 'part')/]
[/CLASS(product_definition, 'part', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]}

5.1.19.23.1 description

AIM element: product_definition.description

5.1.19.23.2 name

AIM element: product_definition.name
Reference path: /NAME_ASSGN(product_definition)/

5.1.19.23.3 profile to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
product_definition = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=

external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.19.23.4 profile to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.144
Reference path: product_definition
identification_item = product_definition
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.23.5 profile to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')/}

5.1.19.24 PROFILE_BOUNDARY_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition_relationship, 'profile boundary relationship', 'profile relationship')/]
[/CLASS(product_definition_relationship, 'profile relationship', 'structural part relationship')/]
[/CLASS(product_definition_relationship, 'structural part relationship', 'part relationship')/]
[/CLASS(product_definition_relationship, 'part relationship', 'item relationship')/]
[/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/ROOT_CLASS(product_definition_relationship, 'definable object')/]

[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]}

5.1.19.24.1 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.19.24.2 boundary_index

AIM element: #1: shape_aspect
Source: ISO 10303-41
Reference path: #1: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('panel system')/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect
{/CLASS_ID(shape_aspect, 'border')/}

5.1.19.24.3 displacement

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.235
Reference path: product_definition_relationship
product_definition_relationship.relating_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect <-
{/CLASS_ID(shape_aspect, 'border)/'}
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile boundary relationship')/}
represented_definition = shape_aspect_relationship <-
/PDR_NAME('profile boundary relationship design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('displacement', length_measure)/

5.1.19.24.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.19.24.5 profile_boundary_relationship to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.91
Reference path: product_definition_relationship
identification_item = product_definition_relationship
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.24.6 profile_boundary_relationship to external_instance_reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'profile')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'profile')/]}

5.1.19.24.7 profile_boundary_relationship to external_instance_reference (as external_item_2)

AIM element: #1: external_item_2 is an external panel system
#2: external_item_2 is an external seam
#3: external_item_2 is an external interiour cutout
Source: #1: product_definition_relationship.related_product_definition
#2: PATH
#3: PATH
Reference path: #1: {product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'panel system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'panel system')/]}

#2: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{[/CLASS_ID(shape_aspect, 'seam')/]
[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'seam')/]}

#3: product_definition_relationship

product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{[/CLASS_ID(shape_aspect, 'interior cutout')/]
[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'interior cutout')/]}

5.1.19.24.8 profile_boundary_relationship to profile (as item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, 'profile')/}}

5.1.19.24.9 profile_boundary_relationship to item (as item_2)

#1: item_2 is a panel system
#2: item_2 is another profile
#3: item_2 is a plate
#4: item_2 is a seam
#5: item_2 is an interior cutout

AIM element: #1: product_definition_relationship. related_product_definition
#2: product_definition_relationship. related_product_definition
#3: product_definition_relationship. related_product_definition
#4: PATH
#5: PATH
Source: #1:ISO 10303-41
#2:ISO 10303-41
#3:ISO 10303-41
Reference path: #1: {product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('panel system')/ <-
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{(/CLASS_ID(shape_aspect, 'border')/)
(/CLASS_ID(shape_aspect, 'moulded surface')/)}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile boundary relationship')/}}


```
#2: {product_definition_relationship
{/SAME_GROUP(product_definition_relationship.  relating_product_definition,
product_definition_relationship.related_product_definition, 'panel system')/}
product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('profile')/ <-
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'trace line')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile boundary relationship')/}}

#3: product_definition_relationship
{[/SAME_GROUP(product_definition_relationship.relating_product_definition,
product_definition_relationship.related_product_definition, 'panel system')/]
 [product_definition_relationship.  related_product_definition ->
/PD_HAS_FUNC('bracket')/]}
product_definition_relationship.  related_product_definition ->
/PDCD_CLASS_ID('plate')/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'moulded surface')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile boundary relationship')/}

#4: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'seam')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile boundary relationship')/}

#5: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
```


property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'interior cutout')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile boundary relationship')/}

5.1.19.25 PROFILE_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: /SUBTYPE(explicit_profile_cross_section)/ (See 5.1.19.11)
/SUBTYPE(library_profile_cross_section)/ (See 5.1.19.15)
/SUBTYPE(parametric_profile_cross_section)/ (See 5.1.19.17)

5.1.19.25.1 designation

AIM element: shape_aspect.name
Source: ISO 10303-41

5.1.19.25.2 profile_cross_section to section_properties (as section_-properties)

AIM element: PATH
Rules: 5.2.4.147
Reference path: shape_aspect = represented_definition <-
property_definition_representation.definition
property_definition_representation
{property_definition_representation.name = 'section properties'}
property_definition_representation.used_representation ->
representation
{/CLASS_ID(representation, 'section properties')/}

5.1.19.26 PROFILE_CURVE_TRACE_LINE

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.334
Reference path: {/ROOT_CLASS(shape_aspect, 'profile curve trace line')/}

5.1.19.26.1 curve

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.199
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('profile curve trace line design parameters')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('curve', curve)/

5.1.19.26.2 displacement

AIM element: value_representation_item.value_component
Source: ISO 10303-42
Rules: 5.2.4.215, 5.2.4.236
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('profile curve trace line design parameters')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('displacement', vector)/

5.1.19.27 PROFILE_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.122
Reference path:
{[/CLASS(product_definition_shape, 'profile design definition', 'structural part design definition')/]
[/CLASS(product_definition_shape, 'structural part design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.19.27.1 profile_design_definition to cartesian_point (as border)

NOTE Shape_aspect.description should specify the index of the cartesian_point in the border list

AIM element: PATH
Rules: 5.2.4.148
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect =
represented_definition <-

/SDR_NAME('border representation')/ ->
representation
representation.items[i] ->
(/GEO_REP_ITEM('.UNUSED.', cartesian_point)/)]}

5.1.19.27.2 profile_design_definition to line (as border)

NOTE Shape_aspect.description should specify the index of the line in the border list

AIM element: PATH
Rules: 5.2.4.148
Reference_path:product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect =
represented_definition <-
/SDR_NAME('border representation')/ ->
representation
representation.items[i] ->
(/GEO_REP_ITEM('.UNUSED.', line)/)]}

5.1.19.27.3 profile_design_definition to point_on_curve (as border)

NOTE Shape_aspect.description should specify the index of the point_on_curve in the border list

AIM element: PATH
Rules: 5.2.4.148
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect =
represented_definition <-
/SDR_NAME('border representation')/ ->
representation
representation.items[i] ->
(/GEO_REP_ITEM('.UNUSED.', point_on_curve)/)]}

5.1.19.27.4 profile_design_definition to profile_boundary_relationship (as border)

NOTE Shape_aspect.description should specify the index of the profile_boundary_relationship in the border list

AIM element: PATH
Rules: 5.2.4.148
Reference path: product_definition_shape <-

shape_aspect.of_shape
shape_aspect
{[shape_aspect.description = 'i']
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile boundary relationship')/}]}

5.1.19.27.5 cross_section_placement

AIM element: axis2_placement_3d
Source: ISO 10303-42
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/SDR_NAME('profile design parameters')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('cross section placement', axis2_placement_3d)/

5.1.19.27.6 description

AIM element: product_definition_shape.description
Source: ISO 10303-41

5.1.19.27.7 mass

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.253
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/SDR_NAME('profile design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('mass', mass_measure)/

5.1.19.27.8 mirrored

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.277
Reference path: product_definition_shape <=
property_definition = represented_definition <-
/SDR_NAME('profile design parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('mirrored')/
{(descriptive_representation_item.description = 'TRUE')

(descriptive_representation_item.description = 'FALSE')}

5.1.19.27.9 trace_line

#1: trace_line is a profile_curve_trace_line
#2: trace_line is a profile_trace_line_relationship
#3: trace_line is an external_instance_reference

AIM element: #1: PATH
#2: PATH
#3: PATH
Rules: 5.2.4.148
Reference path: #1: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.name = 'trace line']
[/ROOT_CLASS(shape_aspect, 'trace line')/]
[/CLASS_ID(shape_aspect, 'profile curve trace line')/]}

#2: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.name = 'trace line']
[/ROOT_CLASS(shape_aspect, 'trace line')/]
[shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile trace line relationship')/}]}

#3: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.name = 'trace line']
[/ROOT_CLASS(shape_aspect, 'trace line')/]
[/EXT_INST_REF(product_definition_shape, 'ship moulded form schema', 'moulded form')/]}

5.1.19.27.10 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.19.27.11 profile_design_definition to profile_cross_section (as cross_-section)

AIM element: PATH
Rules: 5.2.4.121
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{(shape_aspect.name = 'cross section')
(/CLASS_ID(shape_aspect, 'profile cross section')/)}

5.1.19.27.12 profile_design_definition to profile (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'profile')/}

5.1.19.27.13 profile_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.138
Reference path: product_definition_shape
identification_item = product_definition_shape
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.27.14 profile_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.19.27.15 profile_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.19.27.16 profile_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.19.27.17 profile_design_definition to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.19.27.18 profile_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.19.27.19 profile_design_definition to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('representation')/

property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.19.27.20 profile_design_definition to twist_location (as twist)

AIM element: PATH
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[shape_aspect.name = 'twist']
[/CLASS_ID(shape_aspect, 'twist location')/]}

5.1.19.28 PROFILE_FUNCTIONAL_DEFINITION

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.162
Reference path: {[/CLASS(property_definition, 'profile functional definition', 'structural part functional definition')/]
[/CLASS(property_definition, 'structural part functional definition', 'functional definition')/]
[/CLASS(property_definition, 'functional definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.19.28.1 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.19.28.2 the_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.357, 5.2.4.200
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('profile function parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'function'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description = 'stiffener')
(descriptive_representation_item.description = 'longitudinal stiffener')

(descriptive_representation_item.description = 'transversal stiffener')
(descriptive_representation_item.description = 'carling')
(descriptive_representation_item.description = 'beam')
(descriptive_representation_item.description = 'deck beam')
(descriptive_representation_item.description = 'pillar')
(descriptive_representation_item.description = 'hold pillar')
(descriptive_representation_item.description = 'flange')
(descriptive_representation_item.description = 'web')
(descriptive_representation_item.description = 'frame')
(descriptive_representation_item.description = 'user defined')}

5.1.19.28.3 user_def_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.254
Reference path: property_definition = represented_definition <-
/PDR_NAME('profile function parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'user def function'}
representation_item =>
descriptive_representation_item descriptive_representation_item.description

5.1.19.28.4 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.19.28.5 profile_functional_definition to profile (as defined_for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
{/CLASS_ID(product_definition, 'profile')/}

5.1.19.28.6 profile_functional_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.170
Reference path: property_definition
identification_item = property_definition
identification_item<-
applied_identification_assignment.items[i]

applied_identification_assignment

5.1.19.29 PROFILE_RELATIONSHIP

#1: profile_relationship is a profile_trace_line_relationship
#2: profile_relationship is a profile_boundary_relationship

AIM element: #1: /SUBTYPE(profile_trace_line_relationship)/ (See 5.1.19.30)
#2: /SUBTYPE(profile_boundary_relationship)/ (See 5.1.19.24)

5.1.19.29.1 description

AIM element: #1: /SUBTYPE(profile_trace_line_relationship)/ (See 5.1.19.30.3)
#2: /SUBTYPE(profile_boundary_relationship)/ (See 5.1.19.24.1)

5.1.19.29.2 version_id

AIM element: #1: /SUBTYPE(profile_trace_line_relationship)/ (See 5.1.19.30.5)
#2: /SUBTYPE(profile_boundary_relationship)/ (See 5.1.19.24.4)

5.1.19.29.3 profile_relationship to external_instance_reference (as external_item_1)

AIM element: #1: /SUBTYPE(profile_trace_line_relationship)/ (See 5.1.19.30)
#2: /SUBTYPE(profile_boundary_relationship)/ (See 5.1.19.24)

5.1.19.29.4 profile_relationship to external_instance_reference (as external_item_2)

AIM element: #1: /SUBTYPE(profile_trace_line_relationship)/ (See 5.1.19.30.7)
#2: /SUBTYPE(profile_boundary_relationship)/ (See 5.1.19.24.7)

5.1.19.29.5 profile_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(profile_trace_line_relationship)/ (See 5.1.19.30.8)
#2: /SUBTYPE(profile_boundary_relationship)/ (See 5.1.19.24.5)

5.1.19.29.6 profile_relationship to profile (as item_1)

AIM element: #1: /SUBTYPE(profile_trace_line_relationship)/ (See 5.1.19.30)
#2: /SUBTYPE(profile_boundary_relationship)/ (See 5.1.19.24)

5.1.19.29.7 profile_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(profile_trace_line_relationship)/ (See 5.1.19.30)
#2: /SUBTYPE(profile_boundary_relationship)/ (See 5.1.19.24)

5.1.19.30 PROFILE_TRACE_LINE_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition_relationship, 'profile trace line relationship', 'profile relationship')/]
[/CLASS(product_definition_relationship, 'profile relationship', 'structural part relationship')/]
[/CLASS(product_definition_relationship, 'structural part relationship', 'part relationship')/]
[/CLASS(product_definition_relationship, 'part relationship', 'item relationship')/][/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/ROOT_CLASS(product_definition_relationship, 'definable object')/]
[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]}

5.1.19.30.1 auxiliary_line

AIM element: #1: PATH
Rules: 5.2.4.215, 5.2.4.237
Reference path: #1: product_definition_relationship
product_definition_relationship.relating_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'trace line')/}
shape_aspect_relationship = represented_definition <-
/PDR_NAME('profile trace line relationship design parameters')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('auxiliary line', line)/

5.1.19.30.2 boundary_index

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('panel system')/ <-
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{/CLASS_ID(shape_aspect, 'border')/}

5.1.19.30.3 displacement

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.237
Reference path: product_definition_relationship
product_definition_relationship.relating_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect <-
{/CLASS_ID(shape_aspect, 'trace line')/}
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship = represented_definition <-
/PDR_NAME('profile trace line relationship design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('displacement', length_measure)/

5.1.19.30.4 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.19.30.5 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.19.30.6 profile_trace_line_relationship to external_instance_reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'profile')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'profile')/]}

5.1.19.30.7 profile_trace_line_relationship to external_instance_reference (as external_item_2)

AIM element: #1: product_definition_relationship.related_product_definition
#2: PATH
#3: PATH
Source: #1: ISO 10303-41
Rules:
Reference path: #1: {product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'panel system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'panel system')/]}

#2: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/CLASS_ID(shape_aspect, 'seam')/]
[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'seam')/]}

#3: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{[/CLASS_ID(shape_aspect, 'interior cutout')/]
[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'interior cutout')/]}

5.1.19.30.8 profile_trace_line_relationship to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.91
Reference path: product_definition_relationship
identification_item = product_definition_relationship
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.30.9 profile_trace_line_relationship to profile (as item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, 'profile')/}}

5.1.19.30.10 profile_trace_line_relationship to item (as item_2)

#1: profile_trace_line_relationship to panel_system (as item_2)
#2: profile_trace_line_relationship to seam (as item_2)
#3: profile_trace_line_relationship to interior_cutout (as item_2)

AIM element: #1: product_definition_relationship.related_product_definition
#2: PATH
#3: PATH
Source: #1:ISO 10303-41
Reference path: #1: {product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('panel system')/ ->
property_definition.definition
property_definition =>
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'border')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile trace line relationship')/}}

#2: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{[/CLASS_ID(shape_aspect, 'seam')/]
[shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile trace line relationship')/}]}

#3: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'interior cutout')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'profile trace line relationship')/}

5.1.19.31 PROPRIETARY_PROFILE_CROSS_SECTION

#1: proprietary_profile_cross_section is a flanged_plate_cross_section

AIM element: #1: /SUPERTYPE(flanged_plate_cross_section)/ (See 5.1.19.12)

5.1.19.31.1 designation

AIM element: #1: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.1)

5.1.19.31.2 proprietary_profile_cross_section to section_properties (as section_properties)

AIM element: #1: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.32 ROUND_BAR_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.335
Reference path: {[/CLASS(shape_aspect, 'round bar cross section', 'bar profile cross section')/]
[/CLASS(shape_aspect, 'bar profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.32.1 designation

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.32.2 diameter

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.203
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('round bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('diameter', positive_length_measure)/

5.1.19.32.3 round_bar_cross_section to section_properties (as section_-properties)

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.33 SECTION_PROPERTIES

AIM element: representation
Source: ISO 10303-43
Reference path: {/ROOT_CLASS(representation, 'section properties')/}

5.1.19.33.1 area

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.278
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('area', area_measure)/

5.1.19.33.2 moi_u

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.278
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('moi u', inertia_moment_measure)/

5.1.19.33.3 moi_v

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.278
Reference path: representation
representation.items[i] ->

/VAL_REP_ITEM('moi v', inertia_moment_measure)/

5.1.19.33.4 moi_uv

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.278
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('moi uv', inertia_moment_measure)/

5.1.19.33.5 na_u

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.278
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('na u', positive_length_measure)/

5.1.19.33.6 na_v

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.278
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('na v', positive_length_measure)/

5.1.19.33.7 nominal_mass_per_len

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.278
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('nominal mass per length', weight_per_length_measure)/

5.1.19.33.8 tr

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.278
Reference path: representation
representation.items[i] ->
/VAL_REP_ITEM('tr', inertia_moment_measure)/

5.1.19.34 SQUARE_TUBE_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.336
Reference path: {[/CLASS(shape_aspect, 'square tube cross section', 'non circular profile cross section')/]
[/CLASS(shape_aspect, 'non circular profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.34.1 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.207
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('square tube cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('depth', positive_length_measure)/

5.1.19.34.2 designation

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.1)

5.1.19.34.3 wall_thk

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.207
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('square tube cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('wall thk', positive_length_measure)/

5.1.19.34.4 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.207
Reference path: shape_aspect =
represented_definition <-
/PDR_NAME('square tube cross section dimensions')/ ->

representation
representation.items[i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.19.34.5 square_tube_cross_section to section_properties (as section_-properties)

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.35 STRUCTURAL_PART

#1: structural_part is a plate
#2: structural_part is a profile
#3: structural_part is a corrugated_part

AIM element: #1: /SUBTYPE(plate)/ (See 5.1.19.18)
#2: /SUBTYPE(profile)/ (See 5.1.19.23)
#3: /SUBTYPE(corrugated_part)/ (See 5.1.19.8)

5.1.19.35.1 description

AIM element: #1: /SUBTYPE(plate)/ (See 5.1.19.18)
#2: /SUBTYPE(profile)/ (See 5.1.19.23)
#3: /SUBTYPE(corrugated_part)/ (See 5.1.19.8)

5.1.19.35.2 name

AIM element: #1: /SUBTYPE(plate)/ (See 5.1.19.18)
#2: /SUBTYPE(profile)/ (See 5.1.19.23)
#3: /SUBTYPE(corrugated_part)/ (See 5.1.19.8)

5.1.19.35.3 structural_part to external_reference (as documentation)

AIM element: #1: /SUBTYPE(plate)/ (See 5.1.19.18)
#2: /SUBTYPE(profile)/ (See 5.1.19.23)
#3: /SUBTYPE(corrugated_part)/ (See 5.1.19.8)

5.1.19.35.4 structural_part to global_id (as id)

AIM element: #1: /SUBTYPE(plate)/ (See 5.1.19.18)
#2: /SUBTYPE(profile)/ (See 5.1.19.23)
#3: /SUBTYPE(corrugated_part)/ (See 5.1.19.8)

5.1.19.35.5 structural_part to ship (as ship_context)

AIM element: #1: /SUBTYPE(plate)/ (See 5.1.19.18)
#2: /SUBTYPE(profile)/ (See 5.1.19.23)
#3: /SUBTYPE(corrugated_part)/ (See 5.1.19.8)

5.1.19.36 STRUCTURAL_PART_DESIGN_DEFINITION

#1: structural_part_design_definition is a plate_design_definition
#2: structural_part_design_definition is a profile_design_definition
#3: structural_part_design_definition is a corrugated_part_design_definition

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.1 description

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20.4)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27.6)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9.4)

5.1.19.36.2 mass

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.3 version_id

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.4 structural_part_design_definition to structural_part (as defined_for)

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.5 structural_part_design_definition to global_id (as id)

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.6 structural_part_design_definition to derived_unit (as local_-units)

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.7 structural_part_design_definition to named_unit (as local_-units)

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.8 structural_part_design_definition to advanced_brep_shape (as representations)

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.9 structural_part_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.10 structural_part_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.36.11 structural_part_design_definition to non_manifold_surface_-shape (as representations)

AIM element: #1: /SUBTYPE(plate_design_definition)/ (See 5.1.19.20)
#2: /SUBTYPE(profile_design_definition)/ (See 5.1.19.27)
#3: /SUBTYPE(corrugated_part_design_definition)/ (See 5.1.19.9)

5.1.19.37 STRUCTURAL_PART_FUNCTIONAL_DEFINITION

#1: structural_part_functional_definition is a plate_functional_definition
#2: structural_part_functional_definition is a profile_functional_definition

AIM element: #1: /SUBTYPE(plate_functional_definition)/ (See 5.1.19.21)
#2: /SUBTYPE(profile_functional_definition)/ (See 5.1.19.28)

5.1.19.37.1 description

AIM element: #1: /SUBTYPE(plate_functional_definition)/ (See 5.1.19.21.1)
#2: /SUBTYPE(profile_functional_definition)/ (See 5.1.19.28.1)

5.1.19.37.2 the_function

AIM element: #1: /SUBTYPE(plate_functional_definition)/ (See 5.1.19.21)
#2: /SUBTYPE(profile_functional_definition)/ (See 5.1.19.28)

5.1.19.37.3 user_def_function

AIM element: #1: /SUBTYPE(plate_functional_definition)/ (See 5.1.19.21)
#2: /SUBTYPE(profile_functional_definition)/ (See 5.1.19.28)

5.1.19.37.4 version_id

AIM element: #1: /SUBTYPE(plate_functional_definition)/ (See 5.1.19.21)
#2: /SUBTYPE(profile_functional_definition)/ (See 5.1.19.28)

5.1.19.37.5 structural_part_functional_definition to structural_part (as defined_for)

AIM element: #1: /SUBTYPE(plate_functional_definition)/ (See 5.1.19.21)
#2: /SUBTYPE(profile_functional_definition)/ (See 5.1.19.28)

5.1.19.37.6 structural_part_functional_definition to global_id (as id)

AIM element: #1: /SUBTYPE(plate_functional_definition)/ (See 5.1.19.21)
#2: /SUBTYPE(profile_functional_definition)/ (See 5.1.19.28)

5.1.19.38 STRUCTURAL_PART_RELATIONSHIP

#1: structural_part_symmetry_relationship
#2: plate_relationship
#3: profile_relationship

AIM element: #1: /SUBTYPE(structural_part_symmetry_relationship)/ (See 5.1.19.39)
#2: /SUBTYPE(plate_relationship)/ (See 5.1.19.22)
#3: /SUBTYPE(profile_relationship)/ (See 5.1.19.29)

5.1.19.38.1 description

AIM element: #1: /SUBTYPE(structural_part_symmetry_relationship)/ (See 5.1.19.39.1)
#2: /SUBTYPE(plate_relationship)/ (See 5.1.19.22.1)
#3: /SUBTYPE(profile_relationship)/ (See 5.1.19.29.1)

5.1.19.38.2 version_id

AIM element: #1: /SUBTYPE(structural_part_symmetry_relationship)/ (See 5.1.19.39)
#2: /SUBTYPE(plate_relationship)/ (See 5.1.19.22)
#3: /SUBTYPE(profile_relationship)/ (See 5.1.19.29)

5.1.19.38.3 structural_part_relationship to external_instance_reference (as external_item_1)

AIM element: #1: /SUBTYPE(structural_part_symmetry_relationship)/ (See 5.1.19.39.4)
#2: /SUBTYPE(plate_relationship)/ (See 5.1.19.22.3)
#3: /SUBTYPE(profile_relationship)/ (See 5.1.19.29.3)

5.1.19.38.4 structural_part_relationship to external_instance_reference (as external_item_2)

AIM element: #1: /SUBTYPE(structural_part_symmetry_relationship)/ (See 5.1.19.39)
#2: /SUBTYPE(plate_relationship)/ (See 5.1.19.22)
#3: /SUBTYPE(profile_relationship)/ (See 5.1.19.29)

5.1.19.38.5 structural_part_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(structural_part_symmetry_relationship)/ (See 5.1.19.39)
#2: /SUBTYPE(plate_relationship)/ (See 5.1.19.22)
#3: /SUBTYPE(profile_relationship)/ (See 5.1.19.29)

5.1.19.38.6 structural_part_relationship to structural_part (as item_1)

AIM element: #1: /SUBTYPE(structural_part_symmetry_relationship)/ (See 5.1.19.39)
#2: /SUBTYPE(plate_relationship)/ (See 5.1.19.22)
#3: /SUBTYPE(profile_relationship)/ (See 5.1.19.29)

5.1.19.38.7 structural_part_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(structural_part_symmetry_relationship)/ (See 5.1.19.39)
#2: /SUBTYPE(plate_relationship)/ (See 5.1.19.22)
#3: /SUBTYPE(profile_relationship)/ (See 5.1.19.29)

5.1.19.39 STRUCTURAL_PART_SYMMETRY_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Rules: 5.2.4.354
Reference path:
{[/CLASS(product_definition_relationship, 'structural part symmetry relationship', 'structural part relationship')/]
[/CLASS(product_definition_relationship, 'structural part relationship', 'part relationship')/]
[/CLASS(product_definition_relationship, 'part relationship', 'item relationship')/]
[/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/ROOT_CLASS(product_definition_relationship, 'definable object')/]
[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]}

5.1.19.39.1 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.19.39.2 mirroring_plane

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.210
Reference path: product_definition_relationship
{product_definition_relationship.name = '.UNUSED.'}
product_definition_relationship = characterized_product_definition
characterized_product_definition = characterized_definition <-
property_definition.definition
property_definition
{property_definition.name = '.UNUSED.'}
property_definition = represented_definition <-
/PDR_NAME('structural part symmetry relationship design parameters')/ ->
representation
representation.items[i] ->

/GEO_REP_ITEM('mirroring plane', plane)/

5.1.19.39.3 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.19.39.4 structural_part_ symmetry_relationship to external_instance_-reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
([/CLASS_ID(product_definition, 'plate')/]
[EXT_INST_REF(product_definition, 'ship structures schema', 'plate')/])
([/CLASS_ID(product_definition, 'profile')/]
[EXT_INST_REF(product_definition, 'ship structures schema', 'profile')/])
([/CLASS_ID(product_definition, 'corrugated part')/]
[(product_definition, 'ship structures schema', 'corrugated part')/])}

5.1.19.39.5 structural_part_symmetry_relationship to external_instance_-reference (as external_item_2)

AIM element: product_definition_relationship.related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship
product_definition_relationship.related_product_definition ->
product_definition
([/CLASS_ID(product_definition, 'plate')/]
[EXT_INST_REF(product_definition, 'ship structures schema', 'plate')/])
([/CLASS_ID(product_definition, 'profile')/]
[EXT_INST_REF(product_definition, 'ship structures schema', 'profile')/])
([/CLASS_ID(product_definition, 'corrugated part')/]
[EXT_INST_REF(product_definition, 'ship structures schema', 'corrugated part')/])}

5.1.19.39.6 structural_part_symmetry_relationship to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.91
Reference path: product_definition_relationship
identification_item = product_definition_relationship
identification_item<-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.19.39.7 structural_part_symmetry_relationship to structural_part (as item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
(/CLASS_ID(product_definition, 'plate')/)
(/CLASS_ID(product_definition, 'profile')/)
(/CLASS_ID(product_definition, 'corrugated part')/)}

5.1.19.39.8 structural_part_symmetry_relationship to structural_part (as item_2)

AIM element: product_definition_relationship. related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.related_product_definition ->
product_definition
(/CLASS_ID(product_definition, 'plate')/)
(/CLASS_ID(product_definition, 'profile')/)
(/CLASS_ID(product_definition, 'corrugated part')/)}

5.1.19.40 TWIST_LOCATION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.339
Reference path: {/ROOT_CLASS(shape_aspect, 'twist location')/}

5.1.19.40.1 location

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.214
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('twist location design parameters')/ ->
representation =>
representation.items[i] ->
/GEO_REP_ITEM('location', point_on _curve)/

5.1.19.40.2 orientation

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.214
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('twist location design parameters')/ ->
representation =>
representation.items[i] ->

/GEO_REP_ITEM('direction', direction)/

5.1.19.41 T_BAR_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.338
Reference path: {[/CLASS(shape_aspect, 't bar cross section', 'angle profile cross section')/]
[/CLASS(shape_aspect, 'angle profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.41.1 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.212
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('t bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('depth', positive_length_measure)/

5.1.19.41.2 designation

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.1)

5.1.19.41.3 flange_thk

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.212
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('t bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('flange thk', positive_length_measure)/

5.1.19.41.4 k

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.212
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('t bar cross section dimensions')/ ->
representation
representation.items[i] ->

/VAL_REP_ITEM('k', positive_length_measure)/

5.1.19.41.5 radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.212
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('t bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('radius', positive_length_measure)/

5.1.19.41.6 web_thk

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.212
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('t bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('web thk', positive_length_measure)/

5.1.19.41.7 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.212
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('t bar cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.19.41.8 t_bar_cross_section to section_properties (as section_-properties)

AIM element: /SUPERTYPE(profile_cross_section)/ -- (See 5.1.19.25.2)

5.1.19.42 W_SHAPE_CROSS_SECTION

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.340
Reference path: {[/CLASS(shape_aspect, 'w shape cross section', 'flanged profile cross section')/]
[/CLASS(shape_aspect, 'flanged profile cross section', 'parametric profile cross section')/]
[/CLASS(shape_aspect, 'parametric profile cross section', 'profile cross section')/]
[/ROOT_CLASS(shape_aspect, 'profile cross section')/]}

5.1.19.42.1 depth

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.280
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('w shape cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('depth', positive_length_measure)/

5.1.19.42.2 designation

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.1)

5.1.19.42.3 flange_thk

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.280
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('w shape cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('flange thk', positive_length_measure)/

5.1.19.42.4 k

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.280
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('w shape cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('k', positive_length_measure)/

5.1.19.42.5 radius

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.280
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('w shape cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('radius', positive_length_measure)/

5.1.19.42.6 web_thk

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.280
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('w shape cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('web thk', positive_length_measure)/

5.1.19.42.7 width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.280
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('w shape cross section dimensions')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('width', positive_length_measure)/

5.1.19.42.8 w_shape_cross_section to section_properties (as section_-properties)

AIM element: /SUPERTYPE(profile_cross_section)/ (See 5.1.19.25.2)

5.1.20 structural_systems UoF

5.1.20.1 BUILT_PROFILE

AIM element: [product_definition]
[group]
Source: ISO 10303-41
ISO 10303-41
Reference path: {[/LINK_TO_GROUP(product_definition)/]
[/CLASS(product_definition, 'built profile', ‘profile’)/]

[/CLASS(product_definition, 'profile', 'structural part')/]
[/CLASS(product_definition, 'structural part', 'part')/]
[/CLASS(product_definition, 'part', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]
[/CLASS(product_definition, 'built profile', 'structural system')/]
[/CLASS(product_definition, 'structural system', 'system')/]
[/CLASS(product_definition, 'system', 'item structure')/]
[/CLASS(product_definition, 'item structure', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]
[/CLASS(product_definition, 'item structure', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.20.1.1 the_class

AIM element: group.name
Source: ISO 10303-41
Reference path: product_definition
classification_item = product_definition
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = 'class membership'}
classification_assignment.assigned_classification ->
group
group.name
{group.name = 'built profile class'}

5.1.20.1.2 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.20.1.3 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.20.1.4 built_profile to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
external_identification_item = product_definition
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(document, ‘document reference with address’)/})

5.1.20.1.5 built_profile to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product,'ship')/}

5.1.20.1.6 built_profile to external_instance_reference (as external_items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition, 'item structure')/
{([/CLASS_ID(product_definition, 'plate')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'plate')/])
([/CLASS_ID(product_definition, 'profile')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'profile')/])}

5.1.20.1.7 built_profile to external_instance_reference (as external_-relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{[/CLASS_ID(product_definition_relationship, 'structural part relationship')/]
[/EXT_INST_REF(product_definition_relationship, 'ship structures schema',
'structural part relationship')/]}

5.1.20.1.8 built_profile to item (as items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition, 'item structure')/
{(/CLASS_ID(product_definition, 'plate')/)
(/CLASS_ID(product_definition, 'profile')/)}

5.1.20.1.9 built_profile to item_relationship (as relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{/CLASS_ID(product_definition_relationship, 'structural part relationship')/}

5.1.20.1.10 built_profile to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.1.11 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.20.2 CORRUGATED_STRUCTURE

AIM element: [product_definition]
[group]
Source: ISO 10303-41
Reference path: {[/LINK_TO_GROUP(product_definition)/]
[/CLASS(product_definition, 'corrugated structure', 'structural system')/]
[/CLASS(product_definition, 'structural system', 'system')/]
[/CLASS(product_definition, 'system', 'item structure')/]
[/CLASS(product_definition, 'item structure', 'definable object')/]
[/CLASS(product_definition, 'item structure', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]
[/CLASS(product_definition, 'system', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]}

5.1.20.2.1 the_class

AIM element: group.name
Source: ISO 10303-41
Reference path: product_definition
classification_item = product_definition
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = ‘class membership‘}
classification_assignment.assigned_classification ->
group
group.name
{group.name = ‘corrugated structure class’}

5.1.20.2.2 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.20.2.3 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.20.2.4 corrugated_structure to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
external_identification_item = product_definition
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(document, ‘document reference with address’)/})

5.1.20.2.5 corrugated_structure to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product,'ship')/}

5.1.20.2.6 corrugated_structure to external_instance_reference (as external_items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition, 'item structure')/
{([/CLASS_ID(product_definition, 'plate')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'plate')/])
([/CLASS_ID(product_definition, 'corrugated part')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'corrugated part')/])}

5.1.20.2.7 corrugated_structure to external_instance_reference (as external_relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{[/CLASS_ID(product_definition_relationship, 'plate relationship')/]
[/EXT_INST_REF(product_definition_relationship, 'ship structures schema',
'structural part relationship')/]}

5.1.20.2.8 corrugated_structure to item (as items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition, 'item structure')/
{(/CLASS_ID(product_definition, 'plate')/)
(/CLASS_ID(product_definition, 'corrugated part')/)}

5.1.20.2.9 corrugated_structure to item_relationship (as relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{/CLASS_ID(product_definition_relationship, 'structural part relationship')/}

5.1.20.2.10 corrugated_structure to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.2.11 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.20.3 CORRUGATED_STRUCTURE_BOUNDARY_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition_relationship, 'corrugated structure boundary relationship', 'corrugated structure relationship')/]
[/CLASS(product_definition_relationship, 'corrugated structure relationship', 'structural system relationship')/]
[/CLASS(product_definition_relationship, 'structural system relationship', 'system relationship')/]
[/CLASS(product_definition_relationship, 'system relationship', 'item relationship')/]
[/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/ROOT_CLASS(product_definition_relationship, 'definable object')/]
[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]}

5.1.20.3.1 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.20.3.2 corrugated_structure_boundary_relationship to item (as item_-2)

#1: related item is a corrugated_structure
#2: related item is a panel_system
#3: related item is a seam

AIM element: #1: product_definition_relationship.related_product_definition
#2: product_definition_relationship.related_product_definition
#3: PATH
Source: #1: ISO 10303-41
#2: ISO 10303-41
Reference path: #1:{product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID<'corrugated structure'> <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{CLASS_ID<shape_aspect, 'moulded surface'>}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{CLASS_ID<shape_aspect_relationship,
'corrugated structure boundary relationship'>}}

#2: {[product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('panel system')/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'moulded surface')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'corrugated structure boundary
relationship')/}]}

#3: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{CLASS_ID<shape_aspect, 'seam'>}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{CLASS_ID<shape_aspect_relationship,
'corrugated structure boundary relationship'>}

5.1.20.3.3 displacement

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.217
Reference path: product_definition_relationship
product_definition_relationship.relating_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect <-
{/CLASS_ID(shape_aspect, 'border')/}
shape_aspect_relationship =
{/CLASS_ID(shape_aspect_relationship, 'panel system boundary relationship')/}
represented_definition <-
/PDR_NAME('corrugated structure boundary relationship design parameters')/> ->
representation
representation.items[i] ->

/VAL_REP_ITEM('displacement', length_measure)/

5.1.20.3.4 corrugated_structure_boundary_relationship to corrugated_-structure (as item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, ‘corrugated structure’)/}

5.1.20.3.5 corrugated_structure_boundary_relationship to external_-instance_reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: product_definition_relationship
product_definition_relationship.relating_product_definition ->
product_definition
{[/CLASS_ID(product_definition, ‘corrugated structure’)/]
[/EXT_INST_REF(product_definition, 'ship structures schema', ‘corrugated structure’)/]}

5.1.20.3.6 corrugated_structure_boundary_relationship to external_-instance_reference (as external_item_2)

#1: related item is a corrugated_structure
#2: related item is a panel_system
#3: related item is a seam

AIM element: #1: product_definition_relationship.related_product_definition
#2: product_definition_relationship.related_product_definition
#3: PATH
Source: #1: ISO 10303-41
#2: ISO 10303-41
#3: ISO 10303-41
Reference path: #1: {product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'corrugated structure')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'corrugated structure')/]}

#2: {product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'panel system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'panel system')/]}

#3: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/CLASS_ID(shape_aspect, 'seam')/]
[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'seam')/]}

5.1.20.3.7 corrugated_structure_boundary_relationship to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_relationship
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.3.8 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.20.4 CORRUGATED_STRUCTURE_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.100
Reference path: {[/CLASS(product_definition_shape, 'corrugated structure design definition', 'structural system design definition')/]
[/CLASS(product_definition_shape, 'structural system design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.20.4.1 description

AIM element: product_definition_shape.description
Source: ISO 10303-41

5.1.20.4.2 corrugated_structure_design_definition to corrugated_structure_boundary_relationship (as border)

NOTE Shape_aspect.description should specify the index of the corrugated_structure_boundary_relationship in the border list

AIM element: PATH
Rules: 5.2.4.35
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect.description = 'i']
[shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'corrugated structure boundary relationship')/}]}

5.1.20.4.3 corrugated_structure_design_definition to structural_system_adjacency_relationship (as border)

NOTE Shape_aspect.description should specify the index of the structural_system_adjacency_relationship in the border list

AIM element: PATH
Rules: 5.2.4.35
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect.description = 'i']
[shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'structural system adjacency relationship')/}]}

5.1.20.4.4 corrugation_width

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.223
Reference path: product_definition_shape <=
property_definition
property_definition = represented_definition <-
/SDR_NAME('corrugated structure design parameters')/ ->
representation
representation.items[i] ->

/VAL_REP_ITEM('corrugation width', length_measure)/

5.1.20.4.5 thickness

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.287
Reference path: product_definition_shape <=
property_definition
property_definition = represented_definition <-
/SDR_NAME('corrugated structure design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('thickness', length_measure)/

5.1.20.4.6 corrugated_structure_design_definition to external_instance_reference (as moulded_surface)

AIM element: PATH
Rules: 5.2.4.37
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'moulded surface')/]
[/EXT_INST_REF(shape_aspect, 'ship moulded form schema', 'moulded form')/]}

5.1.20.4.7 corrugated_structure_design_definition to surface (as moulded_surface)

AIM element: PATH
Rules: 5.2.4.37
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'moulded surface')/]
[shape_aspect = represented_definition <-
/SDR_NAME('moulded surface representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', surface)/]}

5.1.20.4.8 corrugated_structure_design_definition to non_manifold_surface_shape (as moulded_surface)

AIM element: PATH
Rules: 5.2.4.37
Reference path: product_definition_shape <-
shape_aspect.of_shape

shape_aspect
{[/ROOT_CLASS(shape_aspect, 'moulded surface')/]
[shape_aspect = represented_definition <-
/PDR_NAME('moulded surface representation')/ ->
representation =>
shape_representation =>
non_manifold_surface_shape]}

5.1.20.4.9 corrugated_structure_design_definition to corrugated_structure (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'corrugated structure')/}

5.1.20.4.10 tightness

AIM element: descriptive_representation_item.description
Source: ISO 10303-41
Rules: 5.2.4.287, 5.2.4.359
Reference path: product_definition_shape <=
property_definition
property_definition = represented_definition <-
/SDR_NAME('corrugated structure design parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('tightness')/
{(descriptive_representation_item.description = ‘unspecified’)
(descriptive_representation_item.description = ‘non tight’)
(descriptive_representation_item.description = ‘air tight’)
(descriptive_representation_item.description = ’fume tight’)
(descriptive_representation_item.description = ’gas tight’)
(descriptive_representation_item.description = ‘oil tight’)
(descriptive_representation_item.description = ‘water tight’)
(descriptive_representation_item.description = ‘weather tight’)
(descriptive_representation_item.description = ‘user defined’)}

5.1.20.4.11 user_defined_tightness

AIM element: descriptive_representation_item.description
Source: ISO 10303-41
Rules: 5.2.4.223
Reference path: product_definition_shape <=
property_definition
property_definition = represented_definition <-
/SDR_NAME('corrugated structure design parameters')/ ->
representation
representation.items[i] ->

/DSC_REP_ITEM('user defined tightness')/

5.1.20.4.12 corrugated_structure_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('corrugated structure design representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.20.4.13 corrugated_structure_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('corrugated structure design representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.20.4.14 corrugated_structure_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('corrugated structure design representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.20.4.15 corrugated_structure_design_definition to advanced_brep_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('corrugated structure design representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.20.4.16 definition to derived_unit (as local_units)

AIM element: /SUPERTYPE(structural_system_design_definition)/ (See 5.1.20.18.9)

5.1.20.4.17 definition to named_unit (as local_units)

AIM element: /SUPERTYPE(structural_system_design_definition)/ (See 5.1.20.18.10)

5.1.20.4.18 corrugated_structure_design_definition to global_id (as id)

AIM element: /SUPERTYPE(structural_system_design_definition)/ (See 5.1.20.18.11)

5.1.20.4.19 version_id

AIM element: /SUPERTYPE(structural_system_design_definition)/ (See 5.1.20.18.12)

5.1.20.5 CORRUGATED_STRUCTURE_FUNCTIONAL_DEFINITION

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.152
Reference path: {[/CLASS(property_definition, 'corrugated structure functional definition', 'structural system functional definition')/]
[/CLASS(property_definition, 'structural system functional definition', 'functional definition')/]
[/CLASS(property_definition, 'functional definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.20.5.1 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.20.5.2 corrugated_structure_functional_definition to corrugated_-structure (as defined_for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'corrugated structure')/}

5.1.20.5.3 the_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.357, 5.2.4.288
Reference path: property_definition = represented_definition <-
/PDR_NAME('corrugated structure function parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'function'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description = 'deck')
(descriptive_representation_item.description = 'strength deck')
(descriptive_representation_item.description = 'accomodation deck')
(descriptive_representation_item.description = 'main deck')
(descriptive_representation_item.description = 'weather deck')
(descriptive_representation_item.description = 'navigation deck')
(descriptive_representation_item.description = 'platform deck')
(descriptive_representation_item.description = 'deck in superstructure')
(descriptive_representation_item.description = 'bulkhead')
(descriptive_representation_item.description = 'longitudinal bulkhead')
(descriptive_representation_item.description = 'transversal bulkhead')
(descriptive_representation_item.description = 'strength bulkhead')
(descriptive_representation_item.description = 'wash bulkhead')
(descriptive_representation_item.description = 'collision bulkhead')
(descriptive_representation_item.description = 'hold bulkhead')
(descriptive_representation_item.description = 'tank bulkhead')
(descriptive_representation_item.description = 'wing bulkhead')
(descriptive_representation_item.description = 'superstructure front bulkhead')
(descriptive_representation_item.description = 'superstructure side bulkhead')
(descriptive_representation_item.description = 'superstructure aft bulkhead')
(descriptive_representation_item.description = 'stool')
(descriptive_representation_item.description = 'wall')
(descriptive_representation_item.description = 'hatch cover')
(descriptive_representation_item.description = 'user defined')}

5.1.20.5.4 user_def_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.224
Reference path: property_definition = represented_definition <-
/PDR_NAME('corrugated structure function parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('user def function')/

5.1.20.5.5 corrugated_structure_functional_definition to global_id (as id)

AIM element: /SUPERTYPE(structural_system_functional_definition)/ (See 5.1.20.19.5)

5.1.20.5.6 version_id

AIM element: /SUPERTYPE(structural_system_functional_definition)/ (See 5.1.20.19.6)

5.1.20.6 CORRUGATED_STRUCTURE_RELATIONSHIP

#1: corrugated_structure_relationship is a corrugated_structure_boundary_-relationship

AIM element: #1: /SUBTYPE(corrugated_structure_boundary_relationship)/ (See 5.1.20.3)

5.1.20.6.1 description

AIM element: #1: /SUBTYPE(corrugated_structure_boundary_relationship)/ - (See 5.1.20.3.1)

5.1.20.6.2 corrugated_structure_relationship to corrugated_structure (as item_1)

AIM element: #1: /SUBTYPE(corrugated_structure_boundary_relationship)/ - (See 5.1.20.3.4)

5.1.20.6.3 corrugated_structure_relationship to external_instance_-reference (as external_item_1)

AIM element: #1: /SUBTYPE(corrugated_structure_boundary_relationship)/ - (See 5.1.20.3.5)

5.1.20.6.4 corrugated_structure_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(corrugated_structure_boundary_relationship)/ (See 5.1.20.3.2)

5.1.20.6.5 corrugated_structure_relationship to external_instance_-reference (as external_item_2)

AIM element: #1: /SUBTYPE(corrugated_structure_boundary_relationship)/ (See 5.1.20.3.6)

5.1.20.6.6 corrugated_structure_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(corrugated_structure_boundary_relationship)/ (See 5.1.20.3.7)

5.1.20.6.7 version_id

AIM element: #1: /SUBTYPE(corrugated_structure_boundary_relationship)/ (See 5.1.20.3.8)

5.1.20.7 PANEL_SYSTEM

AIM element: [product_definition]
[group]
Source: ISO 10303-41
Reference path: {[/LINK_TO_GROUP(product_definition)/]
[/CLASS(product_definition, 'panel system', 'structural system')/]
[/CLASS(product_definition, 'structural system', 'system')/]
[/CLASS(product_definition, 'system', 'item structure')/]
[/CLASS(product_definition, 'item structure', 'definable object')/]
[/CLASS(product_definition, 'item structure', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]
[/CLASS(product_definition, 'system', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]}

5.1.20.7.1 the_class

AIM element: group.name
Source: ISO 10303-41
Reference path: product_definition
classification_item = product_definition
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = ‘class membership‘}
classification_assignment.assigned_classification ->
group
group.name
{group.name = ‘panel system class’}

5.1.20.7.2 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.20.7.3 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.20.7.4 panel_system to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
external_identification_item = product_definition
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(document, ‘document reference with address’)/})

5.1.20.7.5 panel_system to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product,'ship')/}

5.1.20.7.6 panel_system to external_instance_reference (as external_items)

AIM element: PATH
Rules: 5.2.4.76
Reference path: group <-
/GROUPS(product_definition, 'item structure')/
{([/CLASS_ID(product_definition, 'structural part')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'structural part')/])
([/CLASS_ID(product_definition, 'panel system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'panel system')/])}

5.1.20.7.7 panel_system to external_instance_reference (as external_-relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{([/CLASS_ID(product_definition_relationship, 'structural part relationship')/]
[/EXT_INST_REF(product_definition_relationship, 'ship structures schema',
'structural part relationship')/])
([/CLASS_ID(product_definition_relationship, 'structural system relationship')/]
[/EXT_INST_REF(product_definition_relationship, 'ship structures schema',
'structural system relationship')/])}

5.1.20.7.8 panel_system to item (as items)

AIM element: PATH
Rules: 5.2.4.76
Reference path: group <-
/GROUPS(product_definition, 'item structure')/
{(/CLASS_ID(product_definition, 'structural part')/)
(/CLASS_ID(product_definition, 'panel system')/)}

5.1.20.7.9 panel_system to item_relationship (as relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{(/CLASS_ID(product_definition_relationship, 'structural part relationship')/)
(/CLASS_ID(product_definition_relationship, 'structural system relationship')/)}

5.1.20.7.10 panel_system to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]

applied_identification_assignment

5.1.20.7.11 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(group)/

5.1.20.8 PANEL_SYSTEM_BOUNDARY_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition_relationship, 'panel system boundary relationship', 'panel system relationship')/]
[/CLASS(product_definition_relationship, 'panel system relationship', 'structural system relationship')/]
[/CLASS(product_definition_relationship, 'structural system relationship', 'system relationship')/]
[/CLASS(product_definition_relationship, 'system relationship', 'item relationship')/]
[/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/ROOT_CLASS(product_definition_relationship, 'definable object')/]
[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]}

5.1.20.8.1 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.20.8.2 panel_system_boundary_relationship to item (as item_2)

#1: related item is a panel_system
#2: related item is a profile
#3: related item is a seam

AIM element: #1: product_definition_relationship.related_product_definition
#2: product_definition_relationship.related_product_definition
#3: PATH
Source: #1: ISO 10303-41
#2: ISO 10303-41
Reference path: #1: {product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('panel system')/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-

{/CLASS_ID(shape_aspect, 'moulded surface')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'panel system boundary relationship')/}}

#2: {[product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('profile')/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{(/CLASS_ID(shape_aspect, 'top line')/)
(/CLASS_ID(shape_aspect, 'profile cross section')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'panel system boundary relationship')/}]}

#3: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'seam')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'panel system boundary relationship')/}

5.1.20.8.3 displacement

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.233
Reference path: product_definition_relationship
product_definition_relationship.relating_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect <-
{/CLASS_ID(shape_aspect, 'border')/}
shape_aspect_relationship =
{/CLASS_ID(shape_aspect_relationship, 'panel system boundary relationship')/}

represented_definition <-
/PDR_NAME('panel system boundary relationship design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('displacement', length_measure)/

5.1.20.8.4 panel_system_boundary_relationship to panel_system (as item_-1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, 'panel system')/}}

5.1.20.8.5 panel_system_boundary_relationship to external_instance_-reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'panel system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'panel system')/]}

5.1.20.8.6 panel_system_boundary_relationship to external_instance_-reference (as external_item_2)

#1: related item is a panel_system
#2: related item is a profile
#3: related item is a seam

AIM element: #1: product_definition_relationship.related_product_definition
#2: product_definition_relationship.related_product_definition
#3: PATH
Source: #1: ISO 10303-41
#2: ISO 10303-41
Reference path: #1: {product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'panel system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'panel system')/]}

#2: {product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'profile')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'profile')/]}

#3: product_definition_relationship

product_definition_relationship.related_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/CLASS_ID(shape_aspect, 'seam')/]
[/EXT_INST_REF(shape_aspect, 'ship structures schema', 'seam')/]}

5.1.20.8.7 panel_system_boundary_relationship to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_relationship
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.8.8 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.20.9 PANEL_SYSTEM_CURVE_BOUNDARY

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.332
Reference path: {[/ROOT_CLASS(shape_aspect, 'panel system curve boundary')/]}

5.1.20.9.1 displacement

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.218
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('panel system curve boundary design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('displacement', length_measure)/

5.1.20.9.2 curve

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.294
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('panel system curve boundary design parameters')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('curve', bounded_curve)/

5.1.20.10 PANEL_SYSTEM_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.118
Reference path: {[/CLASS(product_definition_shape, 'panel system design definition', 'structural system design definition')/]
[/CLASS(product_definition_shape, 'structural system design definition', 'system design definition')/]
[/CLASS(product_definition_shape, 'system design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.20.10.1 description

AIM element: product_definition_shape.description
Source: ISO 10303-41

5.1.20.10.2 material_offset

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.295
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('panel system design parameters')/ ->
representation representation.items[i] ->
/VAL_REP_ITEM('material offset', ratio_measure)/

5.1.20.10.3 stiffener_spacing

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.248
Reference path: product_definition_shape <=
property_definition =

represented_definition <-
/PDR_NAME('panel system design parameters')/ ->
representation representation.items[i] ->
/VAL_REP_ITEM('stiffener spacing', length_measure)/

5.1.20.10.4 thickness

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215 ,5.2.4.295
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('panel system design parameters')/ ->
representation representation.items[i] ->
/VAL_REP_ITEM('thickness', length_measure)/

5.1.20.10.5 panel_system_design_definition to panel_system (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'panel system')/}

5.1.20.10.6 panel_system_design_definition to external_instance_reference (as moulded_surface)

AIM element: PATH
Rules: 5.2.4.75
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'moulded surface')/]
[/EXT_INST_REF(shape_aspect, 'ship moulded form schema', 'moulded form')/]}

5.1.20.10.7 panel_system_design_definition to non_manifold_surface_shape (as moulded_surface)

AIM element: PATH
Rules: 5.2.4.75
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'moulded surface')/]
[shape_aspect = represented_definition <-
/PDR_NAME('moulded surface representation')/ ->
representation =>

shape_representation =>
non_manifold_surface_shape]}

5.1.20.10.8 panel_system_design_definition to surface (as moulded_surface)

AIM element: PATH
Rules: 5.2.4.75
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'moulded surface')/]
[shape_aspect = represented_definition <-
/SDR_NAME('moulded surface representation')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('.UNUSED.', surface)/]}

5.1.20.10.9 panel_system_design_definition to panel_system_-boundary_relationship (as border)

NOTE Shape_aspect.description should specify the index of the panel_system_boundary_relationship in the border list

AIM element: PATH
Rules: 5.2.4.73
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect.description = 'i']
[shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'panel system boundary relationship')/}]}

5.1.20.10.10 panel_system_design_definition to panel_system_curve_boundary (as border)

NOTE Shape_aspect.description should specify the index of the panel_system_curve_boundary in the border list

AIM element: PATH
Rules: 5.2.4.73
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/CLASS_ID(shape_aspect, 'panel system curve boundary')/]

[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect.description = 'i']}

5.1.20.10.11 panel_system_design_definition to panel_system_plane_boundary (as border)

NOTE Shape_aspect.description should specify the index of the panel_system_plane_boundary in the border list

AIM element: PATH
Rules: 5.2.4.73
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/CLASS_ID(shape_aspect, 'panel system plane boundary')/]
[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect.description = 'i']}

5.1.20.10.12 panel_system_design_definition to structural_system_adjacency_relationship (as border)

NOTE Shape_aspect.description should specify the index of the structural_system_adjacency_relationship in the border list

AIM element: PATH
Rules: 5.2.4.73
Reference path: product_definition_shape <-
shape_aspect.of_shape
shape_aspect
{[/ROOT_CLASS(shape_aspect, 'boundary')/]
[shape_aspect.description = 'i']
[shape_aspect <-
shape_aspect_relationship.relating_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'structural system adjacency relationship')/}]}

5.1.20.10.13 tightness

AIM element: descriptive_representation_item.description
Source: ISO 10303-41
Rules: 5.2.4.359, 5.2.4.295
Reference path: product_definition_shape <=
property_definition
property_definition = represented_definition <-
/SDR_NAME('panel system design parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('tightness')/

{(descriptive_representation_item.description = ‘unspecified’)
(descriptive_representation_item.description = ‘non tight’)
(descriptive_representation_item.description = ‘air tight’)
(descriptive_representation_item.description = ’fume tight’)
(descriptive_representation_item.description = ’gas tight’)
(descriptive_representation_item.description = ‘oil tight’)
(descriptive_representation_item.description = ‘water tight’)
(descriptive_representation_item.description = ‘weather tight’)
(descriptive_representation_item.description = ‘user defined’)}

5.1.20.10.14 user_defined_tightness

AIM element: descriptive_representation_item.description
Source: ISO 10303-41
Rules: 5.2.4.248
Reference path: product_definition_shape <=
property_definition
property_definition = represented_definition <-
/SDR_NAME('panel sytsem design parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('user defined tightness')/

5.1.20.10.15 panel_system_design_definition to non_manifold_surface_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME(‘panel system design representation’)/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.20.10.16 panel_system_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME(‘panel system design representation’)/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.20.10.17 panel_system_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('panel system design representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.20.10.18 panel_system_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('panel system design representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.20.10.19 panel_system_design_definition to derived_unit (as local_-units)

AIM element: /SUPERTYPE(structural_system_design_definition)/ (See 5.1.20.18.9)

5.1.20.10.20 panel_system_design_definition to named_unit (as local_-units)

AIM element: /SUPERTYPE(structural_system_design_definition)/ (See 5.1.20.18.10)

5.1.20.10.21 panel_system_design_definition to global_id (as id)

AIM element: /SUPERTYPE(structural_system_design_definition)/ (See 5.1.20.18.11)

5.1.20.10.22 version_id

AIM element: /SUPERTYPE(structural_system_design_definition)/ (See 5.1.20.18.12)

5.1.20.11 PANEL_SYSTEM_FUNCTIONAL_DEFINITION

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.159
Reference path: {[/CLASS(property_definition, 'panel system functional definition', 'structural system functional definition')/]
[/CLASS(property_definition, 'structural system functional definition', 'functional definition')/]
[/CLASS(property_definition, 'functional definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.20.11.1 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.20.11.2 panel_system_functional_definition to panel_system (as defined_for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'panel system')/}

5.1.20.11.3 the_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.357, 5.2.4.296
Reference path: property_definition = represented_definition <-
/PDR_NAME('panel system function parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'function'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description = 'deck')
(descriptive_representation_item.description = 'strength deck')
(descriptive_representation_item.description = 'accomodation deck')
(descriptive_representation_item.description = 'main deck')
(descriptive_representation_item.description = 'weather deck')
(descriptive_representation_item.description = 'navigation deck')

(descriptive_representation_item.description = 'platform deck')
(descriptive_representation_item.description = 'deck in superstructure')
(descriptive_representation_item.description = 'bottom')
(descriptive_representation_item.description = 'inner bottom')
(descriptive_representation_item.description = 'double bottom')
(descriptive_representation_item.description = 'outer shell')
(descriptive_representation_item.description = 'inner shell')
(descriptive_representation_item.description = 'double shell')
(descriptive_representation_item.description = 'bulkhead')
(descriptive_representation_item.description = 'longitudinal bulkhead')
(descriptive_representation_item.description = 'transversal bulkhead')
(descriptive_representation_item.description = 'strength bulkhead')
(descriptive_representation_item.description = '**wash bulkhead**')
(descriptive_representation_item.description = 'collision bulkhead')
(descriptive_representation_item.description = 'hold bulkhead')
(descriptive_representation_item.description = 'tank bulkhead')
(descriptive_representation_item.description = 'wing bulkhead')
(descriptive_representation_item.description = 'superstructure front bulkhead')
(descriptive_representation_item.description = 'superstructure side bulkhead')
(descriptive_representation_item.description = 'superstructure aft bulkhead')
(descriptive_representation_item.description = 'stool')
(descriptive_representation_item.description = 'wall')
(descriptive_representation_item.description = 'transom')
(descriptive_representation_item.description = 'sheer strake')
(descriptive_representation_item.description = 'tank bottom')
(descriptive_representation_item.description = 'tank side')
(descriptive_representation_item.description = 'tank top')
(descriptive_representation_item.description = 'hatch cover')
(descriptive_representation_item.description = 'keel')
(descriptive_representation_item.description = 'duct keel')
(descriptive_representation_item.description = 'hatchway coaming')
(descriptive_representation_item.description = 'hatchway sidecoaming')
(descriptive_representation_item.description = 'hatchway endcoaming')
(descriptive_representation_item.description = 'upper boom')
(descriptive_representation_item.description = 'lower boom')
(descriptive_representation_item.description = 'frame')
(descriptive_representation_item.description = 'web frame')
(descriptive_representation_item.description = 'vertical web frame')
(descriptive_representation_item.description = 'transverse web frame')
(descriptive_representation_item.description = 'floor')
(descriptive_representation_item.description = 'transverse floor')
(descriptive_representation_item.description = 'bracket')
(descriptive_representation_item.description = 'hopper')
(descriptive_representation_item.description = 'stern frame')
(descriptive_representation_item.description = 'user defined')}

5.1.20.11.4 user_def_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.234
Reference path: property_definition = represented_definition <-
/PDR_NAME('panel system function parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('user def function')/

5.1.20.11.5 panel_system_functional_definition to global_id (as id)

AIM element: /SUPERTYPE(structural_system_functional_definition)/ (See 5.1.20.19.5)

5.1.20.11.6 version_id

AIM element: /SUPERTYPE(structural_system_functional_definition)/ (See 5.1.20.19.6)

5.1.20.12 PANEL_SYSTEM_PLANE_BOUNDARY

AIM element: shape_aspect
Source: ISO 10303-41
Rules: 5.2.4.333
Reference path: {/ROOT_CLASS(shape_aspect, 'panel system plane boundary')/}

5.1.20.12.1 plane

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.297
Reference path: shape_aspect = represented_definition <-
/PDR_NAME('panel system plane boundary design parameters')/ ->
representation representation.items[i] ->
/GEO_REP_ITEM('plane', plane)/

5.1.20.13 PANEL_SYSTEM_RELATIONSHIP

#1: panel_system_relationship is a panel_system_boundary_relationship

AIM element: #1: /SUBTYPE(panel_system_boundary_relationship)/ (See 5.1.20.8)

5.1.20.13.1 description

AIM element: #1: /SUBTYPE(panel_system_boundary_relationship)/ (See 5.1.20.8.1)

5.1.20.13.2 panel_system_relationship to panel_system (as item_1)

AIM element: #1: /SUBTYPE(panel_system_boundary_relationship)/ (See 5.1.20.8.4)

5.1.20.13.3 panel_system_relationship to external_instance_reference (as external_item_1)

AIM element: #1: /SUBTYPE(panel_system_boundary_relationship)/ (See 5.1.20.8.5)

5.1.20.13.4 panel_system_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(panel_system_boundary_relationship)/ (See 5.1.20.8.2)

5.1.20.13.5 panel_system_relationship to external_instance_reference (as external_item_2)

AIM element: #1: /SUBTYPE(panel_system_boundary_relationship)/ (See 5.1.20.8.6)

5.1.20.13.6 panel_system_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(panel_system_boundary_relationship)/ (See 5.1.20.8.7)

5.1.20.13.7 version_id

AIM element: #1: /SUBTYPE(panel_system_boundary_relationship)/ (See 5.1.20.8.8)

5.1.20.14 PLATE_STRAKE

AIM element: [product_definition]
[group]
Source: ISO 10303-41
Reference path: {[/LINK_TO_GROUP(product_definition)/]
[/CLASS(product_definition, 'plate strake', 'structural system')/]
[/CLASS(product_definition, 'structural system', 'system')/]
[/CLASS(product_definition, 'system', 'item structure')/]
[/CLASS(product_definition, 'item structure', 'definable object')/]
[/CLASS(product_definition, 'item structure', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]
[/CLASS(product_definition, 'system', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]}

5.1.20.14.1 the_class

AIM element: group.name
Source: ISO 10303-41
Reference path: product_definition
classification_item = product_definition
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = ‘class membership‘}
classification_assignment.assigned_classification ->
group
group.name
{group.name = ‘plate strake class’}

5.1.20.14.2 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.20.14.3 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.20.14.4 plate_strake to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
external_identification_item = product_definition
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(document, ‘document reference with address’)/})

5.1.20.14.5 plate_strake to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product,'ship')/}

5.1.20.14.6 plate_strake to external_instance_reference (as external_items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition, 'item structure')/
{[/CLASS_ID(product_definition, 'plate')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'plate')/]}

5.1.20.14.7 plate_strake to external_instance_reference (as external_-relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{[/CLASS_ID(product_definition_relationship, 'plate relationship')/]
[/EXT_INST_REF(product_definition_relationship, 'ship structures schema', 'plate relationship')/]}

5.1.20.14.8 plate_strake to item (as items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition, 'item structure')/
{/CLASS_ID(product_definition, 'plate')/}

5.1.20.14.9 plate_strake to item_relationship (as relationships)

from ITEM_STRUCTURE

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{/CLASS_ID(product_definition_relationship, 'plate relationship')/}

5.1.20.14.10 plate_strake to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.14.11 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.20.15 PLATE_STRAKE_FUNCTIONAL_DEFINITION

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.161
Reference path: {[/CLASS(property_definition, 'plate strake functional definition', 'structural system functional definition')/]
[/CLASS(property_definition, 'structural system functional definition', 'functional definition')/]
[/CLASS(property_definition, 'functional definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.20.15.1 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.20.15.2 plate_strake_functional_definition to plate_strake (as defined_-for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'plate strake')/}

5.1.20.15.3 the_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.357, 5.2.4.299
Reference path: property_definition = represented_definition <-
/PDR_NAME(‘plate strake function parameters’)/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'function'}
representation_item =>
descriptive_representation_item
{[(descriptive_representation_item.description = 'plating')
(descriptive_representation_item.description = 'transom')
(descriptive_representation_item.description = 'user defined')]}

5.1.20.15.4 user_def_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.251
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('plate strake function parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('user def function')/

5.1.20.15.5 plate_strake_functional_definition to global_id (as id)

AIM element: /SUPERTYPE(structural_system_functional_definition)/ (See 5.1.20.19.5)

5.1.20.15.6 version_id

AIM element: /SUPERTYPE(structural_system_functional_definition)/ (See 5.1.20.19.6)

5.1.20.16 STRUCTURAL_SYSTEM

AIM element: [product_definition]
[group]
Source: ISO 10303-41
Reference path: {[/LINK_TO_GROUP(product_definition)/]
[/CLASS(product_definition, 'structural system', 'system')/]
[/CLASS(product_definition, 'system', 'item structure')/]
[/CLASS(product_definition, 'item structure', 'definable object')/]
[/CLASS(product_definition, 'item structure', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]
[/CLASS(product_definition, 'system', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]
[(/CLASS(product_definition, 'general structural system', 'structural system')/)
(/CLASS(product_definition, 'section', 'structural system')/)
(/CLASS(product_definition, 'panel system', 'structural system')/)
(/CLASS(product_definition, 'corrugated structure', 'structural system')/)
(/CLASS(product_definition, 'block', 'structural system')/)
(/CLASS(product_definition, 'plate strake', 'structural system')/)
(/CLASS(product_definition, 'built profile', 'structural system')/)
(/CLASS(product_definition, 'user defined', 'structural system')/)]}

5.1.20.16.1 the_class

AIM element: group.name
Source: ISO 10303-41
Reference path: product_definition
classification_item = product_definition
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = ‘class membership‘}
classification_assignment.assigned_classification ->
group
group.name
{(group.name = 'general structural system class')
(group.name = 'section class')
(group.name = 'panel system class')
(group.name = 'corrugated structure class')
(group.name = 'block class')
(group.name = 'plate strake class')
(group.name = 'built profile class')
(group.name = 'user defined class')}

5.1.20.16.2 user_def_class

AIM element: group.name
Source: ISO 10303-41
Rules: 5.2.4.358
Reference path: product_definition
classification_item = product_definition
classification_item <-
applied_classification_assignment.items[i]
applied_classification_assignment
applied_classification_assignment <=
classification_assignment
{classification_assignment.role ->
classification_role
classification_role.name = ‘user defined class‘}
classification_assignment.assigned_classification ->
group
group.name

5.1.20.16.3 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.20.16.4 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.20.16.5 structural_system to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
external_identification_item = product_definition
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(document, ‘document reference with address’)/})

5.1.20.16.6 structural_system to ship (as ship_context)

AIM element: PATH
Source: ISO 10303-41
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, 'ship')/}

5.1.20.16.7 structural_system to external_instance_reference (as external_items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition, 'item structure')/
{([/CLASS_ID(product_definition, 'structural part')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'structural part')/])
([/CLASS_ID(product_definition, 'structural system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'structural system')/])}

5.1.20.16.8 structural_system to external_instance_reference (as external_relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{([/CLASS_ID(product_definition_relationship, 'structural part relationship')/]
[/EXT_INST_REF(product_definition_relationship, 'ship structures schema', 'structural part relationship')/])
([/CLASS_ID(product_definition_relationship, 'structural system relationship')/]
[/EXT_INST_REF(product_definition_relationship, 'ship structures schema', 'structural system relationship')/])}

5.1.20.16.9 structural_system to item (as items)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition, 'item structure')/

{(/CLASS_ID(product_definition, 'structural part')/)
(/CLASS_ID(product_definition, 'structural system')/)}

5.1.20.16.10 structural_system to item_relationship (as relationships)

AIM element: PATH
Reference path: group <-
/GROUPS(product_definition_relationship, 'item structure')/
{(/CLASS_ID(product_definition_relationship, 'structural part relationship')/)
(/CLASS_ID(product_definition_relationship, 'structural system relationship')/)}

5.1.20.16.11 structural_system to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.16.12 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.20.17 STRUCTURAL_SYSTEM_ADJACENCY_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition_relationship, 'structural system adjacency relationship', 'structural system relationship')/]
[/CLASS(product_definition_relationship, 'structural system relationship', 'system relationship')/]
[/CLASS(product_definition_relationship, 'system relationship', 'item relationship')/]
[/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/ROOT_CLASS(product_definition_relationship, 'definable object')/]
[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]}

5.1.20.17.1 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.20.17.2 boundary_index

AIM element: shape_aspect
Source: ISO 10303-41
Reference path: product_definition_relationship
product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('structural system')/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect
{(/CLASS_ID(shape_aspect, 'border')/)
(/EXT_INST_REF(shape_aspect, 'ship structures schema', 'border')/)}

5.1.20.17.3 displacement

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.221
Reference path: product_definition_relationship
product_definition_relationship.relating_product_definition ->
/PDCD/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-
shape_aspect <-
{/CLASS_ID(shape_aspect, 'border')/}
shape_aspect_relationship =
represented_definition <-
/PDR_NAME(‘structural system adjacency relationship design parameters‘)/ ->
representation
{[representation.name = '.UNUSED.']
[representation.id = '.UNUSED.']}
representation.items[i] ->
/VAL_REP_ITEM('displacement', length_measure)/

5.1.20.17.4 structural_system_adjacency_relationship to structural_-system (as item_2)

AIM element: product_definition_relationship.related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.related_product_definition ->
/PDCD_CLASS_ID('structural system')/ <-
property_definition.definition
property_definition =>
product_definition_shape
product_definition_shape <-

shape_aspect.of_shape
shape_aspect <-
{/CLASS_ID(shape_aspect, 'border')/}
shape_aspect_relationship.related_shape_aspect
shape_aspect_relationship
{/CLASS_ID(shape_aspect_relationship, 'structural system adjacency relationship')/}}

5.1.20.17.5 structural_system_adjacency_relationship to structural_-system (as item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, ‘structural system’)/}}

5.1.20.17.6 structural_system_adjacency_relationship to external_-instance_reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
[/CLASS_ID(product_definition, ‘structural system’)/]
[/EXT_INST_REF(product_definition, 'ship structures schema', ‘structural system’)/]}

5.1.20.17.7 structural_system_adjacency_relationship to external_-instance_reference (as external_item_2)

AIM element: product_definition_relationship.related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'structural system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'structural system')/]}

5.1.20.17.8 structural_system_adjacency_relationship to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_relationship
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.17.9 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.20.18 STRUCTURAL_SYSTEM_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.130
Reference path: {[/CLASS(product_definition, 'structural system design definition', 'design definition')/]
[/CLASS(product_definition, 'design definition', 'definition')/]
[/CLASS(product_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition, 'versionable object')/]}

5.1.20.18.1 description

AIM element: product_definition_shape.description
Source: ISO 10303-41

5.1.20.18.2 tightness

AIM element: descriptive_representation_item.description
Source: ISO 10303-41
Rules: 5.2.4.359, 5.2.4.303
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('structural system design parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('tightness')/
{(descriptive_representation_item.description = ‘unspecified’)
(descriptive_representation_item.description = ‘non tight’)
(descriptive_representation_item.description = ‘air tight’)
(descriptive_representation_item.description = ’fume tight’)
(descriptive_representation_item.description = ’gas tight’)
(descriptive_representation_item.description = ‘oil tight’)
(descriptive_representation_item.description = ‘water tight’)
(descriptive_representation_item.description = ‘weather tight’)
(descriptive_representation_item.description = ‘user defined’)}

5.1.20.18.3 user_defined_tightness

AIM element: descriptive_representation_item.description
Source: ISO 10303-41
Rules: 5.2.4.256
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('structural system design parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('user defined tightness')/

5.1.20.18.4 structural_system_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('structural system design representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.20.18.5 structural_system_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('structural system design representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.20.18.6 structural_system_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('structural system design representation')/ ->
property_definition_representation.used_representation ->

representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.20.18.7 structural_system_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition <-
represented_definition <-
/SDR_NAME('structural system design representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.20.18.8 structural_system_design_definition to structural_system (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
/PROP_TO_PROD_DEF/
{/CLASS_ID(product_definition, 'structural system')/}

5.1.20.18.9 structural_system_design_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.20.18.10 structural_system_design_definition to named_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.20.18.11 structural_system_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.137
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.18.12 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.20.19 STRUCTURAL_SYSTEM_FUNCTIONAL_DEFINITION

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.164
Reference path: {[/CLASS(property_definition, 'structural system functional definition', 'functional definition')/]
[/CLASS(property_definition, 'functional definition', 'definition')/]
[/CLASS(property_definition, 'definition', 'versionable object')/]
[/ROOT_CLASS(property_definition, 'versionable object')/]}

5.1.20.19.1 description

AIM element: property_definition.description
Source: ISO 10303-41

5.1.20.19.2 structural_system_functional_definition to structural_system (as defined_for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'structural system')/}

5.1.20.19.3 the_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.357, 5.2.4.304
Reference path: property_definition =
represented_definition <-
/PDR_NAME('structural system function parameters')/ ->
representation
representation.items[i] ->
representation_item
{representation_item.name = 'function'}
representation_item =>
descriptive_representation_item
{(descriptive_representation_item.description = 'ship structure')
(descriptive_representation_item.description = 'deck')
(descriptive_representation_item.description = 'strength deck')
(descriptive_representation_item.description = 'accomodation deck')
(descriptive_representation_item.description = 'main deck')
(descriptive_representation_item.description = 'weather deck')
(descriptive_representation_item.description = 'navigation deck')
(descriptive_representation_item.description = 'platform deck')
(descriptive_representation_item.description = 'deck in superstructure')
(descriptive_representation_item.description = 'bottom')
(descriptive_representation_item.description = 'inner bottom')
(descriptive_representation_item.description = 'double bottom')
(descriptive_representation_item.description = 'outer shell')
(descriptive_representation_item.description = 'inner shell')
(descriptive_representation_item.description = 'double shell')
(descriptive_representation_item.description = 'bulkhead')
(descriptive_representation_item.description = 'longitudinal bulkhead')
(descriptive_representation_item.description = 'transversal bulkhead')
(descriptive_representation_item.description = 'strength bulkhead')
(descriptive_representation_item.description = 'wash bulkhead')
(descriptive_representation_item.description = 'collision bulkhead')
(descriptive_representation_item.description = 'hold bulkhead')
(descriptive_representation_item.description = 'tank bulkhead')
(descriptive_representation_item.description = 'wing bulkhead')
(descriptive_representation_item.description = 'superstructure front bulkhead')
(descriptive_representation_item.description = 'superstructure side bulkhead')
(descriptive_representation_item.description = 'superstructure aft bulkhead')
(descriptive_representation_item.description = 'stool')
(descriptive_representation_item.description = 'wall')
(descriptive_representation_item.description = 'plating')
(descriptive_representation_item.description = 'transom')
(descriptive_representation_item.description = 'deck house')
(descriptive_representation_item.description = 'superstructure')
(descriptive_representation_item.description = 'mid ship')
(descriptive_representation_item.description = 'fore ship')

(descriptive_representation_item.description = 'aft ship')
(descriptive_representation_item.description = 'engine area')
(descriptive_representation_item.description = 'cargo area')
(descriptive_representation_item.description = 'accomodation area')
(descriptive_representation_item.description = 'machinery casing')
(descriptive_representation_item.description = 'sheer strake')
(descriptive_representation_item.description = 'tank bottom')
(descriptive_representation_item.description = 'tank side')
(descriptive_representation_item.description = 'tank top')
(descriptive_representation_item.description = 'hatch cover')
(descriptive_representation_item.description = 'keel')
(descriptive_representation_item.description = 'duct keel')
(descriptive_representation_item.description = 'hatchway coaming')
(descriptive_representation_item.description = 'hatchway sidecoaming')
(descriptive_representation_item.description = 'hatchway endcoaming')
(descriptive_representation_item.description = 'upper boom')
(descriptive_representation_item.description = 'lower boom')
(descriptive_representation_item.description = 'frame')
(descriptive_representation_item.description = 'web frame')
(descriptive_representation_item.description = 'vertical web frame')
(descriptive_representation_item.description = 'transverse web frame')
(descriptive_representation_item.description = 'floor')
(descriptive_representation_item.description = 'transverse floor')
(descriptive_representation_item.description = 'cross tie')
(descriptive_representation_item.description = 'stringer')
(descriptive_representation_item.description = 'girder')
(descriptive_representation_item.description = 'longitudinal girder')
(descriptive_representation_item.description = 'deck beam')
(descriptive_representation_item.description = 'engine foundation')
(descriptive_representation_item.description = 'bracket')
(descriptive_representation_item.description = 'hopper')
(descriptive_representation_item.description = 'stern frame')
(descriptive_representation_item.description = 'user defined')}

5.1.20.19.4 user_def_function

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.257
Reference path: property_definition =
represented_definition <-
/PDR_NAME('structural system function parameters')/ ->
representation
representation.items[i] ->
/DSC_REP_ITEM('user def function')/

5.1.20.19.5 structural_system_functional_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.169
Reference path: property_definition
identification_item = property_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.19.6 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.20.20 STRUCTURAL_SYSTEM_RELATIONSHIP

#1: structural_system_relationship is a corrugated_structure_relationship
#2: structural_system_relationship is a panel_system_relationship
#3: structural_system_relationship is a structural_system_adjacency_relationship
#4: structural_system_relationship is a structural_system_symmetry_relationship

AIM element: #1: /SUBTYPE(corrugated_structure_relationship)/ (See 5.1.20.6)
#2: /SUBTYPE(panel_system_relationship)/ (See 5.1.20.13)
#3: /SUBTYPE(structural_system_adjacency_relationship)/ (See 5.1.20.17)
#4: /SUBTYPE(structural_system_symmetry_relationship)/ (See 5.1.20.21)

5.1.20.20.1 description

AIM element: #1: /SUBTYPE(corrugated_structure_relationship)/ (See 5.1.20.6.1)
#2: /SUBTYPE(panel_system_relationship)/ (See 5.1.20.13.1)
#3: /SUBTYPE(structural_system_adjacency_relationship)/ (See 5.1.20.17.3)
#4: /SUBTYPE(structural_system_symmetry_relationship)/ (See 5.1.20.21.1)

5.1.20.20.2 structural_system_relationship to structural_system (as item_-1)

AIM element: #1: /SUBTYPE(corrugated_structure_relationship)/ (See 5.1.20.6.2)
#2: /SUBTYPE(panel_system_relationship)/ (See 5.1.20.13.2)
#3: /SUBTYPE(structural_system_adjacency_relationship)/ (See 5.1.20.17.5)
#4: /SUBTYPE(structural_system_symmetry_relationship)/ (See 5.1.20.21.4)

5.1.20.20.3 structural_system_relationship to external_instance_reference (as external_item_1)

AIM element: #1: /SUBTYPE(corrugated_structure_relationship)/ (See 5.1.20.6.3)
#2: /SUBTYPE(panel_system_relationship)/ (See 5.1.20.13.3)
#3: /SUBTYPE(structural_system_adjacency_relationship)/ (See 5.1.20.17.6)
#4: /SUBTYPE(structural_system_symmetry_relationship)/ (See 5.1.20.21.5)

5.1.20.20.4 structural_system_relationship to item (as item_2)

AIM element: #1: /SUBTYPE(corrugated_structure_relationship)/ (See 5.1.20.6.4)
#2: /SUBTYPE(panel_system_relationship)/ (See 5.1.20.13.4)
#3: /SUBTYPE(structural_system_adjacency_relationship)/ (See 5.1.20.17.4)
#4: /SUBTYPE(structural_system_symmetry_relationship)/ (See 5.1.20.21.3)

5.1.20.20.5 structural_system_relationship to external_instance_reference (as external_item_2)

AIM element: #1: /SUBTYPE(corrugated_structure_relationship)/ (See 5.1.20.6.5)
#2: /SUBTYPE(panel_system_relationship)/ (See 5.1.20.13.5)
#3: /SUBTYPE(structural_system_adjacency_relationship)/ (See 5.1.20.17.7)
#4: /SUBTYPE(structural_system_symmetry_relationship)/ (See 5.1.20.21.6)

5.1.20.20.6 structural_system_relationship to global_id (as id)

AIM element: #1: /SUBTYPE(corrugated_structure_relationship)/ (See 5.1.20.6.6)
#2: /SUBTYPE(panel_system_relationship)/ (See 5.1.20.13.6)
#3: /SUBTYPE(structural_system_adjacency_relationship)/ (See 5.1.20.17.8)
#4: /SUBTYPE(structural_system_symmetry_relationship)/ (See 5.1.20.21.7)

5.1.20.20.7 version_id

AIM element: #1: /SUBTYPE(corrugated_structure_relationship)/ (See 5.1.20.6.7)
#2: /SUBTYPE(panel_system_relationship)/ (See 5.1.20.13.7)
#3: /SUBTYPE(structural_system_adjacency_relationship)/ (See 5.1.20.17.9)
#4: /SUBTYPE(structural_system_symmetry_relationship)/ (See 5.1.20.21.8)

5.1.20.21 STRUCTURAL_SYSTEM_SYMMETRY_RELATIONSHIP

AIM element: product_definition_relationship
Source: ISO 10303-41
Rules: 5.2.4.355
Reference path: {[/CLASS(product_definition_relationship, 'structural system symmetry relationship', 'structural system relationship')/]
[/CLASS(product_definition_relationship, 'structural system relationship', 'system relationship')/]
[/CLASS(product_definition_relationship, 'system relationship', 'item relationship')/]

[/CLASS(product_definition_relationship, 'item relationship', 'definable object')/]
[/ROOT_CLASS(product_definition_relationship, 'definable object')/]
[/CLASS(product_definition_relationship, 'item relationship', 'versionable object')/]
[/ROOT_CLASS(product_definition_relationship, 'versionable object')/]}

5.1.20.21.1 description

AIM element: product_definition_relationship.description
Source: ISO 10303-41

5.1.20.21.2 mirroring_plane

AIM element: PATH
Rules: 5.2.4.215, 5.2.4.211
Reference path: product_definition_relationship
product_definition_relationship = characterized_product_definition
characterized_product_definition = characterized_definition <-
property_definition.definition
property_definition
property_definition = represented_definition <-
/PDR('structural system symmetry relationship design parameters')/ ->
representation
representation.items[i] ->
/GEO_REP_ITEM('mirroring plane', plane)/

5.1.20.21.3 structural_system_symmetry_relationship to structural_system (as item_2)

AIM element: product_definition_relationship.related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.related_product_definition ->
product_definition
{/CLASS_ID(product_definition, 'structural system')/}}

5.1.20.21.4 structural_system_symmetry_relationship to structural_system (as item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, ‘structural system’)/}}

5.1.20.21.5 structural_system_symmetry_relationship to external_-instance_reference (as external_item_1)

AIM element: product_definition_relationship.relating_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.relating_product_definition ->
product_definition
[/CLASS_ID(product_definition, ‘structural system’)/]
[/EXT_INST_REF(product_definition, 'ship structures schema', ‘structural system’)/]}

5.1.20.21.6 structural_system_symmetry_relationship to external_-instance_reference (as external_item_2)

AIM element: product_definition_relationship.related_product_definition
Source: ISO 10303-41
Reference path: {product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, 'structural system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'structural system')/]}

5.1.20.21.7 structural_system_symmetry_relationship to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_relationship
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.20.21.8 version_id

AIM element: applied_identification_assignment.assigned_id
Source: ISO 10303-218
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_relationship)/

5.1.21 welds UoF

5.1.21.1 BEVELED_GROOVE_WELD

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.94
Reference path: {[/CLASS(product_definition_shape, ‘beveled groove weld‘, ‘groove weld‘)/]
[/CLASS(product_definition_shape, ‘groove weld‘, ‘welded joint design definition‘)/]
[/CLASS(product_definition_shape, ‘welded joint design definition‘, ‘structural part joint design definition‘)/]
[/CLASS(product_definition_shape, ‘structural part joint design definition‘, ‘design definition‘)/]
[/CLASS(product_definition_shape, ‘design definition‘, ‘definition‘)/]
[/CLASS(product_definition_shape, ‘definition‘, ‘versionable object‘)/]
[/ROOT_CLASS(product_definition_shape, ‘versionable object‘)/]}

5.1.21.1.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.1.2 endcut_shape

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.283
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘beveled groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘endcut shape‘)/
{(descriptive_representation_item.description = ‘flare single V‘)
(descriptive_representation_item.description = ‘flare double V‘)
(descriptive_representation_item.description = ‘flare single bevel‘)
(descriptive_representation_item.description = ‘flare double bevel‘)
(descriptive_representation_item.description = ‘single bevel‘)
(descriptive_representation_item.description = ‘double bevel‘)
(descriptive_representation_item.description = ‘single V‘)
(descriptive_representation_item.description = ‘double V‘)
(descriptive_representation_item.description = ‘single J‘)
(descriptive_representation_item.description = ‘double J‘)

(descriptive_representation_item.description = ‘single U‘)
(descriptive_representation_item.description = ‘double U‘)
(descriptive_representation_item.description = ‘user defined‘)}

5.1.21.1.3 taper

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.283
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘beveled groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘taper‘)/
{(descriptive_representation_item.description = ‘non taper‘)
(descriptive_representation_item.description = ‘one side taper‘)
(descriptive_representation_item.description = ‘both sides taper‘)}

5.1.21.1.4 taper_angle

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.242
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘beveled groove weld design parameters’)/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM(’taper_angle’, plane_angle_measure)/

5.1.21.1.5 sidedness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.283
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘beveled groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘sidedness’)/
{(descriptive_representation_item.description = ‘one side‘)
(descriptive_representation_item.description = ‘both sides‘)}

5.1.21.1.6 weld_backing_type

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.242
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('beveled groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('weld backing type')/
{(descriptive_representation_item.description = 'none')
(descriptive_representation_item.description = 'permanent')
(descriptive_representation_item.description = 'copper backing bar')
(descriptive_representation_item.description = 'ceramic tape')
(descriptive_representation_item.description = 'flare backing ring')
(descriptive_representation_item.description = 'permanent backing ring')
(descriptive_representation_item.description = 'removable backing ring')
(descriptive_representation_item.description = 'user defined')}

5.1.21.1.7 weld_joint_spacer

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.283
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('beveled groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('weld joint spacer')/
{(descriptive_representation_item.description = 'TRUE')
(descriptive_representation_item.description = 'FALSE')}

5.1.21.1.8 configuration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.283
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('beveled groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('configuration')/
{(descriptive_representation_item.description = 'butt')

(descriptive_representation_item.description = ‘tee‘)
(descriptive_representation_item.description = ‘corner‘)
(descriptive_representation_item.description = ‘lap‘)
(descriptive_representation_item.description = ‘cruciform‘)
(descriptive_representation_item.description = ‘edge joint‘)
(descriptive_representation_item.description = ‘stud‘)
(descriptive_representation_item.description = ‘surfacing‘)
(descriptive_representation_item.description = ‘flare groove‘)
(descriptive_representation_item.description = ‘user defined‘)}

5.1.21.1.9 penetration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.283
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘beveled groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘penetration’)/
{(descriptive_representation_item.description = ‘full‘)
(descriptive_representation_item.description = ‘partial‘)}

5.1.21.1.10 beveled_groove_weld to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘beveled groove weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.1.11 beveled_groove_weld to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘beveled groove weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>

shape_representation =>
non_manifold_surface_shape_representation

5.1.21.1.12 beveled_groove_weld to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('beveled groove weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.1.13 beveled_groove_weld to geometrically_bounded_wireframe_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('beveled groove weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.1.14 beveled_groove_weld to welded_joint (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition =
characterized_product_definition =
product_definition <-
product_definition_relationship.related_product_definition
product_definition_relationship
{/CLASS_ID(product_definition_relationship, 'welded joint')/}

5.1.21.1.15 joint_orientation

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.283
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘beveled groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘joint orientation‘)/
{(descriptive_representation_item.description = ‘face face‘)
(descriptive_representation_item.description = ‘face edge‘)
(descriptive_representation_item.description = ‘edge edge‘)}

5.1.21.1.16 tightness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.283
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('beveled groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘tightness’)/
{(descriptive_representation_item.description = ‘non tight’)
(descriptive_representation_item.description = ‘air tight’)
(descriptive_representation_item.description = ‘fume tight’)
(descriptive_representation_item.description = ‘oil tight’)
(descriptive_representation_item.description = ‘weather tight’)
(descriptive_representation_item.description = ‘water tight’)
(descriptive_representation_item.description = ‘undefined’)}

5.1.21.1.17 beveled_groove_weld to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.1.18 beveled_groove_weld to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.21.1.19 beveled_groove_weld to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.21.1.20 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.21.2 BUTT_GROOVE_WELD

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.95
Reference path: {[/CLASS(product_definition_shape, 'butt groove weld', 'groove weld')/]
[/CLASS(product_definition_shape, 'groove weld', 'welded joint design definition')/]
[/CLASS(product_definition_shape, 'welded joint design definition', 'structural part joint design definition')/]
[/CLASS(product_definition_shape, 'structural part joint design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.21.2.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.2.2 face_shape

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.284
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘butt groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('face shape’)/
{(descriptive_representation_item.description = ‘square‘)
(descriptive_representation_item.description = ‘scarf‘)
(descriptive_representation_item.description = ‘user defined‘)}

5.1.21.2.3 sidedness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.284
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘butt groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘sidedness’)/
{(descriptive_representation_item.description = ‘one side‘)
(descriptive_representation_item.description = ‘both sides‘)}

5.1.21.2.4 weld_backing_type

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.243
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘butt groove weld design parameters’)/ ->
representation

representation.items [i] ->
/DSC_REP_ITEM('weld backing type')/
{(descriptive_representation_item.description = 'none')
(descriptive_representation_item.description = 'permanent')
(descriptive_representation_item.description = 'copper backing bar')
(descriptive_representation_item.description = 'ceramic tape')
(descriptive_representation_item.description = 'flare backing ring')
(descriptive_representation_item.description = 'permanent backing ring')
(descriptive_representation_item.description = 'removable backing ring')
(descriptive_representation_item.description = 'user defined')}

5.1.21.2.5 weld_joint_spacer

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.284
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('butt groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('weld joint spacer')/
{(descriptive_representation_item.description = 'TRUE')
(descriptive_representation_item.description = 'FALSE')}

5.1.21.2.6 configuration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.284
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('butt groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('configuration')/
{(descriptive_representation_item.description = 'butt')
(descriptive_representation_item.description = 'tee')
(descriptive_representation_item.description = 'corner')
(descriptive_representation_item.description = 'lap')
(descriptive_representation_item.description = 'cruciform')
(descriptive_representation_item.description = 'edge joint')
(descriptive_representation_item.description = 'stud')
(descriptive_representation_item.description = 'surfacing')
(descriptive_representation_item.description = 'flare groove')
(descriptive_representation_item.description = 'user defined')}

5.1.21.2.7 penetration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.284
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('butt groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('penetration')/
{(descriptive_representation_item.description = 'full')
(descriptive_representation_item.description = 'partial')}

5.1.21.2.8 butt_groove_weld to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('butt groove weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.21.2.9 butt_groove_weld to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('butt groove weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.2.10 butt_groove_weld to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-

/SDR_NAME(‘butt groove weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.2.11 butt_groove_weld to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘butt groove weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.2.12 butt_groove_weld to welded_joint (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition =
characterized_product_definition =
product_definition <-
product_definition_relationship.related_product_definition
product_definition_relationship
{/CLASS_ID(product_definition_relationship, ’welded joint’)/}

5.1.21.2.13 joint_orientation

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.284
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘butt groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘joint orientation‘)/
{(descriptive_representation_item.description = ‘face face‘)
(descriptive_representation_item.description = ‘face edge‘)
(descriptive_representation_item.description = ‘edge edge‘)}

5.1.21.2.14 tightness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.284
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('butt groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('tightness')/
{(descriptive_representation_item.description = 'non tight')
(descriptive_representation_item.description = 'air tight')
(descriptive_representation_item.description = 'fume tight')
(descriptive_representation_item.description = 'oil tight')
(descriptive_representation_item.description = 'weather tight')
(descriptive_representation_item.description = 'water tight')
(descriptive_representation_item.description = 'undefined')}

5.1.21.2.15 butt_groove_weld to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.2.16 butt_groove_weld to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.21.2.17 butt_groove_weld to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.21.2.18 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.21.3 CONTINUOUS_FILLET_WELD

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.98
Reference path: {[/CLASS(product_definition_shape, ‘continuous fillet weld‘, ‘fillet weld‘)/]
[/CLASS(product_definition_shape, ‘fillet weld‘, ‘welded joint design definition‘)/]
[/CLASS(product_definition_shape, ‘welded joint design definition‘, ‘structural part joint design definition‘)/]
[/CLASS(product_definition_shape, ‘structural part joint design definition‘, ‘design definition‘)/]
[/CLASS(product_definition_shape, ‘design definition‘, ‘definition‘)/]
[/CLASS(product_definition_shape, ‘definition‘, ‘versionable object‘)/]
[/ROOT_CLASS(product_definition_shape, ‘versionable object‘)/]}

5.1.21.3.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.3.2 endcut_shape_type

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.286
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘continuous fillet weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘endcut shape type’)/
{(descriptive_representation_item.description = ‘fillet V‘)
(descriptive_representation_item.description = ‘fillet J‘)
(descriptive_representation_item.description = ‘user defined‘)}

5.1.21.3.3 sidedness

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.286
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('continuous fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('sidedness')/
{(descriptive_representation_item.description = 'one side')
(descriptive_representation_item.description = 'both sides')}

5.1.21.3.4 configuration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.286
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('continuous fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('configuration')/
{(descriptive_representation_item.description = 'butt')
(descriptive_representation_item.description = 'tee')
(descriptive_representation_item.description = 'corner')
(descriptive_representation_item.description = 'lap')
(descriptive_representation_item.description = 'cruciform')
(descriptive_representation_item.description = 'edge joint')
(descriptive_representation_item.description = 'stud')
(descriptive_representation_item.description = 'surfacing')
(descriptive_representation_item.description = 'flare groove')
(descriptive_representation_item.description = 'user defined')}

5.1.21.3.5 penetration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.286
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('continuous fillet weld design parameters')/ ->
representation
representation.items [i] ->

/DSC_REP_ITEM('penetration')/
{(descriptive_representation_item.description = 'full')
(descriptive_representation_item.description = 'partial')}

5.1.21.3.6 continuous_fillet_weld to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('continuous fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.3.7 continuous_fillet_weld to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('continuous fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.21.3.8 continuous_fillet_weld to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('continuous fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.3.9 continuous_fillet_weld to geometrically_bounded_wireframe_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘continuous fillet weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.3.10 continuous_fillet_weld to welded_joint (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition =
characterized_product_definition =
product_definition <-
product_definition_relationship.related_product_definition
product_definition_relationship
{/CLASS_ID(product_definition_relationship, ’welded joint’)/}

5.1.21.3.11 joint_orientation

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.286
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘continuous fillet weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘joint orientation‘)/
{(descriptive_representation_item.description = ‘face face‘)
(descriptive_representation_item.description = ‘face edge‘)
(descriptive_representation_item.description = ‘edge edge‘)}

5.1.21.3.12 tightness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.286
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('continuous fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('tightness')/
{(descriptive_representation_item.description = 'non tight')
(descriptive_representation_item.description = 'air tight')
(descriptive_representation_item.description = 'fume tight')
(descriptive_representation_item.description = 'oil tight')
(descriptive_representation_item.description = 'weather tight')
(descriptive_representation_item.description = 'water tight')
(descriptive_representation_item.description = 'undefined')}

5.1.21.3.13 continuous_fillet_weld to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.3.14 continuous_fillet_weld to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.21.3.15 continuous_fillet_weld to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.21.3.16 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.21.4 ELECTRODE_CHEMICAL_COMPOSITION

AIM element: representation
Source: ISO 10303-43
Reference path: {/ROOT_CLASS(representation, 'electrode chemical composition')/}

5.1.21.4.1 aluminium

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('aluminium', ratio_measure)/

5.1.21.4.2 carbon

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('carbon', ratio_measure)/

5.1.21.4.3 chromium

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM('chromium', ratio_measure)/

5.1.21.4.4 copper

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->

/VAL_REP_ITEM(‘copper‘, ratio_measure)/

5.1.21.4.5 manganese

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM(‘manganese‘, ratio_measure)/

5.1.21.4.6 molybdenum

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM(‘molybdenum‘, ratio_measure)/

5.1.21.4.7 nickel

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM(‘nickel‘, ratio_measure)/

5.1.21.4.8 phosphorus

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM(‘phosphorus‘, ratio_measure)/

5.1.21.4.9 silicon

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM(‘silicon‘, ratio_measure)/

5.1.21.4.10 sulphur

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM(‘sulphur‘, ratio_measure)/

5.1.21.4.11 titanium

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM(‘titanium‘, ratio_measure)/

5.1.21.4.12 vanadium

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM(‘vanadium‘, ratio_measure)/

5.1.21.4.13 zirconium

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.187
Reference path: representation
representation.items [i] ->
/VAL_REP_ITEM(‘zirconium‘, ratio_measure)/

5.1.21.5 FILLET_WELD

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.105
Reference path: {[/CLASS(product_definition_shape, ‘fillet weld‘, ‘welded joint design definition‘)/]
[/CLASS(product_definition_shape, ‘welded joint design definition‘, ‘structural part joint design definition‘)/]
[/CLASS(product_definition_shape, ‘structural part joint design definition‘, ‘design definition‘)/]
[/CLASS(product_definition_shape, ‘design definition‘, ‘definition‘)/]
[/CLASS(product_definition_shape, ‘definition‘, ‘versionable object‘)/]

[/ROOT_CLASS(product_definition_shape, ‘versionable object‘)/]}

5.1.21.5.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.5.2 endcut_shape_type

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.289
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘fillet weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘endcut shape type’)/
{(descriptive_representation_item.description = ‘fillet V‘)
(descriptive_representation_item.description = ‘fillet J‘)
(descriptive_representation_item.description = ‘user defined‘)}

5.1.21.5.3 sidedness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.289
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘fillet weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘sidedness’)/
{(descriptive_representation_item.description = ‘one side‘)
(descriptive_representation_item.description = ‘both sides‘)}

5.1.21.5.4 configuration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.289
Reference path: product_definition_shape <=
property_definition =
represented_definition <-

/PDR_NAME('fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('configuration')/
{(descriptive_representation_item.description = 'butt')
(descriptive_representation_item.description = 'tee')
(descriptive_representation_item.description = 'corner')
(descriptive_representation_item.description = 'lap')
(descriptive_representation_item.description = 'cruciform')
(descriptive_representation_item.description = 'edge joint')
(descriptive_representation_item.description = 'stud')
(descriptive_representation_item.description = 'surfacing')
(descriptive_representation_item.description = 'flare groove')
(descriptive_representation_item.description = 'user defined')}

5.1.21.5.5 penetration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.289
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('penetration')/
{(descriptive_representation_item.description = 'full')
(descriptive_representation_item.description = 'partial')}

5.1.21.5.6 fillet_weld to welded_joint (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition =
characterized_product_definition =
product_definition <-
product_definition_relationship.related_product_definition
product_definition_relationship
{/CLASS_ID(product_definition_relationship, 'welded joint')/}

5.1.21.5.7 fillet_weld to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.21.5.8 fillet_weld to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.5.9 fillet_weld to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.5.10 fillet_weld to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.5.11 joint_orientation

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.289
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘fillet weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘joint orientation‘)/
{(descriptive_representation_item.description = ‘face face‘)
(descriptive_representation_item.description = ‘face edge‘)
(descriptive_representation_item.description = ‘edge edge‘)}

5.1.21.5.12 tightness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.289
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘tightness’)/
{(descriptive_representation_item.description = ‘non tight’)
(descriptive_representation_item.description = ‘air tight’)
(descriptive_representation_item.description = ‘fume tight’)
(descriptive_representation_item.description = ‘oil tight’)
(descriptive_representation_item.description = ‘weather tight’)
(descriptive_representation_item.description = ‘water tight’)
(descriptive_representation_item.description = ‘undefined’)}

5.1.21.5.13 fillet_weld to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.5.14 fillet_weld to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=

property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.21.5.15 fillet_weld to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.21.5.16 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.21.6 GROOVE_WELD

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.107
Reference path: {[/CLASS(product_definition_shape, 'groove weld', 'welded joint design definition')/]
[/CLASS(product_definition_shape, 'welded joint design definition', 'structural part joint design definition')/]
[/CLASS(product_definition_shape, 'structural part joint design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.21.6.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.6.2 sidedness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.290
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('sidedness')/
{(descriptive_representation_item.description = 'one side')
(descriptive_representation_item.description = 'both sides')}

5.1.21.6.3 weld_backing_type

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.246
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('weld backing type')/
{(descriptive_representation_item.description = 'none')
(descriptive_representation_item.description = 'permanent')
(descriptive_representation_item.description = 'copper backing bar')
(descriptive_representation_item.description = 'ceramic tape')
(descriptive_representation_item.description = 'flare backing ring')
(descriptive_representation_item.description = 'permanent backing ring')
(descriptive_representation_item.description = 'removable backing ring')
(descriptive_representation_item.description = 'user defined')}

5.1.21.6.4 weld_joint_spacer

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.290
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('weld joint spacer')/
{(descriptive_representation_item.description = 'TRUE')

(descriptive_representation_item.description = ‘FALSE‘)}

5.1.21.6.5 configuration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.290
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘configuration’)/
{(descriptive_representation_item.description = ‘butt‘)
(descriptive_representation_item.description = ‘tee‘)
(descriptive_representation_item.description = ‘corner‘)
(descriptive_representation_item.description = ‘lap‘)
(descriptive_representation_item.description = ‘cruciform‘)
(descriptive_representation_item.description = ‘edge joint‘)
(descriptive_representation_item.description = ‘stud‘)
(descriptive_representation_item.description = ‘surfacing‘)
(descriptive_representation_item.description = ‘flare groove‘)
(descriptive_representation_item.description = ‘user defined‘)}

5.1.21.6.6 penetration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.290
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘penetration’)/
{(descriptive_representation_item.description = ‘full‘)
(descriptive_representation_item.description = ‘partial‘)}

5.1.21.6.7 groove_weld to welded_joint (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition =
characterized_product_definition =
product_definition <-

product_definition_relationship.related_product_definition
product_definition_relationship
{/CLASS_ID(product_definition_relationship, 'welded joint')/}

5.1.21.6.8 groove_weld to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition= property_definition
represented_definition <-
/SDR_NAME('groove weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.21.6.9 groove_weld to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition= property_definition
represented_definition <-
/SDR_NAME('groove weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.6.10 groove_weld to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition= property_definition
represented_definition <-
/SDR_NAME('groove weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.6.11 groove_weld to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition= property_definition
represented_definition <-
/SDR_NAME(‘groove weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.6.12 joint_orientation

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.290
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘groove weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘joint orientation‘)/
{(descriptive_representation_item.description = ‘face face‘)
(descriptive_representation_item.description = ‘face edge‘)
(descriptive_representation_item.description = ‘edge edge‘)}

5.1.21.6.13 tightness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.290
Reference path: product_definition_shape <=
property_definition
represented_definition= property_definition
represented_definition <-
/PDR_NAME('groove weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘tightness’)/
{(descriptive_representation_item.description = ‘non tight’)
(descriptive_representation_item.description = ‘air tight’)
(descriptive_representation_item.description = ‘fume tight’)
(descriptive_representation_item.description = ‘oil tight’)
(descriptive_representation_item.description = ‘weather tight’)
(descriptive_representation_item.description = ‘water tight’)
(descriptive_representation_item.description = ‘undefined’)}

5.1.21.6.14 groove_weld to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.6.15 groove_weld to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.21.6.16 groove_weld to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.21.6.17 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.21.7 INTERMITTENT_FILLET_WELD

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.109
Reference path: {[/CLASS(product_definition_shape, 'intermittent fillet weld', 'fillet weld')/]
[/CLASS(product_definition_shape, 'fillet weld', 'welded joint design definition')/]
[/CLASS(product_definition_shape, 'welded joint design definition', 'structural part joint design definition')/]
[/CLASS(product_definition_shape, 'structural part joint design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]

[/CLASS(product_definition_shape, ‘definition‘, ‘versionable object‘)/]
[/ROOT_CLASS(product_definition_shape, ‘versionable object‘)/]}

5.1.21.7.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.7.2 cutout_rules

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘intermittent fillet weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘cutout rules’)/
{(descriptive_representation_item.description = ‘none‘)
(descriptive_representation_item.description = ‘fixed rule‘)
(descriptive_representation_item.description = ‘member depth‘)
(descriptive_representation_item.description = ‘percent length‘)}

5.1.21.7.3 end_rules

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘intermittent fillet weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘end rules’)/
{(descriptive_representation_item.description = ‘none‘)
(descriptive_representation_item.description = ‘fixed rule‘)
(descriptive_representation_item.description = ‘member depth‘)
(descriptive_representation_item.description = ‘percent length‘)}

5.1.21.7.4 fillet_alignment

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('intermittent fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('fillet alignment')/
{(descriptive_representation_item.description = 'staggered')
(descriptive_representation_item.description = 'chained')}

5.1.21.7.5 fillet_weld_length

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('intermittent fillet weld design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('fillet weld length', length_measure)/

5.1.21.7.6 fillet_weld_spacing

AIM element: value_representation_item.value _component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('intermittent fillet weld design parameters')/ ->
representation
representation.items [i] ->
/VAL_REP_ITEM('fillet weld spacing', length_measure)/

5.1.21.7.7 penetration_rules

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('intermittent fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('penetration rules')/
{(descriptive_representation_item.description = 'none')
(descriptive_representation_item.description = 'fixed rule')
(descriptive_representation_item.description = 'member depth')
(descriptive_representation_item.description = 'percent length')}

5.1.21.7.8 endcut_shape_type

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('intermittent fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('endcut shape type')/
{(descriptive_representation_item.description = 'fillet V')
(descriptive_representation_item.description = 'fillet J')
(descriptive_representation_item.description = 'user defined')}

5.1.21.7.9 sidedness

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('intermittent fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('sidedness')/
{(descriptive_representation_item.description = 'one side')
(descriptive_representation_item.description = 'both sides')}

5.1.21.7.10 configuration

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('intermittent fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('configuration')/
{(descriptive_representation_item.description = 'butt')
(descriptive_representation_item.description = 'tee')
(descriptive_representation_item.description = 'corner')
(descriptive_representation_item.description = 'lap')
(descriptive_representation_item.description = 'cruciform')
(descriptive_representation_item.description = 'edge joint')
(descriptive_representation_item.description = 'stud')
(descriptive_representation_item.description = 'surfacing')
(descriptive_representation_item.description = 'flare groove')
(descriptive_representation_item.description = 'user defined')}

5.1.21.7.11 penetration

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('intermittent fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('penetration')/
{(descriptive_representation_item.description = 'full')
(descriptive_representation_item.description = 'partial')}

5.1.21.7.12 intermittent_fillet_weld to welded_joint (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition =
characterized_product_definition =
product_definition <-
product_definition_relationship.related_product_definition
product_definition_relationship

{/CLASS_ID(product_definition_relationship, 'welded joint')/}

5.1.21.7.13 intermittent_fillet_weld to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('intermittent fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.7.14 intermittent_fillet_weld to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('intermittent fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.21.7.15 intermittent_fillet_weld to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('intermittent fillet weld shape representation')/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.7.16 intermittent_fillet_weld to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘intermittent fillet weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.7.17 joint_orientation

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘intermittent fillet weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘joint orientation‘)/
{(descriptive_representation_item.description = ‘face face‘)
(descriptive_representation_item.description = ‘face edge‘)
(descriptive_representation_item.description = ‘edge edge‘)}

5.1.21.7.18 tightness

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.293
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('intermittent fillet weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘tightness’)/
{(descriptive_representation_item.description = ‘non tight’)
(descriptive_representation_item.description = ‘air tight’)
(descriptive_representation_item.description = ‘fume tight’)
(descriptive_representation_item.description = ‘oil tight’)
(descriptive_representation_item.description = ‘weather tight’)
(descriptive_representation_item.description = ‘water tight’)
(descriptive_representation_item.description = ‘undefined’)}

5.1.21.7.19 intermittent_fillet_weld to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.7.20 intermittent_fillet_weld to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.21.7.21 intermittent_fillet_weld to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.21.7.22 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.21.8 SPOT_SEAM_WELD

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.128
Reference path: {[/CLASS(product_definition_shape, 'spot seam weld', 'welded joint design definition')/]
[/CLASS(product_definition_shape, 'welded joint design definition', 'structural part joint design definition')/]
[/CLASS(product_definition_shape, 'structural part joint design definition', 'design definition')/]

[/CLASS(product_definition_shape, ‘design definition‘, ‘definition‘)/]
[/CLASS(product_definition_shape, ‘definition‘, ‘versionable object‘)/]
[/ROOT_CLASS(product_definition_shape, ‘versionable object‘)/]}

5.1.21.8.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.8.2 weld_context

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.301
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘spot seam weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘weld context’)/
{(descriptive_representation_item.description = ‘spot weld‘)
(descriptive_representation_item.description = ‘seam weld‘)}

5.1.21.8.3 configuration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.301
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘spot seam weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘configuration’)/
{(descriptive_representation_item.description = ‘butt‘)
(descriptive_representation_item.description = ‘tee‘)
(descriptive_representation_item.description = ‘corner‘)
(descriptive_representation_item.description = ‘lap‘)
(descriptive_representation_item.description = ‘cruciform‘)
(descriptive_representation_item.description = ‘edge joint‘)
(descriptive_representation_item.description = ‘stud‘)
(descriptive_representation_item.description = ‘surfacing‘)
(descriptive_representation_item.description = ‘flare groove‘)
(descriptive_representation_item.description = ‘user defined‘)]}

5.1.21.8.4 penetration

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.301
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘spot seam weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘penetration’)/
{(descriptive_representation_item.description = ‘full‘)
(descriptive_representation_item.description = ‘partial‘)}

5.1.21.8.5 spot_seam_weld to welded_joint (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition =
characterized_product_definition =
product_definition <-
product_definition_relationship.related_product_definition
product_definition_relationship
{/CLASS_ID(product_definition_relationship, ’welded joint’)/}

5.1.21.8.6 spot_seam_weld to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘spot seam weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.21.8.7 spot_seam_weld to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-

/SDR_NAME(‘spot seam weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.8.8 spot_seam_weld to geometrically_bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘spot seam weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.8.9 spot_seam_weld to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘spot seam weld shape representation’)/ ->
property_definition_representation.used_representation ->
{representation.name = 'representations'} =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.8.10 joint_orientation

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.301
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘spot seam weld design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘joint orientation‘)/
{(descriptive_representation_item.description = ‘face face‘)
(descriptive_representation_item.description = ‘face edge‘)
(descriptive_representation_item.description = ‘edge edge‘)}

5.1.21.8.11 tightness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.301
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('spot seam weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('tightness')/
{(descriptive_representation_item.description = 'non tight')
(descriptive_representation_item.description = 'air tight')
(descriptive_representation_item.description = 'fume tight')
(descriptive_representation_item.description = 'oil tight')
(descriptive_representation_item.description = 'weather tight')
(descriptive_representation_item.description = 'water tight')
(descriptive_representation_item.description = 'undefined')}

5.1.21.8.12 spot_seam_weld to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.8.13 spot_seam_weld to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = derived_unit
derived_unit

5.1.21.8.14 spot_seam_weld to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.21.8.15 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.21.9 STRUCTURAL_PART_CONNECTION_IMPLEMENTATION

AIM element: product_definition
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition, 'structural part connection implementation', 'item')/]
[/CLASS(product_definition, 'item', 'definable object')/]
[/ROOT_CLASS(product_definition, 'definable object')/]}

5.1.21.9.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.21.9.2 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.21.9.3 structural_part_connection_implementation to structural_-part_joint (as realization_of)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.related_product_definition
product_definition_relationship
{product_definition_relationship.name = 'realization'}
product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, 'structural part joint')/}

5.1.21.9.4 structural_part_connection_ implementation to external_-reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
product_definition = external_identification_item
external_identification_item <-

applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.21.9.5 structural_part_connection_ implementation to ship (as ship_-context)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, ‘ship‘)/}

5.1.21.9.6 structural_part_connection_ implementation to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.142
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.10 STRUCTURAL_PART_JOINT

AIM element: product_definition
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition, ‘structural part joint‘, ‘item relationship‘)/]
[/CLASS(product_definition, ‘item relationship‘, ‘definable object‘)/]
[/ROOT_CLASS(product_definition, ‘definable object‘)/]
[/CLASS(product_definition, ‘item relationship‘, ‘versionable object‘)/]
[/ROOT_CLASS(product_definition, ‘versionable object‘)/]}

5.1.21.10.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.21.10.2 joint_type

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.209
Reference path: product_definition
characterized_product_definition = product_definition
characterized_definition = characterized_product_definition
characterized_definition <-
property_definition.definition
property_definition =
represented_definition <-
/PDR_NAME('structural part joint design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('joint type')/
{(descriptive_representation_item.description = 'fixed joint')
(descriptive_representation_item.description = 'movable joint')}

5.1.21.10.3 structural_part_joint to structural_part_connection_-implementation (as realization)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = 'realization'}
product_definition_relationship.related_product _definition ->
product_definition
{/CLASS_ID(product_definition, 'structural part connection implementation')/}

5.1.21.10.4 structural_part_joint to item (as item_1)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = 'item 1'}
product_definition_relationship.related_product_definition ->
product_definition
{CLASS_ID(product_definition, 'structural part')}

5.1.21.10.5 structural_part_joint to external_instance_reference (as external_item_1)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘external item 1’}
product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, ‘structural part‘)/]
[/EXT_INST_REF(product_definition, 'ship structures schema', ‘structural part’)/]}

5.1.21.10.6 structural_part_joint to item (as item_2)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘item 2’}
product_definition_relationship.related_product_definition ->
product_definition
{/CLASS_ID(product_definition, ‘structural part‘)/}

5.1.21.10.7 structural_part_joint to external_instance_reference (as external_item_2)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘external item 2’}
product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, ‘structural part‘)/]
[/EXT_INST_REF(product_definition, 'ship structures schema', ‘structural part’)/]}

5.1.21.10.8 structural_part_joint to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.90
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.10.9 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.21.11 STRUCTURAL_PART_JOINT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.129
Reference path: {[/CLASS(product_definition_shape, 'structural part joint design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]

5.1.21.11.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.11.2 joint_orientation

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.302
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('structural part joint design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('joint orientation')/
{(descriptive_representation_item.description = 'face face')
(descriptive_representation_item.description = 'face edge')
(descriptive_representation_item.description = 'edge edge')}

5.1.21.11.3 tightness

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.302
Reference path: product_definition_shape <=

property_definition =
represented_definition <-
/PDR_NAME('structural part joint design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘tightness’)/
{(descriptive_representation_item.description = ‘non tight’)
(descriptive_representation_item.description = ‘air tight’)
(descriptive_representation_item.description = ‘fume tight’)
(descriptive_representation_item.description = ‘oil tight’)
(descriptive_representation_item.description = ‘weather tight’)
(descriptive_representation_item.description = ‘water tight’)
(descriptive_representation_item.description = ‘undefined’)}

5.1.21.11.4 structural_part_joint_design_definition to structural_part_-joint (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
{/CLASS_ID(product_definition, ‘structural part joint‘)/}

5.1.21.11.5 structural_part_joint_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.11.6 structural_part_joint_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.21.11.7 structural_part_joint_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.21.11.8 structural_part_joint_design_definition to advanced_brep_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.11.9 structural_part_joint_design_definition to edge_based_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.11.10 structural_part_joint_design_definition to geometrically_-bounded_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-

/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.11.11 structural_part_joint_design_definition to non_manifold_-surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.21.11.12 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.21.12 WELD

AIM element: product_definition
Source: ISO 10303-41
Reference path: {[/CLASS(product_definition, ‘weld‘, ‘structural part connection implementation‘)/]
[/CLASS(product_definition, ‘structural part connection implementation‘, ‘item‘)/]
[/CLASS(product_definition, ‘item‘, ‘definable object‘)/]
[/ROOT_CLASS(product_definition, ‘definable object‘)/]}

5.1.21.12.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.21.12.2 name

AIM element: product_definition.name
Source: ISO 10303-41
Reference path: /NAME_ASSGN(product_definition)/

5.1.21.12.3 weld to welded_joint (as realization_of)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.related_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘realization’}
product_definition_relationship.relating_product_definition ->
product_definition
{/CLASS_ID(product_definition, ‘welded joint‘)/}

5.1.21.12.4 weld to external_reference (as documentation)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
product_definition = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'documentation']
[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.21.12.5 weld to ship (as ship_context)

AIM element: PATH
Reference path: product_definition
product_definition.formation ->
product_definition_formation
{product_definition_formation.id = 'ship structure'}
product_definition_formation.of_product ->
product
{/CLASS_ID(product, ‘ship‘)/}

5.1.21.12.6 weld to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.13 WELD_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.134
Reference path: {[/CLASS(product_definition_shape, ‘weld design definition‘, ‘design definition‘)/]
[/CLASS(product_definition_shape, ‘design definition‘, ‘definition‘)/]
[/CLASS(product_definition_shape, ‘definition‘, ‘versionable object‘)/]
[/ROOT_CLASS(product_definition_shape, ‘versionable object‘)/]}

5.1.21.13.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.13.2 connection_angle

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.305
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘weld design parameters’)/ ->
representation
representation.items [i] ->
representation_item =>
/VAL_REP_ITEM(‘connection angle‘, plane_angle_measure)/

5.1.21.13.3 end_suspension

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.258
Reference path: product_definition_shape <=
property_definition =

represented_definition <-
/PDR_NAME('weld design parameters')/ ->
representation
representation.items [i] ->
representation_item =>
/VAL_REP_ITEM('end suspension', length_measure)/

5.1.21.13.4 inclination_angle

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.305
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('weld design parameters')/ ->
representation
representation.items [i] ->
representation_item =>
/VAL_REP_ITEM('inclination angle', plane_angle_measure)/

5.1.21.13.5 rotation_angle

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.305
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('weld design parameters')/ ->
representation
representation.items [i] ->
representation_item =>
/VAL_REP_ITEM('rotation angle', plane_angle_measure)/

5.1.21.13.6 shape_of_weld_surface

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.305
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('weld design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('shape of weld surface')/
{(descriptive_representation_item.description = 'flush')
(descriptive_representation_item.description = 'convex')

(descriptive_representation_item.description = ‘concave‘)
(descriptive_representation_item.description = ‘none‘)}

5.1.21.13.7 shrinkage

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.305
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘weld design parameters’)/ ->
representation
representation.items [i] ->
representation_item =>
/VAL_REP_ITEM(‘shrinkage‘, length_measure)/

5.1.21.13.8 start_suspension

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.258
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘weld design parameters’)/ ->
representation
representation.items [i] ->
representation_item =>
/VAL_REP_ITEM(‘start suspension‘, length_measure)/

5.1.21.13.9 weld_cross_section_area

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.215, 5.2.4.258
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘weld design parameters’)/ ->
representation
representation.items [i] ->
representation_item =>
/VAL_REP_ITEM(‘weld cross section area‘, area_measure)/

5.1.21.13.10 weld_size

AIM element: value_representation_item.value_component
Source: ISO 10303-43

Rules: 5.2.4.215, 5.2.4.305
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('weld design parameters')/ ->
representation
representation.items [i] ->
representation_item =>
/VAL_REP_ITEM('weld size', length_measure)/

5.1.21.13.11 weld_geometry

AIM element: curve
Source: ISO 10303-42
Rules: 5.2.4.215, 5.2.4.305
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('weld design parameters')/ ->
representation
representation.items [i] ->
representation_item
{representation_item.name = 'weld geometry'}
representation_item =>
geometric_representation_item =>
curve

5.1.21.13.12 weld_design_definition to weld (as defined_for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition =
characterized_product_definition =
product_definition
{/CLASS_ID(product_definition, 'weld')/}

5.1.21.13.13 weld_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: representation
identification_item = representation <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.13.14 weld_design_definition to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.21.13.15 weld_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.21.13.16 weld_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.13.17 weld_design_definition to edge_based_wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.13.18 weld_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.13.19 weld_design_definition to non_manifold_surface_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
represented_definition = property_definition
represented_definition <-
/SDR_NAME('representation')/
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.21.13.20 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.1.21.14 WELD_FILLER_MATERIAL

AIM element: property_definition
Source: ISO 10303-41
Rules: 5.2.4.165
Reference path: {[/CLASS(property_definition, ‘weld filler material‘, ‘ship material property‘)/]
[/CLASS(property_definition, ‘ship material property‘, ‘definition‘)/]
[/CLASS(property_definition, ‘definition‘, ‘versionable object‘)/]
[ROOT_CLASS(property_definition, ‘versionable object‘)/]}

5.1.21.14.1 description

AIM element: property_definition.description

Source: ISO 10303-41

5.1.21.14.2 chemical_composition_id

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.259
Reference path: property_definition =
represented_definition <-
/PDR_NAME('weld filler material design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('chemical composition id')/

5.1.21.14.3 electrode_id

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.259
Reference path: property_definition =
represented_definition <-
/PDR_NAME('weld filler material design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('electrode id')/

5.1.21.14.4 material_strength_id

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.259
Reference path: property_definition =
represented_definition <-
/PDR_NAME('weld filler material design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('material strength id')/

5.1.21.14.5 notch_impact_work_id

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.259
Reference path: property_definition =
represented_definition <-
/PDR_NAME('weld filler material design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM('notch impact work id')/

5.1.21.14.6 weld_filler_material to electrode_chemical_composition (as chemical_composition)

AIM element: PATH
Reference path: property_definition =
represented_definition <-
/PDR_NAME('chemical composition')/ ->
representation
{/CLASS_ID(representation, 'electrode chemical composition')/}

5.1.21.14.7 weld_filler_material to weld (as defined_for)

AIM element: PATH
Reference path: property_definition
property_definition.definition ->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'weld')/}

5.1.21.14.8 density

AIM element: value_representation_item.value_component
Source: ISO 10303-43
Rules: 5.2.4.306
Reference path: property_definition
represented_definition = property_definition <-
/PDR_NAME('weld filler material design parameters')/ ->
representation
representation.items[i] ->
/VAL_REP_ITEM('density', context_dependent_measure)/

5.1.21.14.9 weld_filler_material to external_reference (as material_-reference)

#1: If as documentation refers to an External_reference

AIM element: PATH
Reference path: (product_definition
product_definition = external_identification_item
external_identification_item <-
applied_external_identification_assignment.items[i]
applied_external_identification_assignment <=
external_identification_assignment <=
identification_assignment
{identification_assignment.role ->
identification_role
{[identification_role.name = 'material reference']

[identification_role.description = 'external reference']})

#2: If as documentation refers to a Document_reference_with_address

AIM element: PATH
Reference path: (product_definition
/DOC_REF(product_definition,'documentation')/
document
{/CLASS_ID(‘document reference with address’)/})

5.1.21.14.10 weld_filler_material to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56, 5.2.4.169
Reference path: property_definition
identification_item = property_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.14.11 weld_filler_material to derived_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit
derived_unit

5.1.21.14.12 weld_filler_material to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = named_unit
named_unit

5.1.21.14.13 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(property_definition)/

5.1.21.15 WELDED_JOINT

AIM element: product_definition
Source: ISO 10303-41

Reference path: {[/CLASS(product_definition, ‘welded joint‘, ‘structural part joint‘)/]
[/CLASS(product_definition, ‘structural part joint‘, ‘item relationship‘)/]
[/CLASS(product_definition, ‘item relationship‘, ‘definable object‘)/]
[/ROOT_CLASS(product_definition, ‘definable object‘)/]
[/CLASS(product_definition, ’item relationship‘, ‘versionable object‘)/]
[/ROOT_CLASS(product_definition, ‘versionable object‘)/]}

5.1.21.15.1 description

AIM element: product_definition.description
Source: ISO 10303-41

5.1.21.15.2 joint_type

AIM element: descriptive_representation _item.description
Source: ISO 10303-45
Rules: 5.2.4.306
Reference path: characterized_product_definition = product_definition
characterized_definition = characterized_product_definition
characterized_definition <-
property_definition.definition
property_definition =
represented_definition <-
/PDR_NAME(‘welded joint design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘joint type’)/
{descriptive_representation_item.description = ’fixed joint’}

5.1.21.15.3 welded_joint to weld (as realization)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘realization’}
product_definition_relationship.related_product _definition ->
product_definition
{/CLASS_ID(product_definition, ‘weld ‘)/}

5.1.21.15.4 welded_joint to external_instance_reference (as external_-item_1)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘external item 1’}
product_definition_relationship.related_product_definition ->
product_definition
[/CLASS_ID(product_definition, ‘structural part‘)/]
[/EXT_INST_REF(product_definition, 'ship structures schema', ‘structural part’)/]}

5.1.21.15.5 welded_joint to external_instance_reference (as external_-item_2)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘external item 2’}
product_definition_relationship.related_product_definition ->
[/CLASS_ID(product_definition, ‘structural part‘)/]
[/EXT_INST_REF(product_definition, 'ship structures schema', ‘structural part’)/]}

5.1.21.15.6 welded_joint to item (as item_1)

AIM element: PATH
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘item 1’}
product_definition_relationship.related_product_definition ->
product_definition
{CLASS_ID(product_definition, ‘structural part‘)}

5.1.21.15.7 welded_joint to item (as item_2)

AIM element: PATH
Source: ISO 10303-41
Reference path: product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘item 2’}
product_definition_relationship.related_product_definition ->
product_definition
{CLASS_ID(product_definition, ‘structural part‘)}

5.1.21.15.8 welded_joint to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition
identification_item = product_definition <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.15.9 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition)/

5.1.21.16 WELDED_JOINT_DESIGN_DEFINITION

AIM element: product_definition_shape
Source: ISO 10303-41
Rules: 5.2.4.132
Reference path: {[/CLASS(product_definition_shape, 'welded joint design definition', 'structural part joint design definition')/]
[/CLASS(product_definition_shape, 'structural part joint design definition', 'design definition')/]
[/CLASS(product_definition_shape, 'design definition', 'definition')/]
[/CLASS(product_definition_shape, 'definition', 'versionable object')/]
[/ROOT_CLASS(product_definition_shape, 'versionable object')/]}

5.1.21.16.1 description

AIM element: property_definition.description
Source: ISO 10303-41
Reference path: product_definition_shape <=
property_definition
property_definition.description

5.1.21.16.2 configuration

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.308
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('welded joint design parameters')/ ->
representation
representation.items [i] ->

/DSC_REP_ITEM(‘configuration‘)/
{(descriptive_representation_item.description = ‘butt‘)
(descriptive_representation_item.description = ‘tee‘)
(descriptive_representation_item.description = ‘corner‘)
(descriptive_representation_item.description = ‘lap‘)
(descriptive_representation_item.description = ‘cruciform‘)
(descriptive_representation_item.description = ‘edge joint‘)
(descriptive_representation_item.description = ‘stud‘)
(descriptive_representation_item.description = ‘surfacing‘)
(descriptive_representation_item.description = ‘flare groove‘)
(descriptive_representation_item.description = ‘user defined‘)}

5.1.21.16.3 penetration

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.308
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘welded joint design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘penetration’)/
{(descriptive_representation_item.description = ‘full‘)
(descriptive_representation_item.description = ‘partial‘)}

5.1.21.16.4 welded_joint_design_definition to non_manifold_surface_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘welded joint shape representation’)/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
non_manifold_surface_shape_representation

5.1.21.16.5 welded_joint_design_definition to edge_based_wireframe_-shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME(‘welded joint shape representation’)/ ->

property_definition_representation.used_representation ->
representation =>
shape_representation =>
edge_based_wireframe_shape_representation

5.1.21.16.6 welded_joint_design_definition to geometrically_bounded_-wireframe_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('welded joint shape representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
geometrically_bounded_wireframe_shape_representation

5.1.21.16.7 welded_joint_design_definition to advanced_brep_shape (as representations)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/SDR_NAME('welded joint shape representation')/ ->
property_definition_representation.used_representation ->
representation =>
shape_representation =>
advanced_brep_shape_representation

5.1.21.16.8 welded_joint_design_definition to welded_joint (as defined_-for)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
property_definition.definition ->
characterized_definition
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'welded joint')/}

5.1.21.16.9 joint_orientation

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.308

Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME(‘welded joint design parameters’)/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘joint orientation‘)/
{(descriptive_representation_item.description = ‘face face‘)
(descriptive_representation_item.description = ‘face edge‘)
(descriptive_representation_item.description = ‘edge edge‘)}

5.1.21.16.10 tightness

AIM element: descriptive_representation_item.description
Source: ISO 10303-45
Rules: 5.2.4.308
Reference path: product_definition_shape <=
property_definition =
represented_definition <-
/PDR_NAME('weld joint design parameters')/ ->
representation
representation.items [i] ->
/DSC_REP_ITEM(‘tightness’)/
{(descriptive_representation_item.description = ‘non tight’)
(descriptive_representation_item.description = ‘air tight’)
(descriptive_representation_item.description = ‘fume tight’)
(descriptive_representation_item.description = ‘oil tight’)
(descriptive_representation_item.description = ‘weather tight’)
(descriptive_representation_item.description = ‘water tight’)
(descriptive_representation_item.description = ‘undefined’)}

5.1.21.16.11 welded_joint_design_definition to global_id (as id)

AIM element: PATH
Rules: 5.2.4.56
Reference path: product_definition_shape
identification_item = product_definition_shape <-
applied_identification_assignment.items[i]
applied_identification_assignment

5.1.21.16.12 welded_joint_design_definition to derived_unit (as local_-units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS(‘local units’)/
unit
unit = derived_unit

derived_unit

5.1.21.16.13 welded_joint_design_definition to named_unit (as local_units)

AIM element: PATH
Reference path: product_definition_shape <=
property_definition
/PROP_DEF_TO_UNITS('local units')/
unit
unit = named_unit
named_unit

5.1.21.16.14 version_id

AIM element: identification_assignment.assigned_id
Source: ISO 10303-41
Rules: 5.2.4.369
Reference path: /VERSION_ID(product_definition_shape)/

5.2 AIM EXPRESS short listing

This clause specifies the EXPRESS schema that uses elements from the integrated resources and the AICs and contains the types, entity specializations, rules, and functions that are specific to this part of ISO 10303. This clause also specifies modifications to the text for constructs that are imported from the integrated resources and the AICs. The definitions and EXPRESS provided in the integrated resources for constructs used in the AIM may include select list items and subtypes that are not imported into the AIM. Requirements stated in the integrated resources that refer to select list items and subtypes apply exclusively to those items that are imported into the AIM.

```
*)
SCHEMA SHIP_STRUCTURES_SCHEMA;
    USE FROM aic_advanced_brep                                -- ISO 10303-514
    (advanced_brep_shape_representation);

    USE FROM aic_edge_based_wireframe                         -- ISO 10303-501
    (edge_based_wireframe_shape_representation);

    USE FROM aic_geometrically_bounded_wireframe              -- ISO 10303-510
    (geometrically_bounded_wireframe_shape_representation);

    USE FROM aic_non_manifold_surface                         -- ISO 10303-508
    (non_manifold_surface_shape_representation);

    USE FROM aic_topologically_bounded_surface                -- ISO 10303-511
    (advanced_face);

    USE FROM action_schema                                    -- ISO 10303-41
    (action,
     action_method,
     action_request_solution,
     executed_action,
     versioned_action_request);

    USE FROM application_context_schema                       -- ISO 10303-41
    (application_context,
```

```
    application_protocol_definition);

  USE FROM approval_schema                                -- ISO 10303-41
  (approval,
   approval_date_time,
   approval_person_organization,
   approval_status);

  USE FROM basic_attribute_schema                         -- ISO 10303-41
  (object_role);

  USE FROM date_time_schema                               -- ISO 10303-41
  (calendar_date,
   date_and_time,
   date_time_role,
   ordinal_date,
   week_of_year_and_day_date);

  USE FROM document_schema                                -- ISO 10303-41
  (document,
   document_representation_type,
   document_usage_constraint);

  USE FROM external_reference_schema                      -- ISO 10303-41
  (external_source,
   external_source_relationship,
   externally_defined_item);

  REFERENCE FROM geometry_schema                          -- ISO 10303-42
  (dummy_gri);

  USE FROM geometric_model_schema                         -- ISO 10303-42
  (faceted_brep,
   geometric_curve_set);

  USE FROM geometry_schema                                -- ISO 10303-42
  (axis1_placement,
   axis2_placement_2d,
   axis2_placement_3d,
   b_spline_curve,
   b_spline_surface,
   b_spline_curve_with_knots,
   b_spline_surface_with_knots,
   bezier_curve,
   bezier_surface,
   bounded_curve,
   cartesian_point,
   composite_curve_on_surface,
   conical_surface,
   curve,
   cylindrical_surface,
   degenerate_pcurve,
   degenerate_toroidal_surface,
   direction,
   elementary_surface,
   ellipse,
   geometric_representation_item,
   circle,
   parabola,
   hyperbola,
   line,
```



```
 plane,
 oriented_surface,
 point_on_curve,
 point_on_surface,
 quasi_uniform_curve,
 quasi_uniform_surface,
 rational_b_spline_curve,
 rational_b_spline_surface,
 spherical_surface,
 surface_of_linear_extrusion,
 surface_of_revolution,
 toroidal_surface,
 uniform_curve,
 uniform_surface,
 vector);

USE FROM group_schema                                    -- ISO 10303-41
(group,
 group_relationship);

USE FROM management_resources_schema                     -- ISO 10303-41
(action_request_assignment,
 approval_assignment,
 classification_assignment,
 classification_role,
 date_and_time_assignment,
 document_reference,
 external_identification_assignment,
 group_assignment,
 identification_assignment,
 identification_assignment_relationship,
 identification_role,
 organization_assignment,
 person_assignment,
 person_and_organization_assignment);

USE FROM material_property_definition_schema             -- ISO 10303-41
(property_definition_relationship);

USE FROM measure_schema                                  -- ISO 10303-41
(amount_of_substance_measure,
 area_measure,
 celsius_temperature_measure,
 context_dependent_measure,
 context_dependent_unit,
 conversion_based_unit,
 count_measure,
 derived_unit,
 derived_unit_element,
 electric_current_measure,
 global_unit_assigned_context,
 length_measure,
 length_unit,
 luminous_intensity_measure,
 mass_measure,
 named_unit,
 plane_angle_measure,
 plane_angle_unit,
 positive_plane_angle_measure,
 positive_length_measure,
 ratio_measure,
```

```
    si_unit,
    solid_angle_measure,
    thermodynamic_temperature_measure,
    thermodynamic_temperature_unit,
    time_measure,
    time_unit,
    volume_measure);

  USE FROM person_organization_schema                    -- ISO 10303-41
  (address,
    person,
    person_and_organization,
    person_and_organization_role,
    personal_address,
    organization,
    organizational_address,
    organizational_project);

  USE FROM product_definition_schema                     -- ISO 10303-41
  (product,
    product_category,
    product_category_relationship,
    product_definition,
    product_definition_relationship,
    product_definition_with_associated_documents,
    product_related_product_category);

  USE FROM product_property_definition_schema            -- ISO 10303-41
  (characterized_object,
    general_property,
    product_definition_shape,
    property_definition,
    shape_aspect,
    shape_aspect_relationship);

  USE FROM product_property_representation_schema        -- ISO 10303-41
  (property_definition_representation,
    shape_definition_representation,
    shape_representation);

  USE FROM qualified_measure_schema                      -- ISO 10303-41
  (descriptive_representation_item);

  USE FROM representation_schema                         -- ISO 10303-43
  (compound_representation_item,
    global_uncertainty_assigned_context,
    set_representation_item,
    list_representation_item,
    item_defined_transformation,
    mapped_item,
    parametric_representation_context,
    representation,
    representation_item,
    representation_map,
    representation_relationship,
    uncertainty_measure_with_unit,
    value_representation_item);

  USE FROM support_resource_schema                       -- ISO 10303-41
  (bag_to_set,
    label);
```



```
   REFERENCE FROM topology_schema                              -- ISO 10303-42
   (dummy_tri,
    edge_loop,
    poly_loop);
(*
```

NOTE - The schemas referenced above can be found in the following parts of ISO 10303:

aic_advanced_brepISO 10303-514
aic_egde_based _wireframe
ISO 10303-501
aic_geometrically_bounded_wireframe
ISO 10303-510
aic_non_manifold_surface
ISO 10303-508
aic_topological_bounded_surface
ISO 10303-511
action_schema ISO 10303-41
application_context_schema
ISO 10303-41
approval_schema ISO 10303-41
basic_attribute_schema
ISO 10303-41
date_time_schema ISO 10303-41
document_schema ISO 10303-41
external_reference_schema
ISO 10303-41
-geometric_model_schema
ISO 10303-42
geometry_schema ISO 10303-42
group_schema ISO 10303-41
management_resources_schema
ISO 10303-41
material_property_definition_schema
ISO 10303-41
measure_schema ISO 10303-41
person_organization_schema
ISO 10303-41
product_definition_schema
ISO 10303-41
product_property_definition_schema
ISO 10303-41
product_property_representation_schema
ISO 10303-41
qualified_measure_schema
ISO 10303-45
representation_schema
ISO 10303-43
support_resource_schema
ISO 10303-41

5.2.1 Fundamental concepts and assumptions

5.2.2 Ship structures types

5.2.2.1 Ship structures type definitions

5.2.2.1.1 action_request_item

An **action_request_item** identifies an **action**, **executed_action**, **group**, **product**, **product_-definition**, **product_definition_relationship**, **product_definition_shape**, or **property_definition** to which an action_request may be assigned.

EXPRESS specification:

```
*)
TYPE action_request_item = SELECT
     (action,
     executed_action,
     group,
     product,
     product_definition,
     product_definition_relationship,
     product_definition_shape,
     property_definition);
END_TYPE;
(*
```

5.2.2.1.2 approval_item

An **approval_item** identifies a **product_definition_shape**, **product_definition**, or **property_-definition** to which an approval may be assigned.

EXPRESS specification:

```
*)
TYPE approval_item = SELECT
     (product_definition_shape,
     product_definition,
     property_definition);
END_TYPE;
(*
```

5.2.2.1.3 classification_item

A **classification_item** identifies an **action**, **action_request_solution**, **compound_representation_-item**, **external_source**, **product_definition_shape**, **product_related_product_category**, **property_definition**, **group**, **product**, **product_definition**, **product_definition_with_associated_-documents**, **product_definition_relationship**, **property_definition_relationship**, **representation**, **shape_aspect**, **shape_aspect_relationship**, or **versioned_action_request** to which a classification may be assigned.

EXPRESS specification:

```
*)
TYPE classification_item = SELECT
     (action,
     action_request_solution,
     compound_representation_item,
     external_source,
     group,
     product,
     product_definition,
     product_definition_with_associated_documents,
     product_definition_relationship,
     product_definition_shape,
     product_related_product_category,
     property_definition,
     property_definition_relationship,
     representation,
     shape_aspect,
     shape_aspect_relationship,
     versioned_action_request);
END_TYPE;
(*
```

5.2.2.1.4 date_and_time_item

A **date_and_time_item** identifies an **action**, **action_request_solution**, **product_definition**, or **versioned_action_request** to which a date may be assigned.

EXPRESS specification:

```
*)
TYPE date_and_time_item = SELECT
     (action,
     action_request_solution,
     product_definition,
     versioned_action_request);
END_TYPE;
(*
```

5.2.2.1.5 document_reference_item

A **document_reference_item** identifies a **group** or **property_definition** to which a document may be assigned.

EXPRESS specification:

```
*)
TYPE document_reference_item = SELECT
     (group,
     property_definition);
END_TYPE;
(*
```

5.2.2.1.6 external_identification_item

An **external_identification_item** identifies a **document**, **document_reference**, **product**, **product_definition**, **product_definition_shape, property_definition,** or **shape_aspect** to which an external_identification may be assigned.

EXPRESS specification:

```
*)
TYPE external_identification_item = SELECT
     (document,
     document_reference,
     product,
     product_definition,
     product_definition_shape,
     property_definition,
     shape_aspect);
END_TYPE;
(*
```

5.2.2.1.7 group_item

A **group_item** identifies an **approval**, **product_definition**, **product_definition_shape**, **product_-definition_with_associated_documents**, **product_definition_relationship**, **representation**, or **shape_aspect** to which a group may be assigned.

EXPRESS specification:

```
*)
TYPE group_item = SELECT
     (approval,
     product_definition,
     product_definition_shape,
     product_definition_with_associated_documents,
     product_definition_relationship,
     representation,
     shape_aspect);
END_TYPE;
(*
```

5.2.2.1.8 identification_item

An **identification_item** identifies an **action_request_solution** , **executed_action**, **product_definition_shape**, **externally_defined_class**, **externally_defined_general_property**, **group**, **product**, **product_definition**, **product_definition_relationship**, **product_related_product_category**, **property_definition**, **property_definition_relationship**, **representation**, **shape_aspect**, **shape_aspect_relationship,** or **versioned_action_request** to which an identification may be assigned.

EXPRESS specification:

```
*)
TYPE identification_item = SELECT
```



```
     (action_request_solution,
     executed_action,
     externally_defined_general_property,
     externally_defined_class,
     group,
     product,
     product_definition,
     product_definition_shape,
     product_definition_relationship,
     product_related_product_category,
     property_definition_relationship,
     property_definition,
     representation,
     shape_aspect,
     shape_aspect_relationship,
     versioned_action_request);
END_TYPE;
(*
```

5.2.2.1.9 organization_item

An **organization_item** identifies an **externally_defined_class**, **document**, **known_source**, **product_definition**, or **property_definition** to which an organization may be identified.

EXPRESS specification:

```
*)
TYPE organization_item = SELECT
     (externally_defined_class,
     document,
     known_source,
     product_definition,
     property_definition);
END_TYPE;
(*
```

5.2.2.1.10 person_item

A **person_item** identifies an **action_request_solution**, **document**, **executed_action**, or **versioned_-action_request** to which a person may be identified.

EXPRESS specification:

```
*)
TYPE person_item = SELECT
     (action_request_solution,
     document,
     executed_action,
     versioned_action_request);
END_TYPE;
(*
```

5.2.2.1.11 person_and_organization_item

A **person_and_organization_item** identifies an **action_request_solution**, **executed_action**, **document**, or **versioned_action_request** to which a person_and_organization may be identified.

EXPRESS specification:

```
*)
TYPE person_and_organization_item = SELECT
     (action_request_solution,
     executed_action,
     document,
     versioned_action_request);
END_TYPE;
(*
```

5.2.3 Ship structures entities

5.2.3.1 Ship structures entity definitions

5.2.3.1.1 applied_action_request_assignment

An **applied_action_request_assignment** specifies those **action_request_item**s to which a referenced **action_request_assignment** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_action_request_assignment
  SUBTYPE OF (action_request_assignment);
  items : SET[1:?] OF action_request_item;
END_ENTITY;
(*
```

Attribute definition:

items: the set of **action_request_item** for which a particular **action_request_assignment** is applicable.

Associated global rules

The following global rule defined in this part of ISO 10303 applies to the **applied_action_request_-assignment** entity:

— change_impact_with_versionable_object_change_event (see 5.2.4.31).

5.2.3.1.2 applied_approval_assignment

An **applied_approval_assignment** specifies those **approval_item**s to which an **approval_-assignment** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_approval_assignment
    SUBTYPE OF (approval_assignment);
      items : SET [1:?] OF approval_item;
    WHERE
    WR1: NOT((SELF\approval_assignment.role.name = 'proposed alternative'))
         OR
         (SIZEOF(QUERY( app <*
         USEDIN (SELF\approval_assignment.assigned_approval,
         'SHIP_STRUCTURES_SCHEMA.APPROVAL_ASSIGNMENT.ASSIGNED_APPROVAL') |
         ('SHIP_STRUCTURES_SCHEMA.APPLIED_APPROVAL_ASSIGNMENT' IN TYPEOF
      (app)) AND
         (app\approval_assignment.role.name='subject')))=1);
END_ENTITY;
(*
```

Attribute definition:

items: the set of **approval_item** for which a particular **approval_assignment** is applicable.

Formal proposition:

WR1: Every **applied_approval_assignment** with a **role** attribute defining an **object_role** with attribute **name** equal to 'proposed alternative' is to reference an **approval** through the **assigned_approval** attribute. That **approval** entity is also to be referenced by another **applied_approval_assignment** through the **assigned_approval** attribute with a **role** attribute defining an **object_role** with a **name** of 'subject'.

Associated global rules:

The following global rules defined in this part of ISO 10303 applies to the **applied_approval_-assignment** entity:

— applied_approval_assignment_has_exactly_one_elements (see 5.2.4.8);

— approval_history_approves_same_definition (see 5.2.4.12).

5.2.3.1.3 applied_classification_assignment

An **applied_classification_assignment** specifies those **classification_item**s to which a **classification_assignment** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_classification_assignment
  SUBTYPE OF (classification_assignment);
  items : SET[1:?] OF classification_item;
END_ENTITY;
(*
```

Attribute definition:

items: the set of **classification_item** for which a particular **classification_assignment** is applicable.

Associated global rules:

The following global rules defined in this part of ISO 10303 applies to the **applied_classification_-assignment** entity:

— action_request_solution_with_identification_assignment (see 5.2.4.2);

— action_with_identification_assignment (see 5.2.4.3);

— assembly_relationship_has_product_definition (see 5.2.4.16);

— centre_location_compound_representation_has_specified_name (see 5.2.4.30);

— class_and_statutory_designation_has_properties (see 5.2.4.33);

— class_parameters_has_properties (see 5.2.4.35);

— corrugated_structure_design_definition_has_at_least_one_class (see 5.2.4.36);

— corrugated_structure_design_definition_has_at_most_one_class (see 5.2.4.37);

— define_maximum_permissible_still_water_bending_moment_values (see 5.2.4.40);

— definition_for_design_still_water_bending_moment_parameters (see 5.2.4.41);

— definition_for_design_vertical_wave_bending_moment_parameters (see 5.2.4.42);

— definition_for_design_vertical_wave_shear_force_parameters (see 5.2.4.43);

— define_maximum_permissible_still_water_shear_force_values (see 5.2.4.44);

— executed_action_with_identification_assignment (see 5.2.4.51);

— freeboard_characteristics_has_properties (see 5.2.4.54);

— global_axis_placement_has_properties (see 5.2.4.55);

— group_with_identification_assignment (see 5.2.4.57);

— homogeneous_ship_material_property_has_named_references (see 5.2.4.58);

— hull_cross_section_contains_specified_classes (see 5.2.4.59);

— hull_cross_section_design_has_at_least_one_revision_with_context (see 5.2.4.61);

— hull_cross_section_design_has_at_most_one_revision_with_context (see 5.2.4.62);

— hull_cross_section_has_at_least_one_revision_with_context (see 5.2.4.63);

— hull_cross_section_optionally_contains_specified_classes (see 5.2.4.64);

— hull_cross_section_result_for_representation_item (see 5.2.4.65);

— library_manufacturing_definition_for_class_references (see 5.2.4.68);

— lightship_definition_has_properties (see 5.2.4.69);

— maximum_representation_items_for_class_library_definition (see 5.2.4.71);

— panel_system_design_definition_has_at_most_one_class (see 5.2.4.75);

— panel_system_design_definition_has_at_least_one_class (see 5.2.4.74);

— panel_system_optionally_contains_specified_classes (see 5.2.4.76);

— plate_design_definition_has_at_least_one_class_references (see 5.2.4.77);

— plate_design_definition_has_at_most_one_class_references (see 5.2.4.78);

— plate_manufacturing_definition_has_class_references (see 5.2.4.79);

— principal_characteristics_has_properties (see 5.2.4.80);

— product_definition_for_shipyard (see 5.2.4.87);

— product_definition_shape_for_manufacturing_definition (see 5.2.4.114);

— product_definition_shape_for_source (see 5.2.4.127);

— product_definition_shape_for_library_design_parameters (see 5.2.4.112);

— product_definition_shape_for_name (see 5.2.4.115);

— product_definition_shape_for_library_definition_parameters (see 5.2.4.111);

— product_definition_shape_for_library_manufacturing_definition (see 5.2.4.113);

— product_definition_shape_with_versionable_object_pattern (see 5.2.4.140);

— product_definition_for_call_sign (see 5.2.4.81);

— product_definition_for_class_notation (see 5.2.4.82);

— product_definition_for_design_still_water_shear_force_values (see 5.2.4.83);

— product_definition_for_flag_state (see 5.2.4.84);

— product_definition_for_port_of_registration (see 5.2.4.85);

— product_definition_for_regulation (see 5.2.4.86);

— product_definition_has_references_with_class_loadline (see 5.2.4.88);

— product_definition_local_co_ordinate_system_with_station_reference (see 5.2.4.89);

— product_definition_relationship_with_definable_object (see 5.2.4.90);

— product_definition_relationship_with_identification_assignment (see 5.2.4.91);

— product_definition_shape_for_assembly_manufacturing (see 5.2.4.92);

— product_definition_shape_for_bevel_design_parameters (see 5.2.4.93);

— product_definition_shape_for_beveled_groove_weld_design (see 5.2.4.94);

— product_definition_shape_for_butt_groove_weld_design (see 5.2.4.95);

— product_definition_shape_for_circular_cutout_design_parameters (see 5.2.4.96);

— product_definition_shape_for_class_torch_vector (see 5.2.4.97);

— product_definition_shape_for_continuous_fillet_weld_design (see 5.2.4.98);

— product_definition_shape_for_corrugated_part_design (see 5.2.4.99);

— product_definition_shape_for_corrugated_structure_design (see 5.2.4.100);

— product_definition_shape_for_corrugation (see 5.2.4.101);

— product_definition_shape_for_drain_hole_cutout_design (see 5.2.4.102);

— product_definition_shape_for_elliptical_cutout_design (see 5.2.4.103);

— product_definition_shape_for_elongated_oval_cutout_design (see 5.2.4.104);

— product_definition_shape_for_fillet_weld_design_parameters (see 5.2.4.105);

— product_definition_shape_for_free_form_interior_cutout_design (see 5.2.4.106);

— product_definition_shape_for_groove_weld_design (see 5.2.4.107);

— product_definition_shape_for_intermittent_fillet_weld_design (see 5.2.4.109);

— product_definition_shape_for_inward_round_corner_design (see 5.2.4.110);

— product_definition_shape_for_outer_contour_representation (see 5.2.4.116);

— product_definition_shape_for_outward_round_corner_design (see 5.2.4.117);

— product_definition_shape_for_panel_system_design_parameters (see 5.2.4.118);

— product_definition_shape_for_part_edge_cutout_design (see 5.2.4.119);

— product_definition_shape_for_plate_design_parameters (see 5.2.4.120);

— product_definition_shape_for_profile_cross_section (see 5.2.4.121);

— product_definition_shape_for_profile_design_parameters (see 5.2.4.122);

— product_definition_shape_for_rectangular_cutout_design (see 5.2.4.123);

— product_definition_shape_for_round_corner_rectangular_cutout (see 5.2.4.124);

— product_definition_shape_for_round_edge_rectangular_cutout (see 5.2.4.125);

— product_definition_shape_for_shear_bevel_design_parameters (see 5.2.4.126);

— product_definition_shape_for_spot_seam_weld_design (see 5.2.4.128);

— product_definition_shape_for_structural_part_joint_design (see 5.2.4.129);

— product_definition_shape_for_structural_system_design (see 5.2.4.130);

— product_definition_shape_for_triangular_cutout_design (see 5.2.4.131);

— product_definition_shape_for_welded_joint_design_parameters (see 5.2.4.132);

— product_definition_shape_has_references_for_centre_of_gravity (see 5.2.4.133);

— product_definition_shape_has_references_with_class_border (see 5.2.4.135);

— product_definition_shape_of_rectangular_cutback_corner_design (see 5.2.4.136);

— product_definition_shape_with_definition_matches_pattern (see 5.2.4.137);

— product_definition_shape_with_identification_assignment (see 5.2.4.138);

— product_definition_shape_with_versionable_object (see 5.2.4.139);

— product_definition_with_date_freeboard_assigned (see 5.2.4.141);

— product_definition_with_definable_object_matches_pattern (see 5.2.4.142);

— product_definition_with_freeboard_assigned_by (see 5.2.4.143);

— product_definition_with_identification_assignment (see 5.2.4.144);

— product_related_product_category_with_identification_assignment (see 5.2.4.145);

— profile_cross_section_has_at_most_one_named_references (see 5.2.4.147);

— profile_design_definition_has_at_most_one_class_references (see 5.2.4.148);

— profile_design_definition_has_zero_or_two_class_references (see 5.2.4.149);

— property_definition_with_versionable_object_matches_pattern (see 5.2.4.172);

— property_definition_for_class_notation (see 5.2.4.150);

— property_definition_for_class_society (see 5.2.4.151);

— property_definition_for_corrugated_structure_function (see 5.2.4.152);

— property_definition_for_edge_cutout_function_parameters (see 5.2.4.153);

— property_definition_for_edge_feature_function_parameters (see 5.2.4.154);

— property_definition_for_homogeneous_ship_material_property (see 5.2.4.155);

— property_definition_for_interior_cutout_function_parameters (see 5.2.4.156);

— property_definition_for_local_co_ordinate_system (see 5.2.4.157);

— property_definition_for_local_co_ordinate_system_with_position (see 5.2.4.158);

— property_definition_for_panel_system_function_parameters (see 5.2.4.159);

— property_definition_for_plate_function_parameters (see 5.2.4.160);

— property_definition_for_plate_strake_function_parameters (see 5.2.4.161);

— property_definition_for_profile_function_parameters (see 5.2.4.162);

— property_definition_for_ship_material_property_parameters (see 5.2.4.163);

— property_definition_for_structural_system_function_parameters (see 5.2.4.164);

— property_definition_for_weld_filler_material_design (see 5.2.4.165);

— property_definition_has_references_with_name_loadline (see 5.2.4.166);

— property_definition_structural_weld_shrinkage_allowance_feature (see 5.2.4.167);

— property_definition_to_structural_added_material_feature_design (see 5.2.4.168);

— property_definition_with_definition_matches_pattern (see 5.2.4.169);

— property_definition_with_identification_assignment (see 5.2.4.170);

— property_definition_with_lightship_weight_item (see 5.2.4.171);

— property_definition_with_weight_and_centre_of_gravity (see 5.2.4.173);

— representation_has_references_with_name_moment_3d (see 5.2.4.216);

— representation_item_for_transformation_to_parent (see 5.2.4.220);

— representation_with_versionable_object_matches_pattern (see 5.2.4.312);

— representation_with_versionable_object_matches_pattern (see 5.2.4.312);

— seam_has_at_most_one_class_references (see 5.2.4.316);

— shape_aspect_for_angle_bar_cross_section_dimensions (see 5.2.4.317);

— shape_aspect_for_border_corner_cutout_boundary_relationship (see 5.2.4.319);

— shape_aspect_for_bulbflat_cross_section_dimensions (see 5.2.4.320);

— shape_aspect_for_channel_cross_section_dimensions (see 5.2.4.321);

— shape_aspect_for_circular_hollow_cross_section_dimensions (see 5.2.4.322);

— shape_aspect_for_composite_feature_design_parameters (see 5.2.4.323);

— shape_aspect_for_composite_feature_has_at_least_two_instances (see 5.2.4.324);

— shape_aspect_for_corner_cutout (see 5.2.4.325);

— shape_aspect_for_corner_cutout_boundary_relationship (see 5.2.4.326);

— shape_aspect_for_corrugation_dimensions (see 5.2.4.327);

— shape_aspect_for_explicit_profile_cross_section_dimensions (see 5.2.4.328);

— shape_aspect_for_flanged_plate_cross_section_dimensions (see 5.2.4.329);

— shape_aspect_for_flare_area_parameters (see 5.2.4.330);

— shape_aspect_for_flat_bar_cross_section_dimensions (see 5.2.4.331);

— shape_aspect_for_panel_system_curve_boundary_design (see 5.2.4.332);

— shape_aspect_for_panel_system_plane_boundary_design (see 5.2.4.333);

— shape_aspect_for_profile_curve_trace_line_design_parameters (see 5.2.4.334);

— shape_aspect_for_round_bar_cross_section_dimensions (see 5.2.4.335);

— shape_aspect_for_square_tube_cross_section_dimensions (see 5.2.4.336);

— shape_aspect_for_structural_cutout_boundary_relationship (see 5.2.4.337);

— shape_aspect_for_t_bar_cross_section_dimensions (see 5.2.4.338);

— shape_aspect_for_twist_location_design_parameters (see 5.2.4.339);

— shape_aspect_for_w_shape_cross_section_dimensions (see 5.2.4.340);

— shape_aspect_relationship_with_definable_object (see 5.2.4.342);

— shape_aspect_relationship_with_definition_matches_pattern (see 5.2.4.343);

— shape_aspect_relationship_with_identification_assignment (see 5.2.4.344);

— shape_aspect_with_definable_object_matches_pattern (see 5.2.4.345);

— shape_aspect_with_identification_assignment (see 5.2.4.346);

— ship_designation_has_one_specified_names (see 5.2.4.347);

— ship_material_property_has_at_most_one_named_references (see 5.2.4.348);

— source_for_library_definition (see 5.2.4.349);

— spacing_position_compound_representation_has_name (see 5.2.4.350);

— spacing_position_with_offset_compound_representation_has_class (see 5.2.4.351);

— spacing_position_with_offset_compound_representation_has_name (see 5.2.4.352);

— structural_class_rejection_has_at_least_one_role_references (see 5.2.4.353);

— structural_part_symmetry_relationship_has_product_definition (see 5.2.4.354);

— structural_system_symmetry_relationship_has_product_definition (see 5.2.4.355);

— user_defined_requires_user_defined_class (see 5.2.4.358);

— versioned_action_request_with_identification_assignment (see 5.2.4.370).

5.2.3.1.4 applied_date_and_time_assignment

An **applied_date_and_time_assignment** specifies those **date_and_time_item**s to which a **date_and_assignment** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_date_and_time_assignment
  SUBTYPE OF (date_and_time_assignment);
  items : SET[1:?] OF date_and_time_item;
END_ENTITY;
(*
```

Attribute definition:

items: the set of **date_and_time_item** for which a particular **date_and_time_assignment** is applicable.

Associated global rules:

The following global rules defined in this part of ISO 10303 applies to the **applied_date_and_time_-assignment** entity:

— caused_when_for_check (see 5.2.4.25);

— caused_when_for_envisaged_version_creation (see 5.2.4.26);

— caused_when_for_version_creation (see 5.2.4.27);

— caused_when_for_version_modification (see 5.2.4.29);

— caused_when_for_version_deletion (see 5.2.4.28);

— date_time_for_change_realisation (see 5.2.4.38):

— date_time_for_change_request (see 5.2.4.39);

— product_definition_with_date_freeboard_assigned (see 5.2.4.141).

5.2.3.1.5 applied_document_reference

An **applied_document_reference** specifies those **document_reference_item**s to which a **document_reference** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_document_reference
  SUBTYPE OF (document_reference);
  items : SET[1:?] OF document_reference_item;
END_ENTITY;
(*
```


Attribute definition:

items: the set of **document_reference_item** for which a particular **document_reference** is applicable.

Associated global rules:

The following global rules defined in this part of ISO 10303 applies to the **applied_document_reference** entity:

— ship_material_property_has_at_most_one_named_references (see 5.2.4.348);

— structural_class_rejection_has_at_least_one_role_references (see 5.2.4.353).

5.2.3.1.6 applied_external_identification_assignment

An **applied_external_identification_assignment** specifies those **external_identification_item**s to which an **external_identification_assignment** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_external_identification_assignment
  SUBTYPE OF (external_identification_assignment);
  items : SET[1:?] OF external_identification_item;
END_ENTITY;
(*
```

Attribute definition:

items: the set of **external_identification_item** for which a particular **external_identification_-assignment** is applicable.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **applied_external_-identification_assignment** entity:

— document_reference_with_address_has_at_least_one_references (see 5.2.4.49).

5.2.3.1.7 applied_group_assignment

An **applied_group_assignment** specifies those **group_item**s to which a **group_assignment** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_group_assignment
  SUBTYPE OF (group_assignment);
  items : SET[1:?] OF group_item;
```



```
END_ENTITY;
(*
```

Attribute definition:

items: the set of **group_item** for which a particular **group_assignment** is applicable.

Associated global rules:

The following global rules defined in this part of ISO 10303 applies to the **applied_group_-assignment** entity:

— applied_group_assignment_has_at_least_one_elements (see 5.2.4.9);

— approval_history_approves_same_definition (see 5.2.4.12);

— approval_history_contains_approval_elements (see 5.2.4.13);

— approval_history_has_at_least_one_member (see 5.2.4.14);

— approvals_references_approval_history (see 5.2.4.15);

— hull_cross_section_contains_specified_classes (see 5.2.4.59);

— hull_cross_section_optionally_contains_specified_classes (see 5.2.4.64);

— members_is_referenced_by_at_least_one_revision (see 5.2.4.72);

— panel_system_optionally_contains_specified_classes (see 5.2.4.76);

— revision_with_context_referenced_for_context_of_revision (see 5.2.4.315);

— shape_aspect_for_border_corner_cutout_boundary_relationship (see 5.2.4.319);

— shape_aspect_for_corner_cutout (see 5.2.4.325);

— shape_aspect_for_corner_cutout_boundary_relationship (see 5.2.4.326);

— shape_aspect_for_structural_cutout_boundary_relationship (see 5.2.4.337);

— unique_approvals_in_approval_history (see 5.2.4.356);

— version_history_has_exactly_one_assigned_group (see 5.2.4.362);

— version_history_referenced_by_multiple_roles (see 5.2.4.364);

— version_history_referenced_by_exactly_one_current_version (see 5.2.4.363);

— versions_is_referenced_by_at_least_one_version_history (see 5.2.4.371).

5.2.3.1.8 applied_identification_assignment

An **applied_identification_assignment** specifies those **identification_item**s to which an **identification_assignment** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_identification_assignment
  SUBTYPE OF (identification_assignment);
  items : SET[1:?] OF identification_item;
END_ENTITY;
(*
```

Attribute definition:

items: the set of **identification_item** for which a particular **identification_assignment** is applicable.

Associated global rules:

The following global rules defined in this part of ISO 10303 applies to the **applied_identification_-assignment** entity:

— alternative_version_relationship_versionable_object (see 5.2.4.4);

— global_id_is_unique (see 5.2.4.56);

— product_definition_for_call_sign (see 5.2.4.81);

— product_definition_for_flag_state (see 5.2.4.84);

— product_definition_for_port_of_registration (see 5.2.4.85);

— product_definition_shape_has_references_for_weld_design (see 5.2.4.134);

— product_with_identification_assignment (see 5.2.4.146);

— ship_designation_has_one_specified_names (see 5.2.4.347);

— version_relationship_associates_with_versionable_object (see 5.2.4.366);

— versionable_object_has_one_version_id (see 5.2.4.369).

5.2.3.1.9 applied_organization_assignment

An **applied_organization_assignment** specifies those **organization_item**s to which an **organization_assignment** is assigned.

EXPRESS specification:


```
*)
ENTITY applied_organization_assignment
  SUBTYPE OF (organization_assignment);
  items : SET[1:?] OF organization_item;
END_ENTITY;
(*
```

Attribute definition:

items: the set of **organization_item** for which a particular **organization_assignment** is applicable.

Associated global rules:

The following global rules defined in this part of ISO 10303 applies to the **applied_organization_-assignment** entity:

— product_definition_for_shipyard (see 5.2.4.87);

— product_definition_with_freeboard_assigned_by (see 5.2.4.143);

— property_definition_for_class_society (see 5.2.4.151).

5.2.3.1.10 applied_person_assignment

An **applied_person_assignment** specifies those **person_item**s to which a **person_assignment** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_person_assignment
  SUBTYPE OF (person_assignment);
  items : SET[1:?] OF person_item;
END_ENTITY;
(*
```

Attribute definition:

items: the set of **person_item** for which a particular **person_assignment** is applicable.

5.2.3.1.11 applied_person_and_organization_assignment

An **applied_person_and_organization_assignment** specifies those **person_and_organization_-item**s to which a **person_and_organization_assignment** is assigned.

EXPRESS specification:

```
*)
ENTITY applied_person_and_organization_assignment
  SUBTYPE OF (person_and_organization_assignment);
  items : SET[1:?] OF person_and_organization_item;
END_ENTITY;
```

```
(*
```

Attribute definition:

items: the set of **person_and_organization_item** for which a particular **person_and_organization_assignment** is applicable.

Associated global rules:

The following global rules defined in this part of ISO 10303 applies to the **applied_-person_and_organization_assignment** entity:

— author_for_change_plan (see 5.2.4.17);

— author_for_change_realisation (see 5.2.4.18):

— author_for_change_request (see 5.2.4.19);

— caused_by_for_check (see 5.2.4.20);

— caused_by_for_envisaged_version_creation (see 5.2.4.21);

— caused_by_for_version_creation (see 5.2.4.22);

— caused_by_for_version_deletion (see 5.2.4.23);

— caused_by_for_version_modification (see 5.2.4.24);

— initiator_for_change_request (see 5.2.4.67).

5.2.3.1.12 class

A **class** is a type of **group** that specifies a type of classification assignment.

EXPRESS specification:

```
*)
ENTITY class
    SUBTYPE OF (group);
    WHERE
    WR1: (SIZEOF(QUERY ( oa <* USEDIN(SELF,
          'SHIP_STRUCTURES_SCHEMA.GROUP_ASSIGNMENT.ASSIGNED_GROUP')|
          NOT ('SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT' IN
         TYPEOF(oa))
        )) =0);
END_ENTITY; -- class
(*
```


Formal proposition:

WR1: Each **class** is not to be referenced by any **applied_group_assignment** through the **assigned_group** attribute.

5.2.3.1.13 externally_defined_class

An **externally_defined_class** is a type of **externally_defined_item** and a type of **class** that specifies a type of classification assignment with external reference.

EXPRESS specification:

```
*)
ENTITY externally_defined_class
  SUBTYPE OF (class, externally_defined_item);
END_ENTITY;
(*
```

Informal proposition:

IP1: The value of **externally_defined_class** attribute **item_id** (inherited from externally_defined_item) is instantiated in accordance to the class code of ISO 13584-42.

Associated global rules:

5.2.3.1.14 externally_defined_general_property

An **externally_defined_general_property** is a type of **general_property** and a type of **externally_-defined_item** that specifies a type of general property with an external reference.

EXPRESS specification:

```
*)
ENTITY externally_defined_general_property
  SUBTYPE OF (general_property, externally_defined_item);
END_ENTITY;
(*
```

5.2.3.1.15 library_class_version_assignment

A **library_class_version_assignment** is a type of **applied_external_identification_assignment** that specifies a type of external identification.

EXPRESS specification:

```
*)
  ENTITY library_class_version_assignment
    SUBTYPE OF (applied_external_identification_assignment);
    WHERE
  WR1: SELF\identification_assignment.role.name = 'class version';
  WR2: SIZEOF (SELF\applied_external_identification_assignment.items) >1;
  WR3: ('SHIP_STRUCTURES_SCHEMA.KNOWN_SOURCE'
        IN TYPEOF (SELF\external_identification_assignment.source));
  END_ENTITY; -- library_class_version_assignment
(*
```

Formal propositions:

WR1: Each **library_class_version_assignment** is to have a **role** attribute that defines an **identification_role** entity that has a **name** attribute with a value of 'property version'.

WR2: Each **library_class_version_assignment** is to have an **items** attribute that is to have only one entity in its list which is a **known_source**.

WR3: Each **library_class_version_assignment** is to have a **source** attribute that defines a **known_-source** entity.

Informal propositions:

IP1: The value of **library_class_version_assignment** attribute **assigned_id** (inherited from **identification_assignment**) is instantiated in accordance to the version code of ISO 13584-42.

5.2.3.1.16 library_property_version_assignment

A **library_property_version_assignment** is a type of **applied_external_identification_assignment** that specifies a type of external identification.

EXPRESS specification:

```
*)
  ENTITY library_applied_version_assignment
    SUBTYPE OF (applied_external_identification_assignment);
   WHERE
  WR1: SELF\identification_assignment.role.name = 'property version';
  WR2: SIZEOF (SELF\applied_external_identification_assignment.items) =1;
  WR3: ('SHIP_STRUCTURES_SCHEMA.KNOWN_SOURCE'
        IN TYPEOF (SELF\external_identification_assignment.source));
  END_ENTITY; -- library_applied_version_assignment
(*
```


Formal propositions:

WR1: Each **library_applied_version_assignment** is to have a **role** attribute that defines an **identification_role** entity that has a **name** attribute with a value of 'property version'.

WR2: Each **library_applied_version_assignment** is to have an **items** attribute that is to have only one entity in its list which is a **known_source**.

WR3: Each **library_applied_version_assignment** is to have a **source** attribute that defines a **known_source** entity.

5.2.3.1.17 known_source

A **known_source** is a type of **external_source** and **pre_defined_item**, and it is a source of information whose name and content are pre-determined in the application protocol.

EXPRESS specification:

```
*)
ENTITY known_source
   SUBTYPE OF (pre_defined_item, external_source);
   WHERE
     WR1:  SELF\pre_defined_item.name = 'ISO 13584 library';
     WR2: (SIZEOF(QUERY ( oa <* USEDIN(SELF,
          'SHIP_STRUCTURES_SCHEMA.APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS')|
           NOT ('SHIP_STRUCTURES_SCHEMA.ORGANIZATION_ASSIGNMENT' IN
           TYPEOF(oa))
           AND (oa.role.name='library supplier'
           ))) =0);
END_ENTITY; -- known_source
(*
```

Formal propositions:

WR1: Each **known_source** is to have a **name** attribute with the value of 'ISO 13584 library';

WR2: Each **known_source** is referenced by at least one through the **items** attribute of the entity **applied_organization_assignment** that references through the **role** attribute an **organization_role** with a **name** attribute of value 'library supplier'.

Informal propositions:

IP1: The value of the **id** attribute of **organization** is instantiated in accordance to the supplier code of ISO 13584-26.

5.2.3.1.18 pre_defined_item

A **pre_defined_item** is used to convey information about the type of source and the information that it contains. The name is standardised by an application protocol so that implementations can use this value as a trigger to employ a PLIB compliant interface. The **known_source** construct is created and used in each AP that requires it.

EXPRESS specification:

```
*)
 ENTITY pre_defined_item;
     name          : label;
  END_ENTITY;  -- pre_defined_item
(*
```

5.2.3.2 Ship structures imported entity modification

5.2.3.2.1 action

The base definition of the **action** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **action** entity:

— caused_by_for_check (see 5.2.4.20);

— caused_by_for_envisaged_version_creation (see 5.2.4.21);

— caused_by_for_version_creation (see 5.2.4.22);

— caused_by_for_version_deletion (see 5.2.4.23);

— caused_by_for_version_modification (see 5.2.4.24);

— caused_when_for_check (see 5.2.4.25);

— caused_when_for_envisaged_version_creation (see 5.2.4.26);

— caused_when_for_version_creation (see 5.2.4.27);

— caused_when_for_version_deletion (see 5.2.4.28);

— caused_when_for_version_modification (see 5.2.4.29);

— change_impact_with_versionable_object_change_event (see 5.2.4.31);

— envisaged_version_creation_has_mandatory_attribute_description (see 5.2.4.50);

— version_creation_has_mandatory_attribute_description (see 5.2.4.360);

— version_deletion_has_mandatory_attribute_description (see 5.2.4.361);

— version_modification_has_mandatory_attribute_description (see 5.2.4.365).

5.2.3.2.2 action_request_solution

The base definition of the **action_request_solution** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **action_request_solution** entity:

— author_for_change_plan (see 5.2.4.17);

— change_plan_has_mandatory_attribute_description (see 5.2.4.32).

5.2.3.2.3 approval

The base definition of the **approval** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **action_request_solution** entity:

— approval_event_with_approval_date_time (see 5.2.4.10);

— approval_event_with_approval_person_organization (see 5.2.4.11);

— no_approvals_except_in_approval_history (see 5.2.4.73).

5.2.3.2.4 approval_date_time

The base definition of the **approval_date_time** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **approval_date_time** entity:

— approval_event_with_approval_date_time (see 5.2.4.10).

5.2.3.2.5 approval_person_organization

The base definition of the **approval_person_organization** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **approval_person_organization** entity:

— approval_event_with_approval_person_organization (see 5.2.4.11).

5.2.3.2.6 classification_role

The base definition of the **classification_role** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **classification_role** entity:

— maximum_representation_items_for_class_library_definition (see 5.2.4.71).

5.2.3.2.7 compound_representation_item

The base definition of the **compound_representation_item** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **compound_representation_item** entity:

— centre_location_compound_representation_has_specified_name (see 5.2.4.30);

— representation_items_for_maximum_permissible_swbm_values (see 5.2.4.230);

— representation_items_for_maximum_permissible_swsf_values (see 5.2.4.231);

— representation_items_specified_for_design_swsf_values (see 5.2.4.260);

— spacing_position_compound_representation_has_name (see 5.2.4.350);

— spacing_position_with_offset_compound_representation_has_class (see 5.2.4.351);

— spacing_position_with_offset_compound_representation_has_name (see 5.2.4.352).

5.2.3.2.8 date_time_role

The base definition of the **date_time_role** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **date_time_role** entity:

— caused_when_for_check (see 5.2.4.25);

— caused_when_for_envisaged_version_creation (see 5.2.4.26);

— caused_when_for_version_creation (see 5.2.4.27);

— caused_when_for_version_deletion (see 5.2.4.28);

— caused_when_for_version_modification (see 5.2.4.29);

— date_time_for_change_realisation (see 5.2.4.38);

— date_time_for_change_request (see 5.2.4.39).

5.2.3.2.9 document

The base definition of the **document** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **document** entity:

— document_has_at_least_one_references (see 5.2.4.45);

— document_has_exactly_one_author (see 5.2.4.46);

— document_reference_has_at_least_one_references (see 5.2.4.47);

— document_reference_reference_with_address_with_references (see 5.2.4.48);

— document_reference_with_address_has_at_least_one_references (see 5.2.4.49).

5.2.3.2.10 document_representation_type

The base definition of the **document_representation_type** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **document_-representation_type** entity:

— document_has_at_least_one_references (see 5.2.4.45).

5.2.3.2.11 document_usage_constraint

The base definition of the **document_usage_constraint** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **document_usage_-constraint** entity:

— document_reference_has_at_least_one_references (see 5.2.4.47);

— document_reference_reference_with_address_with_references (see 5.2.4.48).

5.2.3.2.12 executed_action

The base definition of the **executed_action** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **executed_action** entity:

— author_for_change_realisation (see 5.2.4.18);

— date_time_for_change_realisation (see 5.2.4.38).

5.2.3.2.13 externally_defined_class

The base definition of the **externally_defined_class** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **externally_defined_class** entity:

— externally_defined_class_with_known_source_requirement (see 5.2.4.52).

5.2.3.2.14 executed_action

The base definition of the **executed_action** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **executed_action** entity:

— mandatory_entity_type_for_external_instance_reference (see 5.2.4.70).

5.2.3.2.15 external_source_relationship

The base definition of the **external_source_relationship** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **external_source_relationship** entity:

— mandatory_entity_type_for_external_instance_reference (see 5.2.4.70).

5.2.3.2.16 group

The base definition of the **group** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **group** entity:

— approval_history_has_at_least_one_member (see 5.2.4.14);

— assembly_relationship_has_product_definition (see 5.2.4.16);

— class_and_statutory_designation_has_properties (see 5.2.4.33);

— class_parameters_has_properties (see 5.2.4.35);

— corrugated_structure_design_definition_has_at_least_one_class (see 5.2.4.36);

— corrugated_structure_design_definition_has_at_most_one_class (see 5.2.4.37);

— global_axis_placement_has_properties (see 5.2.4.55);

— homogeneous_ship_material_property_has_named_references (see 5.2.4.58);

— hull_cross_section_contains_specified_classes (see 5.2.4.59);

— hull_cross_section_design_has_at_least_one_revision_with_context (see 5.2.4.61);

— hull_cross_section_design_has_at_most_one_revision_with_context (see 5.2.4.62);

— hull_cross_section_has_at_least_one_revision_with_context (see 5.2.4.63);

— hull_cross_section_optionally_contains_specified_classes (see 5.2.4.64);

— hull_cross_section_result_for_representation_item (see 5.2.4.65);

— maximum_representation_items_for_class_library_definition (see 5.2.4.71);

— members_is_referenced_by_at_least_one_revision (see 5.2.4.72);

— panel_system_design_definition_has_at_most_one_class (see 5.2.4.75);

— panel_system_design_definition_has_at_least_one_class(see 5.2.4.74);

— panel_system_optionally_contains_specified_classes (see 5.2.4.76);

— plate_design_definition_has_at_least_one_class_references (see 5.2.4.77);

— plate_design_definition_has_at_most_one_class_references (see 5.2.4.78);

— plate_manufacturing_definition_has_class_references (see 5.2.4.79);

— principal_characteristics_has_properties (see 5.2.4.80);

— product_definition_for_shipyard (see 5.2.4.87);

— product_definition_shape_for_manufacturing_definition (see 5.2.4.114);

— product_definition_shape_for_source (see 5.2.4.127);

— product_definition_shape_for_library_design_parameters (see 5.2.4.112);

— product_definition_shape_for_name (see 5.2.4.115);

— product_definition_shape_for_library_manufacturing_definition (see 5.2.4.113);

— product_definition_shape_for_library_definition_parameters (see 5.2.4.111);

— product_definition_shape_with_versionable_object_pattern (see 5.2.4.140);

— product_definition_for_call_sign (see 5.2.4.81);

— product_definition_for_class_notation (see 5.2.4.82);

— product_definition_for_flag_state (see 5.2.4.84);

— product_definition_for_port_of_registration (see 5.2.4.85);

— product_definition_for_regulation (see 5.2.4.86);

— product_definition_local_co_ordinate_system_with_station_reference (see 5.2.4.89);

— product_definition_relationship_with_definable_object (see 5.2.4.90);

— product_definition_shape_for_assembly_manufacturing (see 5.2.4.92);

— product_definition_shape_for_beveled_groove_weld_design (see 5.2.4.94);

— product_definition_shape_for_butt_groove_weld_design (see 5.2.4.95);

— product_definition_shape_for_class_torch_vector (see 5.2.4.97);

— product_definition_shape_for_continuous_fillet_weld_design (see 5.2.4.98);

— product_definition_shape_for_corrugated_part_design (see 5.2.4.99);

— product_definition_shape_for_corrugated_structure_design (see 5.2.4.100);

— product_definition_shape_for_corrugation (see 5.2.4.101);

— product_definition_shape_for_fillet_weld_design_parameters (see 5.2.4.105);

— product_definition_shape_for_groove_weld_design (see 5.2.4.107);

— product_definition_shape_for_intermittent_fillet_weld_design (see 5.2.4.109);

— product_definition_shape_for_outer_contour_representation (see 5.2.4.116);

— product_definition_shape_for_panel_system_design_parameters (see 5.2.4.118);

— product_definition_shape_for_plate_design_parameters (see 5.2.4.120);

— product_definition_shape_for_profile_cross_section (see 5.2.4.121);

— product_definition_shape_for_profile_design_parameters (see 5.2.4.122);

— product_definition_shape_for_spot_seam_weld_design (see 5.2.4.128);

— product_definition_shape_for_structural_part_joint_design (see 5.2.4.129);

— product_definition_shape_for_structural_system_design (see 5.2.4.130);

— product_definition_shape_for_welded_joint_design_parameters (see 5.2.4.132);

— product_definition_shape_has_references_for_centre_of_gravity (see 5.2.4.133);

— product_definition_shape_has_references_with_class_border (see 5.2.4.135);

— product_definition_shape_with_definition_matches_pattern (see 5.2.4.137);

— product_definition_shape_with_versionable_object (see 5.2.4.139);

— product_definition_with_definable_object_matches_pattern (see 5.2.4.142);

— profile_cross_section_has_at_most_one_named_references (see 5.2.4.147);

— profile_design_definition_has_at_most_one_class_references (see 5.2.4.148);

— profile_design_definition_has_zero_or_two_class_references (see 5.2.4.149);

— property_definition_with_versionable_object_matches_pattern (see 5.2.4.172);

— property_definition_for_class_notation (see 5.2.4.150);

— property_definition_for_class_society (see 5.2.4.151);

— property_definition_for_corrugated_structure_function (see 5.2.4.152);

— property_definition_for_homogeneous_ship_material_property (see 5.2.4.155);

— property_definition_for_local_co_ordinate_system_with_position (see 5.2.4.158);

— property_definition_for_panel_system_function_parameters (see 5.2.4.159);

— property_definition_for_plate_function_parameters (see 5.2.4.160);

— property_definition_for_plate_strake_function_parameters (see 5.2.4.161);

— property_definition_for_profile_function_parameters (see 5.2.4.162);

— property_definition_for_ship_material_property_parameters (see 5.2.4.163);

— property_definition_with_definition_matches_pattern (see 5.2.4.169);

— representation_item_for_transformation_to_parent (see 5.2.4.220);

— representation_with_versionable_object_matches_pattern (see 5.2.4.312);

— revision_has_mandatory_attribute_description (see 5.2.4.314);

— revision_with_context_referenced_for_context_of_revision (see 5.2.4.315);

— seam_has_at_most_one_class_references (see 5.2.4.316);

— shape_aspect_for_angle_bar_cross_section_dimensions (see 5.2.4.317);

— shape_aspect_for_assembly_manufacturing_position_parameters (see 5.2.4.318);

— shape_aspect_for_border_corner_cutout_boundary_relationship (see 5.2.4.319);

— shape_aspect_for_bulbflat_cross_section_dimensions (see 5.2.4.320);

— shape_aspect_for_channel_cross_section_dimensions (see 5.2.4.321);

— shape_aspect_for_circular_hollow_cross_section_dimensions (see 5.2.4.322);

— shape_aspect_for_composite_feature_design_parameters (see 5.2.4.323);

— shape_aspect_for_corner_cutout (see 5.2.4.325);

— shape_aspect_for_corner_cutout_boundary_relationship (see 5.2.4.326);

— shape_aspect_for_corrugation_dimensions (see 5.2.4.327);

— shape_aspect_for_explicit_profile_cross_section_dimensions (see 5.2.4.328);

— shape_aspect_for_flanged_plate_cross_section_dimensions (see 5.2.4.329);

— shape_aspect_for_flare_area_parameters (see 5.2.4.330);

— shape_aspect_for_flat_bar_cross_section_dimensions (see 5.2.4.331);

— shape_aspect_for_panel_system_curve_boundary_design (see 5.2.4.332);

— shape_aspect_for_panel_system_plane_boundary_design (see 5.2.4.333);

— shape_aspect_for_profile_curve_trace_line_design_parameters (see 5.2.4.334);

— shape_aspect_for_round_bar_cross_section_dimensions (see 5.2.4.335);

— shape_aspect_for_square_tube_cross_section_dimensions (see 5.2.4.336);

— shape_aspect_for_structural_cutout_boundary_relationship (see 5.2.4.337);

— shape_aspect_for_t_bar_cross_section_dimensions (see 5.2.4.338);

— shape_aspect_for_twist_location_design_parameters (see 5.2.4.339);

— shape_aspect_for_w_shape_cross_section_dimensions (see 5.2.4.340);

— shape_aspect_relationship_with_definable_object (see 5.2.4.342);

— shape_aspect_relationship_with_definition_matches_pattern (see 5.2.4.343);

— shape_aspect_with_definable_object_matches_pattern (see 5.2.4.345);

— ship_designation_has_one_specified_names (see 5.2.4.347);

— source_for_library_definition (see 5.2.4.349);

— spacing_position_compound_representation_has_name (see 5.2.4.350);

— spacing_position_with_offset_compound_representation_has_class (see 5.2.4.351);

— spacing_position_with_offset_compound_representation_has_name (see 5.2.4.352);

— structural_class_rejection_has_at_least_one_role_references (see 5.2.4.353);

— structural_part_symmetry_relationship_has_product_definition (see 5.2.4.354);

— structural_system_symmetry_relationship_has_product_definition (see 5.2.4.355);

— unique_approvals_in_approval_history (see 5.2.4.356);

— user_defined_requires_user_defined_class (see 5.2.4.358);

— version_history_has_exactly_one_assigned_group (see 5.2.4.362);

— version_history_referenced_by_exactly_one_current_version (see 5.2.4.363);

— version_history_referenced_by_multiple_roles (see 5.2.4.364);

— versions_is_referenced_by_at_least_one_version_history (see 5.2.4.371).

5.2.3.2.17 identification_assignment_relationship

The base definition of the **identification_assignment_relationship** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **identification_-assignment_relationship** entity:

— alternative_version_relationship_has_mandatory_description (see 5.2.4.5);

— alternative_version_relationship_has_unique_versions (see 5.2.4.6);

— version_relationship_has_mandatory_attribute_description (see 5.2.4.367);

— version_relationship_has_unique_versions (see 5.2.4.368).

5.2.3.2.18 identification_role

The base definition of the **identification_role** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **identification_role** entity:

— global_id_is_unique (see 5.2.4.56);

— identification_role_optional_attribute_description_required (see 5.2.4.66).

5.2.3.2.19 known_source

The base definition of the **known_source** entity is given in ISO 10303-43. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **known_source** entity:

— restrict_name_for_known_source (see 5.2.4.313).

5.2.3.2.20 mapped_item

The base definition of the **mapped_item** entity is given in ISO 10303-43. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **mapped_item** entity:

— representation_item_for_transformation_to_parent (see 5.2.4.220).

5.2.3.2.21 object_role

The base definition of the **object_role** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **object_role** entity:

— applied_approval_assignment_has_exactly_one_elements (see 5.2.4.8);

— applied_group_assignment_has_at_least_one_elements (see 5.2.4.9);

— approval_history_contains_approval_elements (see 5.2.4.13);

— homogeneous_ship_material_property_has_named_references (see 5.2.4.58);

— hull_cross_section_has_at_least_one_revision_with_context (see 5.2.4.63).

5.2.3.2.22 person_and_organization_role

The base definition of the **person_and_organization_role** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **person_and_organization_role** entity:

— author_for_change_plan (see 5.2.4.17);

— author_for_change_realisation (see 5.2.4.18):

— author_for_change_request (see 5.2.4.19);

— caused_by_for_check (see 5.2.4.20);

— caused_by_for_envisaged_version_creation (see 5.2.4.21);

— caused_by_for_version_creation (see 5.2.4.22);

— caused_by_for_version_deletion (see 5.2.4.23);

— caused_by_for_version_modification (see 5.2.4.24);

— initiator_for_change_request (see 5.2.4.67).

5.2.3.2.23 product_definition

The base definition of the **product_definition** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **product_definition** entity:

— class_and_statutory_designation_has_properties (see 5.2.4.33);

— class_parameters_has_properties (see 5.2.4.35);

— define_maximum_permissible_still_water_bending_moment_values (see 5.2.4.40);

— definition_for_design_still_water_bending_moment_parameters (see 5.2.4.41);

— definition_for_design_vertical_wave_bending_moment_parameters (see 5.2.4.42);

— definition_for_design_vertical_wave_shear_force_parameters (see 5.2.4.43);

— define_maximum_permissible_still_water_shear_force_values (see 5.2.4.44);

— freeboard_characteristics_has_properties (see 5.2.4.54);

— lightship_definition_has_properties (see 5.2.4.69);

— product_definition_for_call_sign (see 5.2.4.81);

— product_definition_for_design_still_water_shear_force_values (see 5.2.4.83);

— product_definition_for_flag_state (see 5.2.4.84);

— product_definition_for_port_of_registration (see 5.2.4.85);

— product_definition_for_regulation (see 5.2.4.86);

— product_definition_for_shipyard (see 5.2.4.87);

— product_definition_has_references_with_class_loadline (see 5.2.4.88);

— product_definition_local_co_ordinate_system_with_station_reference (see 5.2.4.89);

— product_definition_shape_for_hull_cross_section_design_definition (see 5.2.4.108);

— product_definition_with_date_freeboard_assigned (see 5.2.4.141);

— product_definition_with_definable_object_matches_pattern (see 5.2.4.142);

— product_definition_with_definable_object_matches_pattern (see 5.2.4.142);

— product_definition_with_freeboard_assigned_by (see 5.2.4.143);

— ship_designation_has_one_specified_names (see 5.2.4.347);

— user_defined_requires_user_defined_class (see 5.2.4.358).

5.2.3.2.24 product_definition_relationship

The base definition of the **product_definition_relationship** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rule defined in this part of ISO 10303 applies to the **product_definition_-relationship** entity:

— product_definition_relationship_with_definable_object (see 5.2.4.90).

5.2.3.2.25 product_definition_shape

The base definition of the **product_definition_shape** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **product_definition_shape** entity:

— corrugated_structure_design_definition_has_at_least_one_class (see 5.2.4.36);

— corrugated_structure_design_definition_has_at_most_one_class (see 5.2.4.37);

— product_definition_shape_for_manufacturing_definition (see 5.2.4.114);

— product_definition_shape_for_source (see 5.2.4.127);

— product_definition_shape_for_library_design_parameters (see 5.2.4.112);

— product_definition_shape_for_name (see 5.2.4.115);

— product_definition_shape_for_library_manufacturing_definition (see 5.2.4.113);

— product_definition_shape_for_library_definition_parameters (see 5.2.4.111);

— product_definition_relationship_with_definable_object (see 5.2.4.90);

— product_definition_shape_for_bevel_design_parameters (see 5.2.4.93);

— product_definition_shape_for_beveled_groove_weld_design (see 5.2.4.94);

— product_definition_shape_for_butt_groove_weld_design (see 5.2.4.95);

— product_definition_shape_for_circular_cutout_design_parameters (see 5.2.4.96);

— product_definition_shape_for_class_torch_vector (see 5.2.4.97);

— product_definition_shape_for_continuous_fillet_weld_design (see 5.2.4.98);

— product_definition_shape_for_corrugated_part_design (see 5.2.4.99);

— product_definition_shape_for_corrugated_structure_design (see 5.2.4.100);

— product_definition_shape_for_corrugation (see 5.2.4.101);

— product_definition_shape_for_drain_hole_cutout_design (see 5.2.4.102);

— product_definition_shape_for_elliptical_cutout_design (see 5.2.4.103);

— product_definition_shape_for_elongated_oval_cutout_design (see 5.2.4.104);

— product_definition_shape_for_fillet_weld_design_parameters (see 5.2.4.105);

— product_definition_shape_for_free_form_interior_cutout_design (see 5.2.4.106);

— product_definition_shape_for_groove_weld_design (see 5.2.4.107);

— product_definition_shape_for_intermittent_fillet_weld_design (see 5.2.4.109);

— product_definition_shape_for_inward_round_corner_design (see 5.2.4.110);

— product_definition_shape_for_outer_contour_representation (see 5.2.4.116);

— product_definition_shape_for_outward_round_corner_design (see 5.2.4.117);

— product_definition_shape_for_panel_system_design_parameters (see 5.2.4.118);

— product_definition_shape_for_part_edge_cutout_design (see 5.2.4.119);

— product_definition_shape_for_plate_design_parameters (see 5.2.4.120);

— product_definition_shape_for_profile_cross_section (see 5.2.4.121);

— product_definition_shape_for_profile_design_parameters (see 5.2.4.122);

— product_definition_shape_for_rectangular_cutout_design (see 5.2.4.123);

— product_definition_shape_for_round_corner_rectangular_cutout (see 5.2.4.124);

— product_definition_shape_for_round_edge_rectangular_cutout (see 5.2.4.125);

— product_definition_shape_for_shear_bevel_design_parameters (see 5.2.4.126);

— product_definition_shape_for_spot_seam_weld_design (see 5.2.4.128);

— product_definition_shape_for_structural_part_joint_design (see 5.2.4.129);

— product_definition_shape_for_structural_system_design (see 5.2.4.130);

— product_definition_shape_for_triangular_cutout_design (see 5.2.4.131);

— product_definition_shape_for_welded_joint_design_parameters (see 5.2.4.132);

— product_definition_shape_has_references_for_centre_of_gravity (see 5.2.4.133);

— product_definition_shape_has_references_for_weld_design (see 5.2.4.134);

— product_definition_shape_has_references_with_class_border (see 5.2.4.135);

— product_definition_shape_of_rectangular_cutback_corner_design (see 5.2.4.136);

— profile_design_definition_has_at_most_one_class_references (see 5.2.4.148);

— profile_design_definition_has_zero_or_two_class_references (see 5.2.4.149);

— representation_restricted_for_groove_weld_design (see 5.2.4.290).

5.2.3.2.26 property_definition

The base definition of the **property_definition** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **property_definition** entity:

— freeboard_characteristics_has_properties (see 5.2.4.54);

— hull_cross_section_has_at_least_one_revision_with_context (see 5.2.4.63);

— lightship_definition_has_properties (see 5.2.4.69);

— product_definition_for_class_notation (see 5.2.4.82);

— product_definition_for_regulation (see 5.2.4.86);

— product_definition_has_references_with_class_loadline (see 5.2.4.88);

— product_definition_local_co_ordinate_system_with_station_reference (see 5.2.4.89);

— property_definition_for_class_notation (see 5.2.4.150);

— property_definition_for_class_society (see 5.2.4.151);

— property_definition_for_corrugated_structure_function (see 5.2.4.152);

— property_definition_for_edge_cutout_function_parameters (see 5.2.4.153);

— property_definition_for_edge_feature_function_parameters (see 5.2.4.154);

— property_definition_for_homogeneous_ship_material_property (see 5.2.4.155);

— property_definition_for_interior_cutout_function_parameters (see 5.2.4.156);

— property_definition_for_local_co_ordinate_system (see 5.2.4.157);

— property_definition_for_local_co_ordinate_system_with_position (see 5.2.4.158);

— property_definition_for_panel_system_function_parameters (see 5.2.4.159);

— property_definition_for_plate_function_parameters (see 5.2.4.160);

— property_definition_for_plate_strake_function_parameters (see 5.2.4.161);

— property_definition_for_profile_function_parameters (see 5.2.4.162);

— property_definition_for_ship_material_property_parameters (see 5.2.4.163);

— property_definition_for_structural_system_function_parameters (see 5.2.4.164);

— property_definition_for_weld_filler_material_design (see 5.2.4.165);

— property_definition_has_references_with_name_loadline (see 5.2.4.166);

— property_definition_structural_weld_shrinkage_allowance_feature (see 5.2.4.167);

— property_definition_to_structural_added_material_feature_design (see 5.2.4.168);

— property_definition_with_lightship_weight_item (see 5.2.4.171);

— property_definition_with_weight_and_centre_of_gravity (see 5.2.4.173);

— representation_item_for_transformation_to_parent (see 5.2.4.220);

— ship_material_property_has_at_most_one_named_references (see 5.2.4.348).

5.2.3.2.27 property_definition_representation

The base definition of the **property_definition_representation** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **property_definition_-representation** entity:

— approvals_references_approval_history (see 5.2.4.15);

— change_impact_with_versionable_object_change_event (see 5.2.4.31);

— class_and_statutory_designation_has_properties (see 5.2.4.33);

— class_notation_with_named_representation_items (see 5.2.4.34);

— class_parameters_has_properties (see 5.2.4.35);

— define_maximum_permissible_still_water_bending_moment_values (see 5.2.4.40);

— definition_for_design_still_water_bending_moment_parameters (see 5.2.4.41);

— definition_for_design_vertical_wave_bending_moment_parameters (see 5.2.4.42);

— definition_for_design_vertical_wave_shear_force_parameters (see 5.2.4.43);

— define_maximum_permissible_still_water_shear_force_values (see 5.2.4.44);

— flare_area_parameters_has_exactly_two_representation_items (see 5.2.4.53);

— freeboard_characteristics_has_properties (see 5.2.4.54);

— global_axis_placement_has_properties (see 5.2.4.55);

— hull_cross_section_design_definition_for_representation_items (see 5.2.4.60);

— lightship_definition_has_properties (see 5.2.4.69);

— principal_characteristics_has_properties (see 5.2.4.80);

— product_definition_for_design_still_water_shear_force_values (see 5.2.4.83);

— product_definition_shape_for_assembly_manufacturing (see 5.2.4.92);

— product_definition_shape_for_bevel_design_parameters (see 5.2.4.93);

— product_definition_shape_for_circular_cutout_design_parameters (see 5.2.4.96);

— product_definition_shape_for_continuous_fillet_weld_design (see 5.2.4.98);

— product_definition_shape_for_corrugated_part_design (see 5.2.4.99);

— product_definition_shape_for_corrugated_structure_design (see 5.2.4.100);

— product_definition_shape_for_drain_hole_cutout_design (see 5.2.4.102);

— product_definition_shape_for_elliptical_cutout_design (see 5.2.4.103);

— product_definition_shape_for_elongated_oval_cutout_design (see 5.2.4.104);

— product_definition_shape_for_fillet_weld_design_parameters (see 5.2.4.105);

— product_definition_shape_for_free_form_interior_cutout_design (see 5.2.4.106);

— product_definition_shape_for_groove_weld_design (see 5.2.4.107);

— product_definition_shape_for_intermittent_fillet_weld_design (see 5.2.4.109);

— product_definition_shape_for_inward_round_corner_design (see 5.2.4.110);

— product_definition_shape_for_outer_contour_representation (see 5.2.4.116);

— product_definition_shape_for_outward_round_corner_design (see 5.2.4.117);

— product_definition_shape_for_panel_system_design_parameters (see 5.2.4.118);

— product_definition_shape_for_part_edge_cutout_design (see 5.2.4.119);

— product_definition_shape_for_plate_design_parameters (see 5.2.4.120);

— product_definition_shape_for_profile_design_parameters (see 5.2.4.122);

— product_definition_shape_for_rectangular_cutout_design (see 5.2.4.123);

— product_definition_shape_for_round_corner_rectangular_cutout (see 5.2.4.124);

— product_definition_shape_for_round_edge_rectangular_cutout (see 5.2.4.125);

— product_definition_shape_for_shear_bevel_design_parameters (see 5.2.4.126);

— product_definition_shape_for_spot_seam_weld_design (see 5.2.4.128);

— product_definition_shape_for_structural_part_joint_design (see 5.2.4.129);

— product_definition_shape_for_structural_system_design (see 5.2.4.130);

— product_definition_shape_for_triangular_cutout_design (see 5.2.4.131);

— product_definition_shape_for_welded_joint_design_parameters (see 5.2.4.132);

— product_definition_shape_has_references_for_weld_design (see 5.2.4.134);

— product_definition_shape_of_rectangular_cutback_corner_design (see 5.2.4.136);

— product_definition_shape_with_definition_matches_pattern (see 5.2.4.137);

— profile_cross_section_has_at_most_one_named_references (see 5.2.4.147);

— property_definition_for_class_notation (see 5.2.4.150);

— property_definition_for_corrugated_structure_function (see 5.2.4.152);

— property_definition_for_edge_cutout_function_parameters (see 5.2.4.153);

— property_definition_for_edge_feature_function_parameters (see 5.2.4.154);

— property_definition_for_homogeneous_ship_material_property (see 5.2.4.155);

— property_definition_for_interior_cutout_function_parameters (see 5.2.4.156);

— property_definition_for_local_co_ordinate_system (see 5.2.4.157);

— property_definition_for_local_co_ordinate_system_with_position (see 5.2.4.158);

— property_definition_for_panel_system_function_parameters (see 5.2.4.159);

— property_definition_for_plate_function_parameters (see 5.2.4.160);

— property_definition_for_plate_strake_function_parameters (see 5.2.4.161);

— property_definition_for_profile_function_parameters (see 5.2.4.162);

— property_definition_for_ship_material_property_parameters (see 5.2.4.163);

— property_definition_for_structural_system_function_parameters (see 5.2.4.164);

— property_definition_for_weld_filler_material_design (see 5.2.4.165);

— property_definition_has_references_with_name_loadline (see 5.2.4.166);

— property_definition_structural_weld_shrinkage_allowance_feature (see 5.2.4.167);

— property_definition_to_structural_added_material_feature_design (see 5.2.4.168);

— property_definition_with_lightship_weight_item (see 5.2.4.171);

— property_definition_with_weight_and_centre_of_gravity (see 5.2.4.173);

— representation_for_angle_bar_cross_section_dimensions (see 5.2.4.175);

— representation_for_assembly_manufacturing_definition (see 5.2.4.176);

— representation_for_bulbflat_cross_section_dimensions (see 5.2.4.177);

— representation_for_channel_cross_section_dimensions (see 5.2.4.178);

— representation_for_circular_hollow_profile_cross_section (see 5.2.4.180);

— representation_for_class_and_statutory_designation (see 5.2.4.181);

— representation_for_corrugated_part_design_parameters (see 5.2.4.182);

— representation_for_electrode_chemical_composition_design (see 5.2.4.187);

— representation_for_explicit_profile_cross_section_shape (see 5.2.4.190);

— representation_for_flanged_plate_cross_section_dimensions (see 5.2.4.191);

— representation_for_flat_bar_cross_section_dimensions (see 5.2.4.192);

— representation_for_global_axis_placement (see 5.2.4.193);

— representation_for_local_co_ordinate_system (see 5.2.4.196);

— representation_for_profile_curve_trace_line_design_parameters (see 5.2.4.199);

— representation_for_profile_function_parameters (see 5.2.4.200);

— representation_for_round_bar_cross_section_dimensions (see5.2.4.203);

— representation_for_square_tube_cross_section_dimensions (see 5.2.4.207);

— representation_for_structural_added_material_feature_design (see 5.2.4.208);

— representation_for_structural_part_joint_design (see 5.2.4.209);

— representation_for_structural_part_symmetry_relationship (see 5.2.4.210);

— representation_for_structural_system_symmetry_relationship (see 5.2.4.211);

— representation_for_t_bar_cross_section_dimensions (see 5.2.4.212);

— representation_for_twist_location_design_parameters (see 5.2.4.214);

— representation_has_references_with_name_moment_3d (see 5.2.4.216);

— representation_item_corrugated_structure_boundary_relationship (see 5.2.4.217);

— representation_item_for_panel_system_curve_boundary_design (see 5.2.4.218);

— representation_item_for_structural_weld_shrinkage_allowance (see 5.2.4.219);

— representation_item_to_structural_system_adjacency_relationship (see 5.2.4.221);

— representation_items_for_corrugated_part_design_parameters (see 5.2.4.222);

— representation_items_for_corrugated_structure_design (see 5.2.4.223);

— representation_items_for_corrugated_structure_function (see 5.2.4.224);

— representation_items_for_panel_system_boundary_relationship (see 5.2.4.233);

— representation_items_for_panel_system_function_parameters (see 5.2.4.234);

— representation_items_for_profile_boundary_relationship_design (see 5.2.4.235);

— representation_items_for_profile_curve_trace_line_design (see 5.2.4.236);

— representation_items_for_profile_trace_line_relationship_design (see 5.2.4.237);

— representation_items_optional_for_beveled_groove_weld_design(see 5.2.4.242);

— representation_items_optional_for_butt_groove_weld_design (see 5.2.4.243);

— representation_items_optional_for_class_notation (see 5.2.4.244);

— representation_items_optional_for_corrugation_dimensions (see 5.2.4.245);

— representation_items_optional_for_groove_weld_design (see 5.2.4.246);

— representation_items_optional_for_panel_system_design (see 5.2.4.248);

— representation_items_optional_for_plate_design_parameters (see 5.2.4.249);

— representation_items_optional_for_plate_function_parameters (see 5.2.4.250);

— representation_items_optional_for_plate_strake_function (see 5.2.4.251);

— representation_items_optional_for_principal_characteristics (see 5.2.4.252);

— representation_items_optional_for_profile_function_parameters (see 5.2.4.254);

— representation_items_optional_for_structural_system_design (see 5.2.4.256);

— representation_items_optional_for_weld_design_parameters (see 5.2.4.258);

— representation_items_optional_for_weld_filler_material_design (see 5.2.4.259);

— representation_items_optional_for_profile_design_parameters (see 5.2.4.253);

— representation_local_co_ordinate_system_with_position_reference (see 5.2.4.261);

— representation_restricted_by_name_class_notation (see 5.2.4.266);

— representation_restricted_by_name_class_parameters (see 5.2.4.267);

— representation_restricted_by_name_corrugation_dimensions (see 5.2.4.268);

— representation_restricted_by_name_flare_area_parameters (see 5.2.4.269);

— representation_restricted_by_name_plate_design_parameters (see 5.2.4.274);

— representation_restricted_by_name_plate_function_parameters (see 5.2.4.275);

— representation_restricted_by_name_principal_characteristics (see 5.2.4.276);

— representation_restricted_by_name_profile_design_parameters (see 5.2.4.277);

— representation_restricted_by_name_section_properties (see 5.2.4.278);

— representation_restricted_by_name_w_shape_cross_section (see 5.2.4.280);

— representation_restricted_for_assembly_manufacturing_position (see 5.2.4.281);

— representation_restricted_for_beveled_groove_weld_design (see 5.2.4.283);

— representation_restricted_for_butt_groove_weld_design (see 5.2.4.284);

— representation_restricted_for_composite_feature_design (see 5.2.4.285);

— representation_restricted_for_continuous_fillet_weld_design (see 5.2.4.286);

— representation_restricted_for_corrugated_structure_design (see 5.2.4.287);

— representation_restricted_for_corrugated_structure_function (see 5.2.4.288);

— representation_restricted_for_fillet_weld_design_parameters (see 5.2.4.289);

— representation_for_homogeneous_ship_material_property (see 5.2.4.291);

— representation_restricted_for_hull_cross_section_design_definition (see 5.2.4.292);

— representation_restricted_for_intermittent_fillet_weld_design (see 5.2.4.293);

— representation_restricted_for_panel_system_curve_boundary (see 5.2.4.294);

— representation_restricted_for_panel_system_design_parameters (see 5.2.4.295);

— representation_restricted_for_panel_system_function (see 5.2.4.296);

— representation_restricted_for_panel_system_plane_boundary (see 5.2.4.297);

— representation_restricted_for_plate_renewal_definition_attributes (see 5.2.4.298);

— representation_restricted_for_plate_strake_function (see 5.2.4.299);

— representation_restricted_for_ship_material_property (see 5.2.4.300);

— representation_restricted_for_spot_seam_weld_design (see 5.2.4.301);

— representation_restricted_for_structural_part_joint_design (see 5.2.4.302);

— representation_restricted_for_structural_system_design (see 5.2.4.303);

— representation_restricted_for_weld_design_parameters (see 5.2.4.305);

— representation_restricted_for_weld_manufacturing_definition (see 5.2.4.307);

— representation_restricted_for_welded_joint_design_parameters (see 5.2.4.308);

— representation_restricted_for_welding_sequence_representation (see 5.2.4.309);

— representation_restricts_profile_renewal_definition_attributes (see 5.2.4.310);

— shape_aspect_for_angle_bar_cross_section_dimensions (see 5.2.4.317);

— shape_aspect_for_assembly_manufacturing_position_parameters (see 5.2.4.318);

— shape_aspect_for_bulbflat_cross_section_dimensions (see 5.2.4.320);

— shape_aspect_for_channel_cross_section_dimensions (see 5.2.4.321);

— shape_aspect_for_circular_hollow_cross_section_dimensions (see 5.2.4.322);

— shape_aspect_for_composite_feature_design_parameters (see 5.2.4.323);

— shape_aspect_for_corrugation_dimensions (see 5.2.4.327);

— shape_aspect_for_explicit_profile_cross_section_dimensions (see 5.2.4.328);

— shape_aspect_for_flanged_plate_cross_section_dimensions (see 5.2.4.329);

— shape_aspect_for_flare_area_parameters (see 5.2.4.330);

— shape_aspect_for_flat_bar_cross_section_dimensions (see 5.2.4.331);

— shape_aspect_for_panel_system_curve_boundary_design (see 5.2.4.332);

— shape_aspect_for_panel_system_plane_boundary_design (see 5.2.4.333);

— shape_aspect_for_profile_curve_trace_line_design_parameters (see 5.2.4.334);

— shape_aspect_for_round_bar_cross_section_dimensions (see 5.2.4.335);

— shape_aspect_for_square_tube_cross_section_dimensions (see 5.2.4.336);

— shape_aspect_for_t_bar_cross_section_dimensions (see 5.2.4.338);

— shape_aspect_for_twist_location_design_parameters (see 5.2.4.339);

— shape_aspect_for_w_shape_cross_section_dimensions (see 5.2.4.340).

5.2.3.2.28 representation

The base definition of the **representation** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **representation** entity:

— class_notation_with_named_representation_items (see 5.2.4.34);

— flare_area_parameters_has_exactly_two_representation_items (see 5.2.4.53);

— hull_cross_section_design_definition_for_representation_items (see 5.2.4.60);

— hull_cross_section_result_for_representation_item (see 5.2.4.65);

— maximum_representation_items_for_class_library_definition (see 5.2.4.71);

— representation_design_vertical_wave_bending_moment(see 5.2.4.174);

— representation_for_angle_bar_cross_section_dimensions (see 5.2.4.175);

— representation_for_assembly_manufacturing_definition (see 5.2.4.176);

— representation_for_bulbflat_cross_section_dimensions (see 5.2.4.177);

— representation_for_channel_cross_section_dimensions (see 5.2.4.178);

— representation_for_circular_cutout_design_parameters (see 5.2.4.179);

— representation_for_circular_hollow_profile_cross_section (see 5.2.4.180);

— representation_for_class_and_statutory_designation (see 5.2.4.181);

— representation_for_corrugated_part_design_parameters (see 5.2.4.182);

— representation_for_design_vertical_wave_shear_force_parameters (see 5.2.4.183);

— representation_for_drain_hole_cutout_design_parameters (see 5.2.4.184);

— representation_for_edge_cutout_function_parameters (see 5.2.4.185);

— representation_for_edge_feature_function_parameters (see 5.2.4.186);

— representation_for_electrode_chemical_composition_design (see 5.2.4.187);

— representation_for_elliptical_cutout_design_parameters (see 5.2.4.188);

— representation_for_elongated_oval_cutout_design_parameters (see 5.2.4.189);

— representation_for_explicit_profile_cross_section_shape (see 5.2.4.190);

— representation_for_flanged_plate_cross_section_dimensions (see 5.2.4.191);

— representation_for_flat_bar_cross_section_dimensions (see 5.2.4.192);

— representation_for_global_axis_placement (see 5.2.4.193);

— representation_for_interior_cutout_function_parameters (see 5.2.4.194);

— representation_for_inward_round_corner_design_parameters (see 5.2.4.195);

— representation_for_local_co_ordinate_system (see 5.2.4.196);

— representation_for_outward_round_corner_design_parameters (see 5.2.4.197);

— representation_for_part_edge_cutout_design_parameters (see 5.2.4.198);

— representation_for_profile_curve_trace_line_design_parameters (see 5.2.4.199);

— representation_for_profile_function_parameters (see 5.2.4.200);

— representation_for_rectangular_cutback_corner_design (see 5.2.4.201);

— representation_for_rectangular_cutout_design_parameters (see 5.2.4.202);

— representation_for_round_corner_rectangular_cutout_design (see 5.2.4.204);

— representation_for_round_edge_rectangular_cutout_design (see 5.2.4.205);

— representation_for_shear_bevel_design_parameters (see 5.2.4.206);

— representation_for_square_tube_cross_section_dimensions (see 5.2.4.207);

— representation_for_structural_added_material_feature_design (see 5.2.4.208);

— representation_for_structural_part_joint_design (see 5.2.4.209);

— representation_for_structural_part_symmetry_relationship (see 5.2.4.210);

— representation_for_structural_system_symmetry_relationship (see 5.2.4.211);

— representation_for_t_bar_cross_section_dimensions (see 5.2.4.212);

— representation_for_triangular_cutout_design_parameters (see 5.2.4.213);

— representation_for_twist_location_design_parameters (see 5.2.4.214);

— representation_for_moment_3d_restricts_representation_item (see 5.2.4.311);

— representation_for_round_bar_cross_section_dimensions (see5.2.4.203);

— representation_has_global_unit_assigned_context (see 5.2.4.215);

— representation_has_references_with_name_moment_3d (see 5.2.4.216);

— representation_item_corrugated_structure_boundary_relationship (see 5.2.4.217);

— representation_item_for_panel_system_curve_boundary_design (see 5.2.4.218);

— representation_item_for_structural_weld_shrinkage_allowance (see 5.2.4.219);

— representation_item_for_transformation_to_parent (see 5.2.4.220);

— representation_item_to_structural_system_adjacency_relationship (see 5.2.4.221);

— representation_items_for_corrugated_part_design_parameters (see 5.2.4.222);

— representation_items_for_corrugated_structure_design (see 5.2.4.223);

— representation_items_for_corrugated_structure_function (see 5.2.4.224);

— representation_items_for_edge_cutout_function_parameters (see 5.2.4.225);

— representation_items_for_edge_feature_function_parameters (see 5.2.4.226);

— representation_items_for_elongated_oval_cutout_design (see 5.2.4.227);

— representation_items_for_interior_cutout_function_parameters (see 5.2.4.228);

— representation_items_for_inward_round_corner_design (see 5.2.4.229);

— representation_items_for_outward_round_corner_design (see 5.2.4.232);

— representation_items_for_panel_system_boundary_relationship (see 5.2.4.233);

— representation_items_for_panel_system_function_parameters (see 5.2.4.234);

— representation_items_for_profile_boundary_relationship_design (see 5.2.4.235);

— representation_items_for_profile_curve_trace_line_design (see 5.2.4.236);

— representation_items_for_profile_trace_line_relationship_design (see 5.2.4.237);

— representation_items_for_rectangular_cutback_corner_design (see 5.2.4.238);

— representation_items_for_shear_bevel_design_parameters (see 5.2.4.240);

— representation_items_optional_for_bevel_design_parameters (see 5.2.4.241);

— representation_items_optional_for_butt_groove_weld_design (see 5.2.4.243);

— representation_items_optional_for_class_notation (see 5.2.4.244);

— representation_items_optional_for_corrugation_dimensions (see 5.2.4.245);

— representation_items_optional_for_groove_weld_design (see 5.2.4.246);

— representation_items_optional_for_owner_designation (see 5.2.4.247);

— representation_items_optional_for_panel_system_design (see 5.2.4.248);

— representation_items_optional_for_plate_design_parameters (see 5.2.4.249);

— representation_items_optional_for_plate_function_parameters (see 5.2.4.250);

— representation_items_optional_for_plate_strake_function (see 5.2.4.251);

— representation_items_optional_for_principal_characteristics (see 5.2.4.252);

— representation_items_optional_for_profile_design_parameters (see 5.2.4.253);

— representation_items_optional_for_profile_function_parameters (see 5.2.4.254);

— representation_items_optional_for_seam_design_parameters (see 5.2.4.255);

— representation_items_optional_for_structural_system_design (see 5.2.4.256);

— representation_items_optional_for_structural_system_function (see 5.2.4.257);

— representation_items_optional_for_weld_design_parameters (see 5.2.4.258);

— representation_items_optional_for_weld_filler_material_design (see 5.2.4.259);

- representation_items_optional_for_beveled_groove_weld_design(see 5.2.4.242);
- representation_maximum_permissible_still_water_bending_moment (see 5.2.4.262);
- representation_of_design_still_water_bending_moment_parameters (see 5.2.4.263);
- representation_local_co_ordinate_system_with_position_reference (see 5.2.4.261);
- representation_of_maximum_permissible_still_water_shear_force (see 5.2.4.264);
- representation_restricted_by_name_bevel_design_parameters (see 5.2.4.265);
- representation_restricted_by_name_class_notation (see 5.2.4.266);
- representation_restricted_by_name_class_parameters (see 5.2.4.267);
- representation_restricted_by_name_corrugation_dimensions (see 5.2.4.268);
- representation_restricted_by_name_flare_area_parameters (see 5.2.4.269);
- representation_restricted_by_name_freeboard_characteristics (see 5.2.4.270);
- representation_restricted_by_name_lightship_definition (see 5.2.4.271);
- representation_restricted_by_name_lightship_weight_item (see 5.2.4.272);
- representation_restricted_by_name_loadline (see 5.2.4.273);
- representation_restricted_by_name_plate_design_parameters (see 5.2.4.274);
- representation_restricted_by_name_plate_function_parameters (see 5.2.4.275);
- representation_restricted_by_name_principal_characteristics (see 5.2.4.276);
- representation_restricted_by_name_profile_design_parameters (see 5.2.4.277);
- representation_restricted_by_name_section_properties (see 5.2.4.278);
- representation_restricted_by_name_table_of_shear_force_values (see 5.2.4.279);
- representation_restricted_by_name_w_shape_cross_section (see 5.2.4.280);
- representation_restricted_for_assembly_manufacturing_position (see 5.2.4.281);
- representation_restricted_weight_and_centre_of_gravity (see 5.2.4.282);
- representation_restricted_for_beveled_groove_weld_design (see 5.2.4.283);
- representation_restricted_for_butt_groove_weld_design (see 5.2.4.284);

— representation_restricted_for_composite_feature_design (see 5.2.4.285);

— representation_restricted_for_continuous_fillet_weld_design (see 5.2.4.286);

— representation_restricted_for_corrugated_structure_design (see 5.2.4.287);

— representation_restricted_for_corrugated_structure_function (see 5.2.4.288);

— representation_restricted_for_fillet_weld_design_parameters (see 5.2.4.289);

— representation_restricted_for_groove_weld_design (see 5.2.4.290);

— representation_for_homogeneous_ship_material_property (see 5.2.4.291);

— representation_restricted_for_hull_cross_section_design_definition (see 5.2.4.292);

— representation_restricted_for_intermittent_fillet_weld_design (see 5.2.4.293);

— representation_restricted_for_panel_system_curve_boundary (see 5.2.4.294);

— representation_restricted_for_panel_system_design_parameters (see 5.2.4.295);

— representation_restricted_for_panel_system_function (see 5.2.4.296);

— representation_restricted_for_panel_system_plane_boundary (see 5.2.4.297);

— representation_restricted_for_plate_renewal_definition_attributes (see 5.2.4.298);

— representation_restricted_for_plate_strake_function (see 5.2.4.299);

— representation_restricted_for_ship_material_property (see 5.2.4.300);

— representation_restricted_for_spot_seam_weld_design (see 5.2.4.301);

— representation_restricted_for_structural_part_joint_design (see 5.2.4.302);

— representation_restricted_for_structural_system_design (see 5.2.4.303);

— representation_restricted_for_structural_system_function (see 5.2.4.304);

— representation_restricted_for_weld_design_parameters (see 5.2.4.305);

— representation_restricted_for_weld_filler_material_design (see 5.2.4.306);

— representation_restricted_for_weld_manufacturing_definition (see 5.2.4.307);

— representation_restricted_for_welded_joint_design_parameters (see 5.2.4.308);

— representation_restricted_for_welding_sequence_representation (see 5.2.4.309);

— representation_restricts_profile_renewal_definition_attributes (see 5.2.4.310);

— source_for_library_definition (see 5.2.4.349);

— user_def_function_description_required (see 5.2.4.357);

— user_defined_tightness_description_required (see 5.2.4.359).

5.2.3.2.29 representation_item

The base definition of the **representation_item** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **representation_item** entity:

— class_notation_with_named_representation_items (see 5.2.4.34);

— hull_cross_section_result_for_representation_item (see 5.2.4.65);

— representation_for_assembly_manufacturing_definition (see 5.2.4.176);

— representation_for_class_and_statutory_designation (see 5.2.4.181);

— representation_for_electrode_chemical_composition_design (see 5.2.4.187);

— representation_for_global_axis_placement (see 5.2.4.193);

— representation_for_local_co_ordinate_system (see 5.2.4.196);

— representation_for_structural_added_material_feature_design (see 5.2.4.208);

— representation_for_structural_part_joint_design (see 5.2.4.209);

— representation_for_structural_system_symmetry_relationship (see 5.2.4.211);

— representation_item_corrugated_structure_boundary_relationship (see 5.2.4.217);

— representation_item_for_panel_system_curve_boundary_design (see 5.2.4.218);

— representation_item_for_structural_weld_shrinkage_allowance (see 5.2.4.219);

— representation_item_to_structural_system_adjacency_relationship (see 5.2.4.221);

— representation_items_for_corrugated_structure_design (see 5.2.4.223);

— representation_items_for_corrugated_structure_function (see 5.2.4.224);

— representation_items_for_panel_system_boundary_relationship (see 5.2.4.233);

— representation_items_for_panel_system_function_parameters (see 5.2.4.234);

— representation_items_optional_for_butt_groove_weld_design (see 5.2.4.243);

— representation_items_optional_for_class_notation (see 5.2.4.244);

— representation_items_optional_for_groove_weld_design (see 5.2.4.246);

— representation_items_optional_for_panel_system_design (see 5.2.4.248);

— representation_items_optional_for_plate_strake_function (see 5.2.4.251);

— representation_items_optional_for_principal_characteristics (see 5.2.4.252);

— representation_items_optional_for_structural_system_design (see 5.2.4.256);

— representation_items_optional_for_weld_design_parameters (see 5.2.4.258);

— representation_items_optional_for_weld_filler_material_design (see 5.2.4.259);

— representation_items_specified_for_design_swsf_values (see 5.2.4.260);

— representation_items_optional_for_beveled_groove_weld_design(see 5.2.4.242);

— representation_local_co_ordinate_system_with_position_reference (see 5.2.4.261);

— representation_restricted_by_name_class_notation (see 5.2.4.266);

— representation_restricted_by_name_class_parameters (see 5.2.4.267);

— representation_restricted_by_name_flare_area_parameters (see 5.2.4.269);

— representation_restricted_by_name_principal_characteristics (see 5.2.4.276);

— representation_restricted_for_assembly_manufacturing_position (see 5.2.4.281);

— representation_restricted_for_beveled_groove_weld_design (see 5.2.4.283);

— representation_restricted_for_butt_groove_weld_design (see 5.2.4.284);

— representation_restricted_for_composite_feature_design (see 5.2.4.285);

— representation_restricted_for_continuous_fillet_weld_design (see 5.2.4.286);

— representation_restricted_for_corrugated_structure_design (see 5.2.4.287);

— representation_restricted_for_corrugated_structure_function (see 5.2.4.288);

— representation_restricted_for_fillet_weld_design_parameters (see 5.2.4.289);

— representation_restricted_for_groove_weld_design (see 5.2.4.290);

— representation_for_homogeneous_ship_material_property (see 5.2.4.291);

— representation_restricted_for_hull_cross_section_design_definition (see 5.2.4.292);

— representation_restricted_for_intermittent_fillet_weld_design (see 5.2.4.293);

— representation_restricted_for_panel_system_curve_boundary (see 5.2.4.294);

— representation_restricted_for_panel_system_design_parameters (see 5.2.4.295);

— representation_restricted_for_panel_system_function (see 5.2.4.296);

— representation_restricted_for_panel_system_plane_boundary (see 5.2.4.297);

— representation_restricted_for_plate_renewal_definition_attributes (see 5.2.4.298);

— representation_restricted_for_plate_strake_function (see 5.2.4.299);

— representation_restricted_for_ship_material_property (see 5.2.4.300);

— representation_restricted_for_spot_seam_weld_design (see 5.2.4.301);

— representation_restricted_for_structural_part_joint_design (see 5.2.4.302);

— representation_restricted_for_structural_system_design (see 5.2.4.303);

— representation_restricted_for_weld_design_parameters (see 5.2.4.305);

— representation_restricted_for_weld_manufacturing_definition (see 5.2.4.307);

— representation_restricted_for_welded_joint_design_parameters (see 5.2.4.308);

— representation_restricted_for_welding_sequence_representation (see 5.2.4.309);

— representation_restricts_profile_renewal_definition_attributes (see 5.2.4.310);

— source_for_library_definition (see 5.2.4.349);

— user_def_function_description_required (see 5.2.4.357);

— user_defined_tightness_description_required (see 5.2.4.359).

5.2.3.2.30 shape_aspect

The base definition of the **shape_aspect** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **shape_aspect** entity:

— corrugated_structure_design_definition_has_at_least_one_class (see 5.2.4.36);

— corrugated_structure_design_definition_has_at_most_one_class (see 5.2.4.37);

— library_manufacturing_definition_for_class_references (see 5.2.4.68);

— panel_system_design_definition_has_at_most_one_class (see 5.2.4.75);

— panel_system_design_definition_has_at_least_one_class(see 5.2.4.74);

— plate_design_definition_has_at_least_one_class_references (see 5.2.4.77);

— plate_design_definition_has_at_most_one_class_references (see 5.2.4.78);

— plate_manufacturing_definition_has_class_references (see 5.2.4.79);

— product_definition_shape_for_manufacturing_definition (see 5.2.4.114);

— product_definition_shape_for_source (see 5.2.4.127);

— product_definition_shape_for_library_design_parameters (see 5.2.4.112);

— product_definition_shape_for_name (see 5.2.4.115);

— product_definition_shape_for_library_manufacturing_definition (see 5.2.4.113);

— product_definition_shape_for_library_definition_parameters (see 5.2.4.111);

— product_definition_shape_for_class_torch_vector (see 5.2.4.97);

— product_definition_shape_for_corrugation (see 5.2.4.101);

— product_definition_shape_for_profile_cross_section (see 5.2.4.121);

— product_definition_shape_has_references_for_centre_of_gravity (see 5.2.4.133);

— product_definition_shape_has_references_with_class_border (see 5.2.4.135);

— profile_cross_section_has_at_most_one_named_references (see 5.2.4.147);

— profile_design_definition_has_at_most_one_class_references (see 5.2.4.148);

— profile_design_definition_has_zero_or_two_class_references (see 5.2.4.149);

— seam_has_at_most_one_class_references (see 5.2.4.316);

— shape_aspect_for_angle_bar_cross_section_dimensions (see 5.2.4.317);

— shape_aspect_for_assembly_manufacturing_position_parameters (see 5.2.4.318);

— shape_aspect_for_border_corner_cutout_boundary_relationship (see 5.2.4.319);

— shape_aspect_for_bulbflat_cross_section_dimensions (see 5.2.4.320);

— shape_aspect_for_channel_cross_section_dimensions (see 5.2.4.321);

— shape_aspect_for_circular_hollow_cross_section_dimensions (see 5.2.4.322);

— shape_aspect_for_composite_feature_design_parameters (see 5.2.4.323);

— shape_aspect_for_composite_feature_has_at_least_two_instances (see 5.2.4.324);

— shape_aspect_for_corner_cutout (see 5.2.4.325);

— shape_aspect_for_corner_cutout_boundary_relationship (see 5.2.4.326);

— shape_aspect_for_corrugation_dimensions (see 5.2.4.327);

— shape_aspect_for_explicit_profile_cross_section_dimensions (see 5.2.4.328);

— shape_aspect_for_flanged_plate_cross_section_dimensions (see 5.2.4.329);

— shape_aspect_for_flare_area_parameters (see 5.2.4.330);

— shape_aspect_for_flat_bar_cross_section_dimensions (see 5.2.4.331);

— shape_aspect_for_panel_system_curve_boundary_design (see 5.2.4.332);

— shape_aspect_for_panel_system_plane_boundary_design (see 5.2.4.333);

— shape_aspect_for_profile_curve_trace_line_design_parameters (see 5.2.4.334);

— shape_aspect_for_round_bar_cross_section_dimensions (see 5.2.4.335);

— shape_aspect_for_square_tube_cross_section_dimensions (see 5.2.4.336);

— shape_aspect_for_structural_cutout_boundary_relationship (see 5.2.4.337);

— shape_aspect_for_t_bar_cross_section_dimensions (see 5.2.4.338);

— shape_aspect_for_twist_location_design_parameters (see 5.2.4.339);

— shape_aspect_for_w_shape_cross_section_dimensions (see 5.2.4.340).

5.2.3.2.31 shape_aspect_relationship

The base definition of the **shape_aspect_relationship** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **shape_aspect_relationship** entity:

— seam_has_at_most_one_class_references (see 5.2.4.316);

— shape_aspect_relationship_attributes_references_are_distinct (see 5.2.4.341).

5.2.3.2.32 versioned_action_request

The base definition of the **versioned_action_request** entity is given in ISO 10303-41. The following modifications apply to this part of ISO 10303.

Associated global rules:

The following global rules defined in this part of ISO 10303 apply to the **versioned_action_request** entity:

— author_for_change_request (see 5.2.4.19);

— date_time_for_change_request (see 5.2.4.39);

— initiator_for_change_request (see 5.2.4.67).

5.2.4 Ship structures rule definitions

5.2.4.1 action_request_solution_connected_to_action

The **action_request_solution_connected_to_action** rule specifies that each instance of type **action_request_solution** with class 'change plan' is connected to an instance of type **action** with class 'change' via an instance of **action_method**.

EXPRESS specification

```
*)
RULE
action_request_solution_connected_to_action
FOR(action_request_solution, action);
```



```
  LOCAL
    t1_set  : SET OF action_request_solution := [];
    t2_set  : SET OF action := [];
    set_3   : SET OF ACTION_METHOD := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
t1_set := QUERY(a <* action_request_solution |
          VALUE_IN(WHICH_CLASS(a), 'change plan'));
t2_set := QUERY(b <* action | VALUE_IN(WHICH_CLASS(b), 'change'));
REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
  set_3 := [];
  REPEAT j := 1 TO HIINDEX(t2_set);
    set_3 := set_3 + [t2_set[j].chosen_method];
  END_REPEAT;
  violate := VALUE_IN(set_3, t1_set[i].method);
END_REPEAT;
WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

action_request_solution: the set of all instances of **action_request_solution** entities.
action: the set of all instances of **action** entities.

Formal propositions:

WR1: Every instance of **action_method** that is the **chosen_method** of an instance of action is at the same time to be the method of an instance of **action_request_solution**.

5.2.4.2 action_request_solution_with_identification_assignment

The **action_request_solution_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE action_request_solution_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF action_request_solution := [];
    t2_set    : SET OF action_request_solution := [];
    arg_list  : LIST OF STRING := [ 'definable object'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'moulded form
  characteristics definition' *)
REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =  arg_LIST[j]);
END_REPEAT;
  (* get all instances of property_definition that have class id *)
REPEAT i := 1 TO HIINDEX(c_a_set);
```



```
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
END_REPEAT;
REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
  violation := NOT (SIZEOF(T2_SET) = 1);
END_REPEAT;
WHERE
  WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **property_definition** that is referenced by an **applied_classification_-assignment** whose assigned_classification has a **name** attribute of value 'definable object' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.3 action_with_identification_assignment

The **action_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE action_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   :  SET OF applied_classification_assignment := [];
    t1_set    :  SET OF action := [];
    t2_set    :  SET OF action := [];
    arg_list  :  LIST OF STRING := [ 'definable object'];
    violation :  LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id *)
REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME =  arg_LIST[j]);
END_REPEAT;

  (* get all instances of property_definition that have class id *)
REPEAT i := 1 TO HIINDEX(c_a_set);
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
    t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
END_REPEAT;
REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
```



```
      violation := NOT (SIZEOF(T2_SET) = 1);
END_REPEAT;
WHERE
  WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **property_definition** that is referenced by an **applied_classification_-assignment** whose assigned_classification has a **name** attribute of value 'definable object' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.4 alternative_version_relationship_versionable_object

The **alternative_version_relationship_versionable_object** rule specifies an instance **identification_assignment_relationship** with class 'alternative version relationship' only relates instances of type **applied_identification_assignment** with class equal to four.

EXPRESS specification:

```
*)
RULE alternative_version_relationship_versionable_object
FOR (applied_identification_assignment);
  LOCAL
    violate1, violate2, violate : LOGICAL := FALSE;
  END_LOCAL;
  (* get all instances of applied_identification_assignment *)
REPEAT i := 1 TO HIINDEX(applied_identification_assignment) BY 1
  WHILE NOT violate;
    violate1 := (SIZEOF(USEDIN(applied_identification_assignment[i],
    ('SHIP_STRUCTURES_SCHEMA.IDENTIFICATION_ASSIGNMENT_RELATIONSHIP.'
     +'RELATING_IDENTIFICATION_ASSIGNMENT')))
     > 0) AND (NOT VALUE_IN(which_class(
      applied_identification_assignment[i]),'versionable object'));
      violate2 := (SIZEOF(USEDIN(applied_identification_assignment[i],
      ('SHIP_STRUCTURES_SCHEMA.IDENTIFICATION_ASSIGNMENT_RELATIONSHIP.'
       +'RELATED_IDENTIFICATION_ASSIGNMENT')))
       > 0) AND (NOT VALUE_IN(which_class(
       applied_identification_assignment[i]),'versionable object'));
      violate := violate1 OR violate2;
END_REPEAT;
WHERE
  WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_identification_assignment: the set of all instances of **applied_identification_assignment** entities.

Formal propositions:

WR1: Every instance of **identification_assignment_relationship** that has an **applied_-classification_assignment** whose **assigned_classification.name** equals 'alternative version relationship' only reference or be referenced by instances of **applied_identification_assignment** that have an **applied_classification_assignment** whose **assigned_classification.name** equals 'versionable object'.

5.2.4.5 alternative_version_relationship_has_mandatory_description

The **alternative_version_relationship_has_mandatory_description** rule specifies that for an instance of **identification_assignment_relationship** with class id 'alternative version relationship' the optional attribute **description** is instantiated.

EXPRESS specification:

```
*)
RULE alternative_version_relationship_has_mandatory_description
FOR (identification_assignment_relationship);
LOCAL
  t1_set  : SET OF identification_assignment_relationship := [];
  violate : LOGICAL := FALSE;
  END_LOCAL;
  (* get all instances of identification_assignment_relationship *)
  (*being classified as 'alternative version relationship' *)
t1_set := QUERY(i <* identification_assignment_relationship |
  VALUE_IN(WHICH_CLASS(i), 'alternative version relationship'));
  (* from all instances found above:
   find those for which attribute description is not instantiated
  *)
violate := (SIZEOF(QUERY(k <* t1_set | NOT EXISTS (k.description))) > 0);
WHERE
  WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

identification_assignment_relationship: the set of all instances of **identification_assignment_-relationship** entities.

Formal propositions:

WR1: The optional attribute **description** is to exist for every instance of **identification_-assignment_relationship** that has an **applied_classification_assignment** whose **assigned_-classification.name** equals 'alternative version relationship'.

5.2.4.6 alternative_version_relationship_has_unique_versions

The **alternative_version_relationship_has_unique_versions** rule specifies that all instance of **identification_assignment_relationship** with class equal to 'alternative version relationship', versionable objects successor and predecessor must be different, which means it must expose different version_ids.

EXPRESS specification:

```
*)
RULE alternative_version_relationship_has_unique_versions
FOR (identification_assignment_relationship);
  LOCAL
    t1_set  : SET OF identification_assignment_relationship := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  (* get all instances of identification_assignment_relationship with   *)
  (* class 'alternative version relationship' *)
   t1_set := QUERY(a <* identification_assignment_relationship |
      VALUE_IN(WHICH_CLASS(a), 'alternative version relationship'));
REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
  violate :=
    (t1_set[i].relating_identification_assignment.assigned_id =
     t1_set[i].related_identification_assignment.assigned_id);
END_REPEAT;
WHERE
  WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

identification_assignment_relationship: the set of all instances of **identification_assignment_-relationship** entities.

Formal propositions:

WR1: The **assigned_id** of the **related_identification_assignment** and the **relating_identification_-assignment** of every instance of **identification_assignment_relationship** that has an **applied_-classification_assignment** whose **assigned_classification.name** equals 'alternative version relationship' is distinct.

5.2.4.7 annotation_compound_representation_has_specified_name

The **annotation_compound_representation_has_specified_name** rule specifies the **item_element** attribute of a **compound_representation_item** with the class id 'annotation' to have in the **list_representation_item** for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list.

EXPRESS specification:

```
*)
RULE annotation_compound_representation_has_specified_name
  FOR (compound_representation_item);
    LOCAL
      c_a_set   : SET OF applied_classification_assignment := [];
      t1_set    : SET OF compound_representation_item := [];
      t2_set    : SET OF representation_item := [];
      arg_list  : LIST OF STRING := ['location', 'text'];
      violation : LOGICAL := FALSE;
    END_LOCAL;
    (* get all classification_assignment instances with id 'annotation' *)
     c_a_set := QUERY(i <* applied_classification_assignment |
 i.ASSIGNED_CLASS.NAME = 'annotation');
```

```
    (* get all instances of compound_representation_item that have class id
     'annotation' *)
REPEAT i := 1 TO HIINDEX(c_a_set);
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
    t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
END_REPEAT;
(* iterate over all compound_representation_item found above; stop, if one
of
   them has not exactly one rep_item for each name in the arg_list *)
REPEAT i:=1 TO HIINDEX(t1_set) WHILE (NOT violation);
  REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
    t2_set := t1_set[i].item_element;
    violation := (SIZEOF(QUERY(items <* t2_set | items.name = arg_list[j]))
    <> 1);
  END_REPEAT;
END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

Compound_representation_item: the set of all instances of **compound_representation_item** entities.

Formal propositions:

WR1: Every instance of **compound_representation_item** that has an **applied_classification_assignment**
whose attribute **assigned_class.name** equals 'annotation' is to have its attribute **item_element** instantiated as a **list_representation_item** which for each value out of 'location' and 'text' collect exactly one instance of **representation_item** whose name attribute equals that value.

5.2.4.8 applied_approval_assignment_has_exactly_one_elements

The **applied_approval_assignment_has_exactly_one_elements** rule specifies the aggregate type attribute **items** of entity data type **applied_approval_assignment** with the **object_role** that identifies entities with attribute **name** equal to 'subject' to have exactly one elements.

EXPRESS specification:

```
*)
RULE  applied_approval_assignment_has_exactly_one_elements
FOR (object_role, applied_approval_assignment);
  WHERE
    WR1: SIZEOF(QUERY(ass_inst <* applied_approval_assignment |
           NOT((ass_inst.role.name = 'subject')
               AND
               (SIZEOF(ass_inst.items) = 1)
              )
           )) = 0;
END_RULE;
(*
```


Argument definitions:

applied_approval_assignment: the set of all instances of **applied_approval_assignment** entities.
object_role: the set of all instances of **object_role** entities.

Formal propositions:

WR1: Every instance of **applied_approval_assignment** that has an attribute role **name** equal to 'subject' collect one elements in its **items** attribute.

5.2.4.9 applied_group_assignment_has_at_least_one_elements

The **applied_group_assignment_has_at_least_one_elements** rule specifies the aggregate type attribute **items** of "assignment" entity type **applied_group_assignment** with the **object_role** identifies an entity with attribute **name** equal to 'approvals' to have at least one elements all of which is of type **approval**.

EXPRESS specification:

```
*)
RULE
applied_group_assignment_has_at_least_one_elements
FOR (object_role, applied_group_assignment);
WHERE
    WR1: SIZEOF(QUERY(ass_inst <* applied_group_assignment |
           NOT((ass_inst.role.name = 'approvals')
               AND
               (SIZEOF(ass_inst.items) >= 1)
               AND
               (SIZEOF(QUERY(item <* ass_inst.items |
                  NOT('SHIP_STRUCTURES_SCHEMA.APPROVAL'
                  IN TYPEOF(item)))) = 0)
              )
           )) = 0;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
object_role: the set of all instances of **object_role** entities.

Formal propositions:

WR1: Every **applied_group_assignment** whose attribute **role** is an entity with **name** equals 'approvals' collect one or more instances of **approval** in its aggregate type attribute **items**.

5.2.4.10 approval_event_with_approval_date_time

The **approval_event_with_approval_date_time** rule specifies that an instance of **approval** with class id 'approval event' is referenced by exactly one instance of **approval_date_time** via **dated_-approval**.

EXPRESS specification:

```
*)
RULE
approval_event_with_approval_date_time
FOR(approval, approval_date_time);
LOCAL
    t1_set  : SET OF approval := [];
    t2_set  : SET OF approval_date_time := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* approval |
    VALUE_IN(WHICH_CLASS(i), 'approval event'));

  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPROVAL_DATE_TIME.' +
      'DATED_APPROVAL'));
    violate := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

approval: the set of all instances of **approval** entities.
approval_date_time: the set of all instances of **approval_date_time** entities.

Formal propositions:

WR1: Every instance of **approval** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'approval event' is referenced by exactly one instance of **approval_date_time** through **dated_approval**.

5.2.4.11 approval_event_with_approval_person_organization

The **approval_event_with_approval_person_organization** rule specifies that an instance of **approval** with class id 'approval event' is referenced by exactly one instance of **approval_person_-organization** via **authorized_approval**.

EXPRESS specification:

```
*)
RULE
approval_event_with_approval_person_organization
FOR(approval, approval_person_organization);
LOCAL
    t1_set  : SET OF approval := [];
    t2_set  : SET OF approval_person_organization := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* approval |
            VALUE_IN(WHICH_CLASS(i), 'approval event'));
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPROVAL_PERSON_ORGANIZATION.' +
```



```
      'AUTHORIZED_APPROVAL'));
    violate := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

approval: the set of all instances of **approval** entities.
approval_person_organization: the set of all instances of **approval_person_organization** entities.

Formal propositions:

WR1: Every instance of **approval** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'approval event' is referenced by exactly one instance of **approval_person_organization** through **authorised_approval**.

5.2.4.12 approval_history_approves_same_definition

The **approval_history_approves_same_definition** rule specifies that all members in a **group** with class id 'approval history' are to approve the same instance with class 'definition'.

EXPRESS specification:

```
*)
RULE
approval_history_approves_same_definition
FOR (applied_group_assignment, applied_approval_assignment);
  LOCAL
    t2_set  :  SET OF APPLIED_GROUP_ASSIGNMENT := [];
    t3_set  :  SET OF GENERIC :=[];
    t4_set  :  SET OF GENERIC :=[];
    t5_set  :  SET OF APPLIED_APPROVAL_ASSIGNMENT := [];
    violate :  LOGICAL := FALSE;
  END_LOCAL;
  t2_set := QUERY(a <* APPLIED_GROUP_ASSIGNMENT |
                VALUE_IN(WHICH_CLASS(a.ASSIGNED_GROUP),
                'approval history'));
  t3_set := QUERY(b <* t2_set[1].items |
    'SHIP_STRUCTURES_SCHEMA.APPROVAL' IN TYPEOF(b));
  t4_set := QUERY(b <* t2_set[1].items |
    VALUE_IN(WHICH_CLASS(b), 'DEFINITION'));
  violate := NOT(SIZEOF(t4_set) = 1);
REPEAT i := 1 TO HIINDEX(t3_set) WHILE NOT violate;
  t5_set := QUERY(a <* APPLIED_APPROVAL_ASSIGNMENT |
  (a.ASSIGNED_APPROVAL = t3_set[i]) AND
  (NOT (VALUE_IN(a.items, t4_set[1]))));
  violate := (SIZEOF(t5_set) > 0);
END_REPEAT;
  WHERE
    WR1: NOT violate;
    WR2: (SIZEOF(t4_set) = 1);
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
applied_approval_assignment: the set of all instances of **applied_approval_assignment** entities.

Formal propositions:

WR1: Every instance of **approval** that is a member of the **items** attribute of an instance of **applied_group_assignment** whose **assigned_group** is an instance of **group** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'approval history' is **assigned_approval** of an instance of **applied_approval_assignment** that collects the same instance that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'definition' in its **items** attribute.

5.2.4.13 approval_history_contains_approval_elements

The **approval_history_contains_approval_elements** specifies that each **group_assignment** with **role** 'approvals' and an assigned **group** with class **approval_history** all **approval** that are collected in the **items** attribute of **assigned_approval** is referenced by instances of assignment entity **applied_approval_assignment** with the **role** 'subject' via the attribute **assigned_approval** that has the same and only one instance in its **items** attribute.

EXPRESS specification:

```
*)
RULE approval_history_contains_approval_elements
FOR (object_role, applied_group_assignment);
  LOCAL
    t1_set          : SET OF applied_approval_assignment := [];
    t3_set          : SET OF approval := [];
    gr_ass_set      : SET OF applied_group_assignment := [];
    violation       : LOGICAL := FALSE;
    first_runs      : LOGICAL := TRUE;
    valid_items_set : SET OF GENERIC := [];
  END_LOCAL;
  gr_ass_set := QUERY(aga <* applied_group_assignment |
       (aga.role.name = 'approvals')
       AND
      ('approval history' IN WHICH_CLASS(aga.assigned_group)));
  REPEAT i := 1 TO HIINDEX(gr_ass_set) WHILE NOT violation;
    t3_set := QUERY(aga_item <* gr_ass_set[i].items |
      'SHIP_STRUCTURES_SCHEMA.APPROVAL' IN TYPEOF(aga_item));
     REPEAT j := 1 TO HIINDEX(t3_set);
      t1_set := QUERY(aga <* bag_to_set(USEDIN(t3_set[j],
         'SHIP_STRUCTURES_SCHEMA.' +
         'APPROVAL_ASSIGNMENT.ASSIGNED_APPROVAL'))|
         aga.role.name = 'subject'
         );
      REPEAT k:= 1 TO HIINDEX(t1_set);
        IF ((first_runs) AND (SIZEOF(t1_set[k].items) > 0)) THEN
          valid_items_set := t1_set[k].items;
          first_runs := FALSE;
        ELSE
          REPEAT l := 1 TO HIINDEX(t1_set[k].items);
            violation := t1_set[k].items[l] :<>: valid_items_set[l];
          END_REPEAT;
```



```
        END_IF;
      END_REPEAT;
    END_REPEAT;
  first_runs := TRUE;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
object_role: the set of all instances of **object_role** entities.

Formal propositions:

WR1: All items of all instances of **applied_group_assignment** whose **role** equals 'subject' and that have an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals **approval_history** that are instances of **approval** is referenced by the same instances of **applied_approval_assignment** whose **role** equals 'subject' through attribute **assigned_approval**.

5.2.4.14 approval_history_has_at_least_one_member

The **approval_history_has_at_least_one_member** rule specifies that each **group** with class id 'approval history' is used by exactly one **applied_group_assignment**.

EXPRESS specification:

```
*)
RULE
approval_history_has_at_least_one_member
FOR (group, applied_group_assignment);
  LOCAL
    t1_set  : SET OF GROUP := [];
    t2_set  : SET OF APPLIED_GROUP_ASSIGNMENT := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  T1_set := QUERY(i <* group | VALUE_IN(WHICH_CLASS(i),
  'approval history'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    t2_set := QUERY(a <* APPLIED_GROUP_ASSIGNMENT |
      a.ASSIGNED_GROUP = t1_set[i]);
    violate := NOT(SIZEOF(t2_set) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.

Formal propositions:

WR1: Every instance of **group** that has an **applied_classification_assignment** whose **assigned_-classification** is a **group** with attribute **name** equals 'approval history' is the **assigned_group** in exactly one instance of **applied_group_assignment**.

5.2.4.15 approvals_references_approval_history

The **approvals_references_approval_history** rule specifies that each instance of type **group** with class 'approval history' is only to be referenced by assignments of type **applied_group_assignment** with role 'approvals' via attribute **assigned_group**.

EXPRESS specification

```
*)
RULE  approvals_references_approval_history
FOR(object_role, applied_group_assignment, group);
  LOCAL
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* group | VALUE_IN(WHICH_CLASS(a), 'approval
history'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_group_assignment |
       NOT ((b.assigned_group = t1_set[i]) AND (b.role.name =
'approvals')));
    violate := SIZEOF(a_set) > 0;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities
object_role: the set of all instances of **object_role** entities.

Formal propositions:

WR1: Every instance of **group** that has an **applied_classification_assignment** whose **assigned_classification** attribute identifies an entity with attribute **name** equals 'approval history' is only to be referenced by instances of type **applied_group_assignment** whose role equals 'approvals' through attribute **assigned_group**.

5.2.4.16 assembly_relationship_has_product_definition

The **assembly_relationship_has_product_definition** rule specifies the relating and the related side of a relationship of type **product_definition_relationship** to be of the same class if **product_-definition_relationship** is of class 'assembly relationship'.

EXPRESS specification:

```
*)
RULE
assembly_relationship_has_product_definition
FOR (group, applied_classification_assignment);
  LOCAL
    cl_ass      : SET OF applied_classification_assignment := [];
    clfied_inst : SET OF product_definition_RELATIONSHIP := [];
  END_LOCAL;
  cl_ass := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME = 'assembly relationship');
  REPEAT i := 1 TO HIINDEX(cl_ass);
    clfied_inst := clfied_inst + cl_ass[i].items;
  END_REPEAT;
  WHERE
    WR1:  SIZEOF(QUERY(pdr <* clfied_inst |
          (NOT(SIZEOF(WHICH_CLASS(pdr.relating_product_definition)) >0))
          )) -
          SIZEOF(QUERY(pdr <* clfied_inst |
         (NOT(SIZEOF(WHICH_CLASS(pdr.related_product_definition)) >0))
         ))
         =0;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.
group: the set of all instances of **group** entities.

Formal propositions:

WR1: For every entity instance of **product_definition_relationship** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'assembly relationship' the instances referenced by the attributes **relating_product_-definition** and **related_product_definition** is referenced by **applied_classification_assignments** whose **assigned_classifications** have identical values in their **name** attributes.

5.2.4.17 author_for_change_plan

The **author_for_change_plan** rule specifies that an instance of **action_request_solution** with **role** 'author' is referenced by exactly one assignment instances of type **applied_person_and_-organization_assignment** that have the **role** 'author'.

EXPRESS specification:

```
*)
RULE
author_for_change_plan
FOR(applied_person_and_organization_assignment,
person_and_organization_role, action_request_solution);
  LOCAL
    t1_set  : SET OF action_request_solution := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
    violate : LOGICAL := FALSE;
```

```
  END_LOCAL;
    t1_set := QUERY(a <* action_request_solution |
              VALUE_IN(WHICH_CLASS(a), 'change plan'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_person_and_organization_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'author')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_person_and_organization_assignment: the set of all instances of **applied_person_and_-organization_assignment** entities.
action_request_solution: the set of all instances of **action_request_solution** entities
person_and_organization_role: the set of all instances of **person_and_organization_role** entities

Formal propositions:
WR1: Every instance of **action_request_solution** whose **role** equals 'author' is referenced by exactly one instance of **applied_person_and_organization_assignment** whose **role** equals 'author'.

5.2.4.18 author_for_change_realisation

The **author_for_change_realisation** rule specifies that an instance of **executed_action** with **role** 'author' is referenced by exactly one assignment instances of type **applied_person_and_-organization_assignment** that have the **role** 'author'.

EXPRESS specification:

```
*)
RULE
author_for_change_realisation
FOR(applied_person_and_organization_assignment,
  person_and_organization_role, executed_action);
  LOCAL
    t1_set  : SET OF executed_action := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* executed_action |
            VALUE_IN(WHICH_CLASS(a), 'change realisation'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_person_and_organization_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'author')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```


Argument definitions:

applied_person_and_organization_assignment: the set of all instances of **applied_person_and_-organization_assignment** entities.
executed_action: the set of all instances of **executed_action** entities
person_and_organization_role: the set of all instances of **person_and_organization_role** entities

Formal propositions:

WR1: Every instance of **executed_action** whose **role** equals 'author' is referenced by exactly one instance of **applied_person_and_organization_assignment** whose **role** equals 'author'.

5.2.4.19 author_for_change_request

The **author_for_change_request** rule specifies that an instance of **versioned_action_request** with **role** 'author' is referenced by exactly one assignment instances of type **applied_person_and_-organization_assignment** that have the **role** 'author'.

EXPRESS specification:
```
*)
RULE
author_for_change_request
FOR(applied_person_and_organization_assignment,
  person_and_organization_role, versioned_action_request);
  LOCAL
    t1_set  : SET OF versioned_action_request := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
   t1_set := QUERY(a <* versioned_action_request |
             VALUE_IN(WHICH_CLASS(a), 'change request'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_person_and_organization_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'author')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_person_and_organization_assignment: the set of all instances of **applied_person_and_-organization_assignment** entities.
versioned_action_request: the set of all instances of **versioned_action_request** entities
person_and_organization_role: the set of all instances of **person_and_organization_role** entities

Formal propositions:

WR1: Every instance of **versioned_action_request** whose **role** equals 'author' is referenced by exactly one instance of **applied_person_and_organization_assignment** whose **role** equals 'author'.

5.2.4.20 caused_by_for_check

The **caused_by_for_check** rule specifies that an instance of **action** with a value 'caused by' is referenced by exactly one assignment instances of type **applied_person_and_organization_-assignment** that have the **role** that references **organization_role** with attribute **name** of value 'caused by'.

EXPRESS specification:

```
*)
RULE
caused_by_for_check
FOR(applied_person_and_organization_assignment,
person_and_organization_role, action);
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
    t1_set := QUERY(a <* action | VALUE_IN(WHICH_CLASS(a), 'check'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
  a_set := QUERY(b <* applied_person_and_organization_assignment |
              (VALUE_IN(b.items, t1_set[i]) AND
              (b.role.name = 'caused by')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_person_and_organization_assignment: the set of all instances of **applied_person_and_-organization_assignment** entities.
action: the set of all instances of **action** entities
person_and_organization_role: the set of all instances of **person_and_organization_role** entities

Formal propositions:

WR1: Every instance of **action** whose **role** equals 'caused by' is referenced by exactly one instance of **applied_person_and_organization_assignment** whose **role** equals 'caused by'.

5.2.4.21 caused_by_for_envisaged_version_creation

The **caused_by_for_envisaged_version_creation** rule specifies that an instance of **action** with **role** 'caused by' is referenced by exactly one assignment instances of type **applied_person_and_-organization_assignment** that have the **role** 'caused by'.

EXPRESS specification:

```
*)
RULE
caused_by_for_envisaged_version_creation
FOR(applied_person_and_organization_assignment,
person_and_organization_role, action);
```



```
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
   t1_set := QUERY(a <* action |
             VALUE_IN(WHICH_CLASS(a), 'envisaged version creation'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_person_and_organization_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'caused by')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_person_and_organization_assignment: the set of all instances of **applied_person_and_-organization_assignment** entities.
action: the set of all instances of **action** entities
person_and_organization_role: the set of all instances of **person_and_organization_role** entities.

Formal propositions:

WR1: Every instance of **action** whose **role** equals 'caused by' is referenced by exactly one instance of **applied_person_and_organization_assignment** whose **role** equals 'caused by'.

5.2.4.22 caused_by_for_version_creation

The **caused_by_for_version_creation** rule specifies that an instance of **action** with **role** 'caused by' is referenced by exactly one assignment instances of type **applied_person_and_organization_-assignment** that have the **role** 'caused by'.

EXPRESS specification:

```
*)
RULE
caused_by_for_version_creation
FOR(applied_person_and_organization_assignment,
  person_and_organization_role, action);
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
   t1_set := QUERY(a <* action |
             VALUE_IN(WHICH_CLASS(a), 'version creation'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_person_and_organization_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'caused by')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
```

```
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_person_and_organization_assignment: the set of all instances of **applied_person_and_-organization_assignment** entities.
action: the set of all instances of **action** entities
person_and_organization_role: the set of all instances of **person_and_organization_role** entities.

Formal propositions:

WR1: Every instance of **action** whose **role** equals 'caused by' is referenced by exactly one instance of **applied_person_and_organization_assignment** whose **role** equals 'caused by'.

5.2.4.23 caused_by_for_version_deletion

The **caused_by_for_version_deletion** rule specifies that an instance of **action** with **role** 'caused by' is referenced by exactly one assignment instances of type **applied_person_and_organization_-assignment** that have the **role** 'caused by'.

EXPRESS specification:

```
*)
RULE
caused_by_for_version_deletion
FOR(applied_person_and_organization_assignment,
  person_and_organization_role, action);
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
   t1_set := QUERY(a <* action |
             VALUE_IN(WHICH_CLASS(a), 'version deletion'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_person_and_organization_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'caused by')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_person_and_organization_assignment: the set of all instances of **applied_person_and_-organization_assignment** entities.
person_and_organization_role: the set of all instances of **person_and_organization_role** entities.
action: the set of all instances of **action** entities

Formal propositions:

WR1: Every instance of **action** whose **role** equals 'caused by' is referenced by exactly one instance of **applied_person_and_organization_assignment** whose **role** equals 'caused by'.

5.2.4.24 caused_by_for_version_modification

The **caused_by_for_version_modification** rule specifies that an instance of **action** with **role** 'caused by' is referenced by exactly one assignment instances of type **applied_person_and_organization_-assignment** that have the **role** 'caused by'.

EXPRESS specification:

```
*)
RULE
caused_by_for_version_modification
FOR(applied_person_and_organization_assignment,
  person_and_organization_role, action);
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* action |
            VALUE_IN(WHICH_CLASS(a), 'version modification'));

  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_person_and_organization_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'caused by')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_person_and_organization_assignment: the set of all instances of **applied_person_and_-organization_assignment** entities.
action: the set of all instances of **action** entities
person_and_organization_role: the set of all instances of **person_and_organization_role** entities.

Formal propositions:

WR1: Every instance of **action** whose **role** equals 'caused by' is referenced by exactly one instance of **applied_person_and_organization_assignment** whose **role** equals 'caused by'.

5.2.4.25 caused_when_for_check

The **caused_when_for_check** rule specifies that an instance of **action** with **role** 'caused when ' is referenced by exactly one assignment instances of type **applied_date_and_time_assignment** that have the **role** 'caused when'.

EXPRESS specification:

```
*)
RULE
caused_when_for_check
FOR(applied_date_and_time_assignment, date_time_role, action);
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_date_and_time_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
   t1_set := QUERY(a <* action | VALUE_IN(WHICH_CLASS(a), 'check'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_date_and_time_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'caused when')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_date_and_time_assignment: the set of all instances of **applied_date_and_time_-assignment** entities.

date_and_time_role: the set of all instances of **date_and_time_role** entities.

action: the set of all instances of **action** entities

Formal propositions:

WR1: Every instance of **action** whose **role** equals 'caused when' is referenced by exactly one instance of **applied_date_and_time_assignment** whose **role** equals 'caused when'.

5.2.4.26 caused_when_for_envisaged_version_creation

The **caused_when_for_envisaged_version_creation** rule specifies that an instance of **action** with **role** 'caused when' is referenced by exactly one assignment instances of type **applied_date_and_-time_assignment** that have the **role** 'caused when'.

EXPRESS specification:

```
*)
RULE
caused_when_for_envisaged_version_creation
FOR(applied_date_and_time_assignment, date_time_role, action);
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_date_and_time_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* action | VALUE_IN(WHICH_CLASS(a),
             'envisaged version creation'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
```



```
    a_set := QUERY(b <* applied_date_and_time_assignment |
                (VALUE_IN(b.items, t1_set[i]) AND
                (b.role.name = 'caused when')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_date_and_time_assignment: the set of all instances of **applied_date_and_time_-assignment** entities.
date_and_time_role: the set of all instances of **date_and_time_role** entities.
action: the set of all instances of **action** entities

Formal propositions:

WR1: Every instance of **action** whose role equals 'caused when' is referenced by exactly one instance of **applied_date_and_time_assignment** whose **role** equals 'caused when'.

5.2.4.27 caused_when_for_version_creation

The **caused_when_for_version_creation** rule specifies that an instance of **action** with **role** 'caused when' is referenced by exactly one assignment instances of type **applied_date_and_time_-assignment** that have the **role** 'caused when'.

EXPRESS specification:

```
*)
RULE
caused_when_for_version_creation
FOR(applied_date_and_time_assignment, date_time_role, action);
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_date_and_time_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* action |
            VALUE_IN(WHICH_CLASS(a), 'version creation'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_date_and_time_assignment |
                (VALUE_IN(b.items, t1_set[i]) AND
                (b.role.name = 'caused when')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_date_and_time_assignment: the set of all instances of **applied_date_and_time_-assignment** entities.
date_and_time_role: the set of all instances of **date_and_time_role** entities.
action: the set of all instances of **action** entities

Formal propositions:

WR1: Every instance of **action** whose **role** equals 'caused when' is referenced by exactly one instance of **applied_date_and_time_assignment** whose **role** equals 'caused when'.

5.2.4.28 caused_when_for_version_deletion

The **caused_when_for_version_deletion** rule specifies that an instance of **action** with **role** 'caused when' is referenced by exactly one assignment instances of type **applied_date_and_time_-assignment** that have the **role** 'caused when'.

EXPRESS specification:

```
*)
RULE
caused_when_for_version_deletion
FOR(applied_date_and_time_assignment, date_time_role, action);
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_date_and_time_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
   t1_set := QUERY(a <* action |
             VALUE_IN(WHICH_CLASS(a), 'version deletion'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_date_and_time_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'caused when')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_date_and_time_assignment: the set of all instances of **applied_date_and_time_-assignment** entities.
date_and_time_role: the set of all instances of **date_and_time_role** entities.
action: the set of all instances of **action** entities

Formal propositions:

WR1: Every instance of **action** whose **role** equals 'caused when' is referenced by exactly one instance of **applied_date_and_time_assignment** whose **role** equals 'caused when'.

5.2.4.29 caused_when_for_version_modification

The **caused_when_for_version_modification** rule specifies that an instance of **action** with **role** 'caused when' is referenced by exactly one assignment instances of type **applied_date_and_time_-assignment** that have the **role** 'caused when'.

EXPRESS specification:

```
*)
(* < applied_date_and_time_assignment, 'caused when', 1, action, 'version
modification'> *)
RULE
caused_when_for_version_modification
FOR(applied_date_and_time_assignment, date_time_role, action);
  LOCAL
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_date_and_time_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* action |
            VALUE_IN(WHICH_CLASS(a), 'version modification'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_date_and_time_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'caused when')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_date_and_time_assignment: the set of all instances of **applied_date_and_time_-assignment** entities.
action: the set of all instances of **action** entities

Formal propositions:

WR1: Every instance of **action** whose **role** equals 'caused when' is referenced by exactly one instance of **applied_date_and_time_assignment** whose **role** equals 'caused when'.

5.2.4.30 centre_location_compound_representation_has_specified_name

The **centre_location_compound_representation_has_specified_name** rule specifies the **item_element** attribute of a **compound_representation_item** with the class id 'centre location' to have in the **list_representation_item** for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list.

EXPRESS specification:

```
*)
RULE centre_location_compound_representation_has_specified_name
FOR (compound_representation_item, applied_classification_assignment);
  LOCAL
```

```
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF Compound_representation_item := [];
    t2_set    : SET OF representation_item := [];
    arg_list  : LIST OF STRING := ['longitudinal location', 'transversal
     location', 'vertical location'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'centre location'
 *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'centre location');
  (* get all instances of compound_representation_item that have class id
 'centre location' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* iterate over all compound_representation_item found above; stop, if
one of them has not exactly one rep_item for each name in the arg_list
  *)
  REPEAT i:=1 TO HIINDEX(t1_set) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        t2_set := t1_set[i].item_element;
   violation := (SIZEOF(QUERY(items <* t2_set | items.name = arg_list[j]))
  <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

Compound_representation_item: the set of all instances of **compound_representation_item** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **compound_representation_item** that has an **applied_classification_-assignment** whose attribute **assigned_classification.name** equals 'centre location' is to have its attribute **item_element** instantiated as a **list_representation_item** which for each value out of 'longitudinal location', 'transversal location', 'vertical location' collect exactly one instance of **representation_item** whose **name** attribute equals that value.

5.2.4.31 change_impact_with_versionable_object_change_event

The **change_impact_with_versionable_object_change_event** rule specifies that each assignment of type **applied_action_request_assignment** with **role** 'change impact' refer to at least one element with class 'versionable object change event' through its attribute **items**.

EXPRESS specification:

```
*)
RULE
change_impact_with_versionable_object_change_event
FOR(applied_action_request_assignment, object_role, action);
  LOCAL
    t1_set  : SET OF applied_action_request_assignment := [];
    a_set   : SET OF action := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(b <* applied_action_request_assignment |
                  (b.role.name= 'change impact'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* t1_set[i].items |
          ('SHIP_STRUCTURES_SCHEMA.ACTION' IN TYPEOF(b)) AND
             VALUE_IN(WHICH_CLASS(b), 'versionable object change event'));
    violate := SIZEOF(a_set) = 0;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_action_request_assignment: the set of all instances of **applied_action_request_-assignment** entities.
object_role: the set of all instances of **object_role** entities.
action: the set of all instances of **action** entities.

Formal propositions:

WR1: Every instance of **applied_action_request_assignment** whose **role** equals 'change impact' is to reference at least one instance that has an **applied_classification_assignment** whose **assigned_-classification** is a **group** with attribute **name** equals 'versionable object change event' through its attribute **items**.

5.2.4.32 change_plan_has_mandatory_attribute_description

The **change_plan_has_mandatory_attribute_description** rule specifies that for an instance of **action_request_solution** with class id 'change plan' the optional attribute **description** is instantiated.

EXPRESS specification:

```
*)
RULE
change_plan_has_mandatory_attribute_description
FOR (action_request_solution);
  LOCAL
    t1_set  : SET OF action_request_solution := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* action_request_solution |
  VALUE_IN(WHICH_CLASS(i), 'change plan'));
  violate := (SIZEOF(QUERY(k <* t1_set |
  NOT EXISTS (k.description))) > 0);
```

```
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

action_request_solution: the set of all instances of **action_request_solution** entities.

Formal propositions:

WR1: The optional attribute **description** exists for every instance of **action_request_solution** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'change plan'.

5.2.4.33 class_and_statutory_designation_has_properties

The **class_and_statutory_designation_has_properties** rule specifies that a **product_definition** with a class id 'class and statutory designation' is referenced by one **property_definition_-representation** with the attribute **name** equal to 'class and statutory designation' via a **property_-definition**.

EXPRESS specification:

```
*)
RULE class_and_statutory_designation_has_properties
FOR (product_definition,
  property_definition_representation,
  group, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    t3_set    : SET OF property_definition := [];
    t4_set    : SET OF product_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'class and statutory designation');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  t2_set:= QUERY(i <* PROPERTY_DEFINITION_REPRESENTATION |
                   i.NAME = 'class and statutory designation');
  REPEAT i := 1 TO HIINDEX(t2_set);
    t3_set := t3_set + t2_set[i].definition;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t3_set);
    t4_set := t4_set + t3_set[i].definition;
  END_REPEAT;
  violation := t1_set <> t4_set;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definitiont: the set of all instances of **product_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equals 'class and statutory designation' is referenced by exactly one instance of **property_definition** through attribute **definition** that in turn is referenced by an instance of **property_definition_representation** through attribute **definition** whose attribute **name** equals 'class and statutory designation'.

5.2.4.34 class_notation_with_named_representation_items

The **class_notation_with_named_representation_items** rule specifies the **items** attribute of a **representation** is to have for each entry in the list one or more **representation_items** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'class notation'.

EXPRESS specification:

```
*)
RULE class_notation_with_named_representation_items
FOR (representation, representation_item,
property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['class notations hull',
                                   'class notations machinery'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
             'USED_REPRESENTATION'))
                          | (temp_prop_def_rep.name = 'class notation')
                       )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
```

```
                    rep_item.name = arg_list[j])) < 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every instance of **representation** that is **the used_representation** in an instance of **property_definition_representation** whose attribute **name** equals 'class notation' for each value out of either 'class notations hull', or 'class notations machinery' collect at least one instance of **representation_item** in its attribute items whose **name** attribute equals that value.

5.2.4.35 class_parameters_has_properties

The **class_parameters_has_properties** rule specifies that a **product_definition** with a class id 'class parameters' is referenced by one **property_definition_representation** with the **name** 'class. parameters' via a **property_definition**.

EXPRESS specification:

```
*)
RULE class_parameters_has_properties
FOR (product_definition,
  property_definition_representation,
  group, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    t3_set    : SET OF property_definition := [];
    t4_set    : SET OF product_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'class parameters');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  t2_set:= QUERY(i <* PROPERTY_DEFINITION_REPRESENTATION |
                   i.NAME = 'class parameters');
  REPEAT i := 1 TO HIINDEX(t2_set);
    t3_set := t3_set + t2_set[i].definition;
  END_REPEAT;
```



```
  REPEAT i := 1 TO HIINDEX(t3_set);
    t4_set := t4_set + t3_set[i].definition;
  END_REPEAT;
  violation := t1_set <> t4_set;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equals 'class parameters' is referenced by exactly one instance of **property_definition** through attribute **definition** that in turn is referenced by an instance of **property_definition_representation** through attribute **definition** whose attribute **name** equals 'class parameters'.

5.2.4.36 corrugated_structure_design_definition_has_at_least_one_class

The **corrugated_structure_design_definition_has_at_least_one_class** rule specifies that an instance of **product_definition_shape** with class id 'corrugated structure design definition' is referenced by at least one instance of **shape_aspect** with class id 'boundary' via **of_shape**.

EXPRESS specification:

```
*)
RULE
corrugated_structure_design_definition_has_at_least_one_class
FOR(product_definition_shape,
  shape_aspect,
  group, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET Of shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'corrugated structure design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
```

```
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set |
      'boundary' IN WHICH_CLASS(t2_inst))) < 1;
END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** that has an **assigned_classification** with attribute **name** equals 'corrugated structure design definition' is referenced by one or more instances of **shape_aspect** that has an **applied_-classification_assignment** which has an **assigned_classification** with attribute **name** equals 'boundary' through attribute **of_shape**.

5.2.4.37 corrugated_structure_design_definition_has_at_most_one_class

The **corrugated_structure_design_definition_has_at_most_one_class** rule specifies that an instance of **product_definition_shape** with class id 'corrugated structure design definition' is referenced by at most one instance of **shape_aspect** with class id 'moulded surface' via **of_shape**.

EXPRESS specification:

```
*)
RULE
corrugated_structure_design_definition_has_at_most_one_class
FOR(product_definition_shape,
  shape_aspect,
  group, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET Of shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME =
                   'corrugated structure design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
```



```
    violation := SIZEOF(QUERY(t2_inst <* t2_set | 'moulded surface'
      IN WHICH_CLASS(t2_inst))) > 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** that has an **assigned_classification** with **name** equals 'corrugated structure design definition' is referenced by one or less instances of **shape_aspect** that have an **applied_classification_assignment** which has an **assigned_classification** with **name** attribute equals 'moulded surface' through attribute **of_shape**.

5.2.4.38 date_time_for_change_realisation

The **date_time_for_change_realisation** rule specifies that an instance of **executed_action** with **role** 'date time' is referenced by exactly one assignment instances of type **applied_date_and_time_-assignment** that have the **role** 'date time'.

EXPRESS specification:

```
*)
RULE
date_time_for_change_realisation
FOR(applied_date_and_time_assignment, date_time_role, executed_action);
  LOCAL
    t1_set  : SET OF executed_action := [];
    a_set   : SET OF applied_date_and_time_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* executed_action |
            VALUE_IN(WHICH_CLASS(a), 'change realisation'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_date_and_time_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'date time')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_date_and_time_assignment: the set of all instances of **applied_date_and_time_-assignment** entities.
date_time_role: the set of all instances of **date_time_role** entities.
executed_action: the set of all instances of **executed_action** entities

Formal propositions:

WR1: Every instance of **executed_action** whose **role** equals 'date time' is referenced by exactly one instance of **applied_date_and_time_assignment** whose **role** equals 'date time'.

5.2.4.39 date_time_for_change_request

The **date_time_for_change_request** rule specifies that an instance of **versioned_action_request** with **role** 'date time' is referenced by exactly one assignment instances of type **applied_date_and_-time_assignment** that have the **role** 'date time'.

EXPRESS specification:

```
*)
RULE
date_time_for_change_request
FOR(applied_date_and_time_assignment, date_time_role,
versioned_action_request);
  LOCAL
    t1_set  : SET OF versioned_action_request := [];
    a_set   : SET OF applied_date_and_time_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
   t1_set := QUERY(a <* versioned_action_request |
             VALUE_IN(WHICH_CLASS(a), 'change request'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_date_and_time_assignment |
               (VALUE_IN(b.items, t1_set[i]) AND
               (b.role.name = 'date time')));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_date_and_time_assignment: the set of all instances of **applied_date_and_time_-assignment** entities.
date_time_role: the set of all instances of **date_time_role** entities.
versioned_action_request: the set of all instances of **versioned_action_request** entities.

Formal propositions:

WR1: Every instance of **versioned_action_request** whose **role** equals 'date time' is referenced by exactly one instance of **applied_date_and_time_assignment** whose **role** equals 'date time'.

5.2.4.40 define_maximum_permissible_still_water_bending_moment_-values

The **define_maximum_permissible_still_water_bending_moment_values** rule specifies that an instance of **product_definition** with class id 'maximum permissible still water bending moment' is referenced by exactly one instance of **property_definition** through the attribute **definition** whose attribute **name** has the value 'maximum permissible still water bending moment values'.

EXPRESS specification:

```
*)
RULE
 define_maximum_permissible_still_water_bending_moment_values
FOR(product_definition,
    property_definition_representation,
    applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with
     id 'maximum permissible still water bending moment' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME =
                  'maximum permissible still water bending moment');
  (* get all instances of T1 that have class
     id 'maximum permissible still water bending moment' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition in t1_set:
     get the property_definition_representation instances
     that are referencing a product_definition instance via
     definition, filter out those property_definition_representation
     instances whose attribute name has the value
     'maximum permissible still water bending  moment values'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
    'maximum permissible still water bending  moment values')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'maximum permissible still water bending moment', is referenced by exactly one **property_definition.definition**, where **property_definition.name** equals 'maximum permissible still water bending moment values'.

5.2.4.41 definition_for_design_still_water_bending_moment_parameters

The **definition_for_design_still_water_bending_moment_parameters** rule specifies that an instance of **product_definition** with class id 'design still water bending moment' is referenced by exactly one instance of **property_definition** through the attribute **definition** whose attribute **name** has the value 'design still water bending moment parameters'.

EXPRESS specification:

```
*)
RULE
 definition_for_design_still_water_bending_moment_parameters
FOR(product_definition,
    property_definition_representation,
    applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id
     'design still water bending moment' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME =
                   'design still water bending moment');

  (* get all instances of T1 that have class
     id 'design still water bending moment' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition in t1_set:
     get the property_definition_representation instances
     that are referencing a product_definition instance via definition,
     filter out those property_definition_representation instances
     whose attribute name has the value 'design still water
     bending moment parameters'  check if their number equals 1  *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
    'design still water bending moment parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with name equals 'design still water bending moment', is referenced by exactly one **property_definition.definition**, where **property_definition.name** equals 'design still water bending moment parameters'.

5.2.4.42 definition_for_design_vertical_wave_bending_moment_-parameters

The **definition_for_design_vertical_wave_bending_moment_parameters** rule specifies that an instance of **product_definition** with class id 'design vertical wave bending moment' is referenced by exactly one instance of **property_definition** through the attribute **definition** whose attribute **name** has the value 'design vertical wave bending moment parameters'.

EXPRESS specification:

```
*)
RULE
 definition_for_design_vertical_wave_bending_moment_parameters
FOR(product_definition,
    property_definition_representation,
    applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with
     id 'design vertical wave bending moment' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME =
                   'design vertical wave bending moment');
  (* get all instances of T1 that have class
     id 'design vertical wave bending moment' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition in t1_set:
     get the property_definition_representation instances
     that are referencing a product_definition instance via
     definition, filter out those property_definition_representation
     instances whose attribute name has the value 'design vertical
     wave bending moment parameters' check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
```



```
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
    'design vertical wave bending moment parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'design vertical wave bending moment', is referenced by exactly one **property_definition.definition**, where **property_definition.name** equals 'design vertical wave bending moment parameters'.

5.2.4.43 definition_for_design_vertical_wave_shear_force_parameters

The **definition_for_design_vertical_wave_shear_force_parameters** rule specifies that an instance of **product_definition** with class id 'design vertical wave shear force' is referenced by exactly one instance of **property_definition** through the attribute **definition** whose attribute **name** has the value 'design vertical wave shear force parameters'.

EXPRESS specification:

```
*)
RULE
 definition_for_design_vertical_wave_shear_force_parameters
FOR(product_definition,
    property_definition_representation,
    applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id
     'design vertical wave shear force' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME =
                   'design vertical wave shear force');
  (* get all instances of T1 that have class
     id 'design vertical wave shear force' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
```



```
  END_REPEAT;
  (* for all instances of product_definition in t1_set:
     get the property_definition_representation instances
     that are referencing a product_definition instance via
     definition, filter out those property_definition_representation
     instances whose attribute name has the value
     'design vertical wave shear force parameters'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
    'design vertical wave shear force parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equals ‘design vertical wave shear force’, is referenced by exactly one **property_definition.definition**, where **property_definition.name** equals ' design vertical wave shear force parameters'.

5.2.4.44 define_maximum_permissible_still_water_shear_force_values

The **define_maximum_permissible_still_water_shear_force_values** rule specifies that an instance of **product_definition** with class id 'maximum permissible still water shear force' is referenced by exactly one instance of **property_definition** through the attribute **definition** whose attribute **name** has the value 'maximum permissible still water shear force values'.

EXPRESS specification:

```
*)
RULE
define_maximum_permissible_still_water_shear_force_values
FOR(product_definition,
    property_definition_representation,
    applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with
     id 'maximum permissible still water shear force' *)
```

```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME =
                   'maximum permissible still water shear force');
  (* get all instances of T1 that have class
     id 'maximum permissible still water shear force' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition in t1_set:
     get the property_definition_representation instances
     that are referencing a product_definition instance via
     definition, filter out those property_definition_representation
     instances whose attribute name has the value
     'maximum permissible still water shear force values'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
    'maximum permissible still water shear force values')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equals ‘maximum permissible still water shear force’, is referenced by exactly one **property_-definition.definition**, where **property_definition.name** equals ‘maximum permissible still water shear force values’.

5.2.4.45 document_has_at_least_one_references

The **document_has_at_least_one_references** rule specifies that an instance of **document_for** id 'document' is referenced by at least one instance of **document_representation_type** via **represented_document**.

EXPRESS specification:

```
*)
RULE document_has_at_least_one_references
FOR(document, document_representation_type);
  LOCAL
    t1_set  : SET OF document := [];
    t2_set  : SET OF document_representation_type := [];
    violate : LOGICAL := FALSE;
```



```
  END_LOCAL;
  t1_set := QUERY(i <* document | VALUE_IN(WHICH_CLASS(i),
                                  'document'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.DOCUMENT_REPRESENTATION_TYPE.'+
      'REPRESENTED_DOCUMENT'));
    violate := SIZEOF(t2_set) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

document: the set of all instances of **document** entities.
document_representation_type: the set of all instances of **document_representation_type** entities.

Formal propositions:

WR1: Every instance of **document** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'document' is referenced by one or more instances of **document_representation_type** through attribute **represented_document**.

5.2.4.46 document_has_exactly_one_author

The **document_has_exactly_one_author** rule specifies that each instance of type **document** is referenced by exactly one instance of type **applied_person_assignment** or **applied_organization_-assignment** or **applied_person_and_organization_assignment**, where the latter instance has the **role** 'author'.

EXPRESS specification:

```
*)
RULE document_has_exactly_one_author
FOR(document);
  LOCAL
    bag_1   : BAG OF applied_person_assignment := [];
    bag_2   : BAG OF applied_person_and_organization_assignment := [];
    bag_3   : BAG OF applied_organization_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  REPEAT i := 1 TO SIZEOF(document) WHILE (NOT violate);
    bag_1 := USEDIN(document[i], 'SHIP_STRUCTURES_SCHEMA.' +
         'APPLIED_PERSON_ASSIGNMENT.ITEMS');
    bag_2 := USEDIN(document[i], 'SHIP_STRUCTURES_SCHEMA.' +
         'APPLIED_PERSON_AND_ORGANIZATION_ASSIGNMENT.ITEMS');
    bag_3 := USEDIN(document[i], 'SHIP_STRUCTURES_SCHEMA.' +
         'APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS');
    violate := NOT ((SIZEOF( bag_1 ) + SIZEOF( bag_2 )+ SIZEOF( bag_3 ))=
  1);
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

document: the set of all instances of **document** entities.

Formal propositions:

WR1: Every instance of **document** is referenced by exactly one instance of **applied_person_-assignment** or **applied_person_and_organization_assignment** whose role equals 'author'.

5.2.4.47 document_reference_has_at_least_one_references

The **document_reference_has_at_least_one_references** rule specifies that an instance of **document** with class id 'document reference' is referenced by at least one instance of **document_usage_-constraint** via **source**.

EXPRESS specification:
```
*)
RULE document_reference_has_at_least_one_references
FOR(document, document_usage_constraint);
  LOCAL
    t1_set  : SET OF document := [];
    t2_set  : SET OF document_usage_constraint := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* document | VALUE_IN(WHICH_CLASS(i),
                                  'document reference'));

  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.DOCUMENT_USAGE_CONSTRAINT.SOURCE'));
    violate := SIZEOF(t2_set) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

document: the set of all instances of **document** entities.
document_usage_constraint: the set of all instances of **document_usage_constraint** entities.

Formal propositions:

WR1: Every instance of **document** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'document reference' is referenced by one or more instances of **document_usage_constraint** through attribute **source**.

5.2.4.48 document_reference_reference_with_address_with_references

The **document_reference_reference_with_address_with_references** rule specifies that an instance of **document** with class id 'document reference reference with address' is referenced by at least one instance of **document_usage_constraint** via **source**.

EXPRESS specification:

```
*)
RULE
document_reference_reference_with_address_with_references
FOR(document, document_usage_constraint);
  LOCAL
    t1_set:  SET OF document := [];
    t2_set:  SET OF document_usage_constraint := [];
    violate: LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* document | VALUE_IN(WHICH_CLASS(i),
            'document reference reference with address'));

  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
          'SHIP_STRUCTURES_SCHEMA.DOCUMENT_USAGE_CONSTRAINT.SOURCE'));
    violate := SIZEOF(t2_set) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

document: the set of all instances of **document** entities.
document_usage_constraint: the set of all instances of document_usage_constraint entities.

Formal propositions:

WR1: Every instance of **document** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'document reference reference with address' is referenced by one or more instances of **document_usage_constraint** through attribute **source**.

5.2.4.49 document_reference_with_address_has_at_least_one_references

The **document_reference_with_address_has_at_least_one_references** rule specifies that an instance of **document** with class id 'document reference with address' is referenced by at least one instance of **applied_external_identification_assignments** via **items**.

EXPRESS specification:

```
*)
RULE document_reference_with_address_has_at_least_one_references
FOR(document,
    applied_external_identification_assignment);
  LOCAL
    t1_set  : SET OF document := [];
    t2_set  : SET OF applied_external_identification_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* document | VALUE_IN(WHICH_CLASS(i),
                                  'document reference with address'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
```

```
    t2_set := bag_to_set(USEDIN(t1_set[i],
'SHIP_STRUCTURES_SCHEMA.APPLIED_EXTERNAL_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violate := SIZEOF(t2_set) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

document: the set of all instances of **document** entities.
applied_external_identification_assignments: the set of all instances of **applied_external_-identification_assignments** entities.

Formal propositions:

WR1: Every instance of **document** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'document reference with address' is referenced by one or more instances of **applied_external_identification_assignments** through attribute **items**.

5.2.4.50 envisaged_version_creation_has_mandatory_attribute_-description

The **envisaged_version_creation_has_mandatory_attribute_description** rule specifies that for an instance of **action** with class id 'envisaged version creation' the optional attribute **description** is instantiated.

EXPRESS specification:

```
*)
RULE
envisaged_version_creation_has_mandatory_attribute_description
FOR (action);
  LOCAL
    t1_set  : SET OF action := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
   t1_set := QUERY(i <* action | VALUE_IN(WHICH_CLASS(i),
   'envisaged version creation'));
   violate := (SIZEOF(QUERY(k <* t1_set |
   NOT EXISTS (k.description))) > 0);
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

action: the set of all instances of **action** entities.

Formal propositions:

WR1: The optional attribute **description** exists for every instance of **action** that has an **applied_-classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'envisaged version creation'.

5.2.4.51 executed_action_with_identification_assignment

The **executed_action_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE executed_action_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF executed_action := [];
    t2_set    : SET OF executed_action := [];
    arg_list  : LIST OF STRING := [ 'definable object'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'moulded form
 characteristics definition' *)
  REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
    c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME =  arg_LIST[j]);
  END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **property_definition** that is referenced by an **applied_classification_-assignment** whose **assigned_classification** has a **name** attribute of value 'definable object' has an **applied_identification_assignment** to define the instance identifier.

5.2.4.52 externally_defined_class_with_known_source_requirement

The **externally_defined_class_with_known_source_requirement** rule specifies that each instance of **externally_defined_class** which references as the source a **known_source** is contained in the set of items of exactly one **applied_organization_assignment** which references as its role an **organization_role** with a name of 'class supplier'. This rule enforces the requirement for every **library_reference** to have a **supplier_bsu**.

EXPRESS specification:

```
*)
RULE externally_defined_class_with_known_source_requirement FOR
(externally_defined_class);
WHERE
  WR1: SIZEOF ( QUERY ( edc <* externally_defined_class |
       ('SHIP_STRUCTURES_SCHEMA.KNOWN_SOURCE' IN TYPEOF ( edc.source ) )
    AND
       ( SIZEOF ( QUERY ( aoa <*
       USEDIN ( edc
       ,'SHIP_STRUCTURES_SCHEMA.APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS' )
       | aoa.  role.name = 'library supplier' ) ) =0 ) ) ) =0;
END_RULE;
(*
```

Argument definitions:

externally_defined_class: the set of all instances of **externally_defined_class** entities.

Formal propositions:

WR1: Each instance of **externally_defined_class** which references as the source a **known_source** is a member in the set of items of an **applied_organization_assignment**. This **organization_assignment** references as its role an **organization_role** with attribute **name** equal to 'library supplier' .

5.2.4.53 flare_area_parameters_has_exactly_two_representation_items

The **flare_area_parameters_has_exactly_two_representation_items** rule specifies the **items** attribute of a **representation** that is the **used_representation** in a **property_definition_-representation** whose **name** attribute has the value 'flare area parameters' to have exactly two **representation_item**s.

EXPRESS specification:

```
*)
RULE flare_area_parameters_has_exactly_two_representation_items
FOR (representation, property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
```



```
                  temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
      'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name = 'flare area parameters')
                 )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
      violation := SIZEOF(reps[i].items) > 2;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_representation** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: The **items** of a **representation** that is the **used_representation** in an instance of **property_definition_representation** with attribute **name** equals 'flare area parameters' collect exactly two instances of **representation_item**.

5.2.4.54 freeboard_characteristics_has_properties

The **freeboard_characteristics_has_properties** rule specifies that a **product_definition** with a class id 'freeboard characteristics' is referenced by one **property_definition_representation** with the **name** 'freeboard characteristics' via a **property_definition**.

EXPRESS specification:

```
*)
RULE freeboard_characteristics_has_properties
FOR (product_definition, property_definition,
property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    t3_set    : SET OF property_definition := [];
    t4_set    : SET OF product_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'freeboard
characteristics' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'freeboard characteristics');
  (* get all instances of product_definition that have class id 'freeboard
 characteristics' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
```

```
    END_REPEAT;
  END_REPEAT;
  (* get all property_definition_representation instances with name
'freeboard characteristics' *)
  t2_set:= QUERY(i <* PROPERTY_DEFINITION_REPRESENTATION |
                  i.NAME = 'freeboard characteristics');
  (* get all property_definition instances which are the .definition of the
  property_definition_representation *)
  REPEAT i := 1 TO HIINDEX(t2_set);
    t3_set := t3_set + t2_set[i].definition;
  END_REPEAT;
  (* get all product_definition instances which are the .definition of the
  property_definition *)
  REPEAT i := 1 TO HIINDEX(t3_set);
    t4_set := t4_set + t3_set[i].definition;
  END_REPEAT;
  (* compare both lists with product_definition instances which have to be
 identical *)
  violation := t1_set <> t4_set;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose attribute **assigned_class.name** equals 'freeboard characteristics' is referenced by exactly one instance of **property_definition** through attribute **definition** that in turn is referenced by an instance of **property_definition_representation** through attribute **definition** whose attribute **name** equals 'freeboard characteristics'.

5.2.4.55 global_axis_placement_has_properties

The **global_axis_placement_has_properties** rule specifies that a **product_definition** with a class id 'global axis placement' is referenced by one **property_definition_representation** with the **name** 'global axis placement' via a **property_definition**.

EXPRESS specification:

```
*)
RULE global_axis_placement_has_properties
FOR (property_definition_representation,
group, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
```



```
    t2_set    : SET OF property_definition_representation := [];
    t3_set    : SET OF property_definition := [];
    t4_set    : SET OF product_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME =
                  'global axis placement');
   REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  t2_set:= QUERY(i <* PROPERTY_DEFINITION_REPRESENTATION |
                  i.NAME = 'global axis placement');
  REPEAT i := 1 TO HIINDEX(t2_set);
    t3_set := t3_set + t2_set[i].definition;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t3_set);
    t4_set := t4_set + t3_set[i].definition;
  END_REPEAT;
  violation := t1_set <> t4_set;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equals 'global axis placement' is referenced by exactly one instance of **property_definition** through attribute **definition** that in turn is referenced by an instance of **property_definition_representation** through attribute **definition** whose attribute **name** equals 'global axis placement'.

5.2.4.56 global_id_is_unique

The **global_id_is_unique** rule specifies that the global identifiers of definable objects are unique.

EXPRESS specification:

```
*)
RULE global_id_is_unique
FOR (applied_identification_assignment);
  LOCAL
    set_1     : SET OF applied_identification_assignment:= [];
    bag_2     : BAG OF STRING := [];
    violation : LOGICAL := FALSE;
```



```
  END_LOCAL;

  (* get all instances of guid  *)
    set_1 := QUERY(i <* applied_identification_assignment |
                        (i.role.name = 'globally unambiguous
  identifier'));
  REPEAT i := 1 TO HIINDEX(set_1);
     bag_2 := bag_2 + [set_1[i].assigned_id];
  END_REPEAT;
  violation := SIZEOF(QUERY(i <* set_1 | NOT(i.assigned_id LIKE '*:*')) )=
  0;
  WHERE
    WR1: VALUE_UNIQUE(bag_2);
    WR2:  NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_identification_assignment: the set of all instances of **applied_identification_assignment** entities.

Formal propositions:

WR1: Every **applied_identification_assignment** has an atttribute **role** that references an **identification_role** with **name** equals 'globally unambiguous identifier', and is to have a unique value for **assigned_id**.

WR2: Attribute id of every instance of **product** that is referenced by an **applied_classification_-assignment** whose **assigned_classification** has a **name** attribute of value 'definable object' match the pattern '*.*'.

5.2.4.57 group_with_identification_assignment

The **group_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE group_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF group := [];
    t2_set    : SET OF group := [];
    arg_list  : LIST OF STRING := ['structural cutout boundary -
  relationship'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id  *)
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =  arg_LIST[j]);
   END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
```



```
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;


    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
    END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **group** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'explicit feature design definition' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.58 homogeneous_ship_material_property_has_named_references

The **homogeneous_ship_material_property_has_named_references** rule specifies that an instance of **property_definition** with class id 'homogeneous ship material property' is referenced by at most one instance of **applied_document_reference** via **items** whose attribute **role** defines an entity with an attribute **name** equals 'material reference'.

EXPRESS specification:

```
*)
RULE
homogeneous_ship_material_property_has_named_references
FOR(object_role, group, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF applied_document_reference := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'homogeneous_ship material property');
    REPEAT i := 1 TO HIINDEX(c_a_set);
       REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
                  t1_set := t1_set + c_a_set[i].items[j];
       END_REPEAT;
  END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
                  t2_set := bag_to_set(USEDIN(t1_set[i],
        'SHIP_STRUCTURES_SCHEMA.APPLIED_DOCUMENT_REFERENCE.ITEMS'));
```



```
                  violation := SIZEOF(QUERY(t2_inst <* t2_set |
        t2_inst.role.name = 'material reference')) > 1;
    END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

object_role: the set of all instances of **object_role** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities

Formal propositions:

WR1: Every instance of **property_definition** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'homogeneous_ship material property' is referenced by one or less instances of **applied_document_reference** whose attribute role.name has a value of 'material reference' through attribute **items**.

5.2.4.59 hull_cross_section_contains_specified_classes

The **hull_cross_section_contains_specified_classes** rule specifies a **product_definition** of class 'hull cross section' that also is a **group**. **group_assignment** attribute **role** identifies entities with attribute **name** equals 'equivalence', typically an **item_structure** in the ARM, which through the **group** only collect **product_definition**s of those classes that are specified by the remaining parameters ('structural system') to this rule.

EXPRESS specification:

```
*)
RULE hull_cross_section_contains_specified_classes
FOR (applied_group_assignment,group, applied_classification_assignment);
  LOCAL
    cl_ass    : SET OF applied_classification_assignment := [];
    pd        : SET OF PRODUCT_DEFINITION := [];
    gr_ass    : SET OF APPLIED_GROUP_ASSIGNMENT := [];
    groups    : SET OF GROUP := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  cl_ass := QUERY(i <* applied_classification_assignment |
                  (i.role.name = 'class membership') AND
                  (i.assigned_class.name = 'hull cross section'));
  REPEAT i := 1 TO HIINDEX(cl_ass);
    REPEAT j := 1 TO HIINDEX(cl_ass[i].items);
      pd := pd + cl_ass[i].items[j];
    END_REPEAT;
  END_REPEAT;
  gr_ass := QUERY(i <* APPLIED_GROUP_ASSIGNMENT |
                  i.role.name = 'equivalence');
  REPEAT i := 1 TO HIINDEX(gr_ass);
    REPEAT j := 1 TO HIINDEX(gr_ass[i].items);
      IF (gr_ass[i].items[j] IN pd) THEN
        groups := groups + gr_ass[i].assigned_group;
```



```
      END_IF;
    END_REPEAT;
  END_REPEAT;
  gr_ass := QUERY(i <* applied_group_assignment |
                 (SIZEOF(i.items) <> 0) AND
                 (i.role.name = 'item structure') AND
                 (i.assigned_group IN groups));

  REPEAT i := 1 TO HIINDEX(gr_ass) WHILE NOT violation;
    violation := SIZEOF(QUERY(inst <* gr_ass[i].items |
                  ('compartment' IN which_class(inst))  AND
                  ('structural system' IN which_class(inst)) )) > 0;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: For every instance of **product_definition** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'hull cross section' and that is in the **items** set of an **applied_group_assignment** whose attribute **role** identifies entities with the **name** equal to 'equivalence', the instance of **group** that is the **assigned_group** of this **applied_-group_assignment** is to only collect instances that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to one out of 'structural system', or 'compartment' in its other **applied_group_assignment** whose attribute **role** identifies entities with the **name** equal to 'item structure'.

5.2.4.60 hull_cross_section_design_definition_for_representation_items

The **hull_cross_section_design_definition_for_representation_items** rule specifies the **items** attribute of a **representation** that is the **used_representation** in a **property_definition_-representation** whose attribute **name** equals 'hull cross section design definition parameters' to have exactly one **representation_item**s.

EXPRESS specification:

```
*)
RULE
hull_cross_section_design_definition_for_representation_items
FOR (property_definition_representation, representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
```

```
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
      'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'hull cross section design definition parameters')
                 )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
      violation := SIZEOF(reps[i].items) > 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_representation** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: The **items** of a **representation** that is the **used_representation** in an instance of **property_definition_representation** whose attribute **name** equals 'hull cross section design definition parameters' is to collect exactly one instance of **representation_item**.

5.2.4.61 hull_cross_section_design_has_at_least_one_revision_with_context

The **hull_cross_section_design_has_at_least_one_revision_with_context** rule specifies that an instance of **product_definition_shape** with class id 'hull cross section design definition' is referenced by at least one **group_assignments** that have the **role** 'members' and that reference a **group** of class 'revision with context'.

EXPRESS specification:

```
*)
RULE
hull_cross_section_design_has_at_least_one_revision_with_context
FOR(group, applied_classification_assignment);
  LOCAL
    c_a_set    : SET OF applied_classification_assignment := [];
    t1_set     : SET OF product_definition_shape := [];
    gr_ass_set : SET OF applied_group_assignment := [];
    violation  : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id *)
  (*'hull cross section design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME =
                   'hull cross section design definition');
```



```
  (* get all instances of product_definition_shape that have class id*)
  (* 'hull cross section design definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    gr_ass_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS'));
    violation := SIZEOF(QUERY(aga <* gr_ass_set |
      (aga.role.name = 'members')
      AND
      ('revision with context' IN WHICH_CLASS(aga.assigned_group)))) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'hull cross section design definition' is to have one or more **applied_group_assignments** whose attribute **role** equals 'members' and whose **assigned_group** has an **applied_classification_assignment** whose **assigned_-classification** is a **group** with attribute **name** equals 'revision with context'.

5.2.4.62 hull_cross_section_design_has_at_most_one_revision_with_-context

The **hull_cross_section_design_has_at_most_one_revision_with_context** rule specifies that an instance of **product_definition_shape** with class id 'hull cross section design definition' is referenced by at most one **applied_group_assignment** that have the **role** 'members' and that reference a **group** of class 'revision with context'.

EXPRESS specification:

```
*)
RULE
hull_cross_section_design_has_at_most_one_revision_with_context
FOR(group, applied_classification_assignment);
  LOCAL
    c_a_set    : SET OF applied_classification_assignment := [];
    t1_set     : SET OF product_definition_shape := [];
    gr_ass_set : SET OF applied_group_assignment := [];
    violation  : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
```

```
                    'hull cross section design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    gr_ass_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS'));
    violation := SIZEOF(QUERY(aga <* gr_ass_set |
     (aga.role.name = 'members')
     AND
     ('revision with context' IN WHICH_CLASS(aga.assigned_group)))) > 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'hull cross section design definition' is to have one or less **applied_group_assignments** whose attribute **role** equals 'members' and whose **assigned_group** has an **applied_classification_assignment** whose **assigned_-classification** is a **group** with attribute **name** equal to 'revision with context'.

5.2.4.63 hull_cross_section_has_at_least_one_revision_with_context

The **hull_cross_section_has_at_least_one_revision_with_context** rule specifies that an instance of **product_definition** with class id 'hull cross section' is referenced by at least one **group_assignments** that have the role 'context of revision' and that reference a **group** of class 'revision with context'.

EXPRESS specification:

```
*)
RULE
hull_cross_section_has_at_least_one_revision_with_context
FOR(object_role, group, applied_classification_assignment);
  LOCAL
    c_a_set    : SET OF applied_classification_assignment := [];
    t1_set     : SET OF product_definition := [];
    gr_ass_set : SET OF applied_group_assignment := [];
    violation  : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id *)
  (*'hull cross section' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'hull cross section');
  (* get all instances of product_definition that have class id *)
```



```
  (*'hull cross section' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    gr_ass_set := bag_to_set(USEDIN(t1_set[i],
        'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS'));
    violation := SIZEOF(QUERY(aga <* gr_ass_set |
       (aga.role.name = 'context of revision')
       AND
      ('revision with context' IN WHICH_CLASS(aga.assigned_group)))) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

object_role: the set of all instances of **object_role** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'hull cross section' is to have one or more **applied_group_assignments** whose attribute **role** equals 'context of revision' and whose **assigned_group** has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'revision with context'.

5.2.4.64 hull_cross_section_optionally_contains_specified_classes

The **hull_cross_section_optionally_contains_specified_classes** rule specifies that a **product_-definition** of class 'hull cross section' that also is a group. **Group_assignment** attribute **role** identifies entities with attribute **name** equals 'equivalence', typically an **item_structure** in the ARM - if not empty - via the **group** reference at least one **product_definition** of class 'structural system'.

EXPRESS specification:

```
*)
RULE
hull_cross_section_optionally_contains_specified_classes
FOR (group, applied_classification_assignment, applied_group_assignment);
  LOCAL
    cl_ass    : SET OF applied_classification_assignment := [];
    pd        : SET OF PRODUCT_DEFINITION := [];
    gr_ass    : SET OF APPLIED_GROUP_ASSIGNMENT := [];
    groups    : SET OF GROUP := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  cl_ass := QUERY(i <* applied_classification_assignment |
                  (i.role.name = 'class membership') AND
```

```
                  (i.assigned_class.name = 'hull cross section'));
  REPEAT i := 1 TO HIINDEX(cl_ass);
    pd := pd + cl_ass[i].items;
  END_REPEAT;
  gr_ass := QUERY(i <* APPLIED_GROUP_ASSIGNMENT |
                  i.role.name = 'equivalence');
  REPEAT i := 1 TO HIINDEX(gr_ass);
    REPEAT j := 1 TO HIINDEX(gr_ass[i].items);  -- should always be just
   one
      IF (gr_ass[i].items[j] IN pd) THEN
        groups := groups + gr_ass[i].assigned_group;
      END_IF;
    END_REPEAT;
  END_REPEAT;
  gr_ass := QUERY(i <* APPLIED_GROUP_ASSIGNMENT |
                  (SIZEOF(i.items) <> 0) AND
                  (i.role.name = 'item structure') AND
                  (i.assigned_group IN groups));

  REPEAT i := 1 TO HIINDEX(gr_ass) WHILE NOT violation;
    violation := SIZEOF(QUERY(inst <* gr_ass[i].items |
                  'structural system' IN WHICH_CLASS(inst))) = 0;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

classification_assignment: the set of all instances of **classification_assignment** entities.
group: the set of all instances of **group** entities.
group_assignment: the set of all instances of **group_assignment** entities.

Formal propositions:

WR1: For every instance of **product_definition** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equals 'hull cross section' and that is in the **items** set of an **applied_group_assignment** whose attribute **role** identifies entities with the **name** equal to 'equivalence', the instance of group that is the **assigned_group** of this **applied_-group_assignment** is to collect nothing at all or at least one instance of **product_definition** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'structural system' in its other **applied_group_assignment** whose attribute **role** identifies entities with the **name** equal to 'item structure'.

5.2.4.65 hull_cross_section_result_for_representation_item

The **hull_cross_section_result_for_representation_item** rule specifies the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list, if the **representation** is of class 'hull cross section result'. There is not to be any **representation_items** with other names than those in the list of names.

EXPRESS specification:

```
*)
RULE
hull_cross_section_result_for_representation_item
FOR (representation, representation_item, group,
 applied_classification_assignment);
  LOCAL
    c_a_set         : SET OF applied_classification_assignment := [];
    classed_rep_set : SET OF representation := [];
       arg_list     : LIST OF STRING := [ 'actual first moment horizontal',
    'actual first moment vertical', 'actual inertia cross',
    'actual inertia horizontal', 'actual inertia torsion',
    'actual inertia vertical', 'actual neutral axis position horizontal',
   'actual neutral axis position vertical', 'actual section area',
   'actual section modulus bottom', 'actual section modulus coaming',
   'actual section modulus deck', 'actual stress bottom',
   'actual stress coaming', 'actual stress deck',
   'actual twist centre horizontal', 'actual twist centre vertical',
   'direction largest inertia', 'direction least inertia',
   'max moment', 'max shear force', 'meter weight', 'min moment',
   'min shear force', 'required section modulus bottom',
   'required section modulus coaming', 'required section modulus deck',
   'required stress bottom', 'required stress coaming', 'required stress
  deck'];
    violation      : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'hull cross section result');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      classed_rep_set:= classed_rep_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i:=1 TO HIINDEX(classed_rep_set) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* classed_rep_set[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every instance of **representation** that has an **applied_classification_assignment** whose attribute **assigned_classification.name** equals 'hull cross section result' for each value out of ['actual first moment horizontal', 'actual first moment vertical', 'actual inertia cross', 'actual inertia horizontal', 'actual inertia torsion', 'actual inertia vertical', 'actual neutral axis position horizontal', 'actual neutral axis position vertical', 'actual section area', 'actual section modulus bottom', 'actual section modulus coaming', 'actual section modulus deck', 'actual stress bottom', 'actual stress coaming', 'actual stress deck', 'actual twist centre horizontal', 'actual twist centre vertical', 'direction largest inertia', 'direction least inertia', 'max moment', 'max shear force', 'meter weight', 'min moment', 'min shear force', 'required section modulus bottom', 'required section modulus coaming', 'required section modulus deck', 'required stress bottom', 'required stress coaming', 'required stress deck'] collect exactly and only one instance of **representation_item** in its attribute **items** whose **name** attribute equals that value.

5.2.4.66 identification_role_optional_attribute_description_required

The **identification_role_optional_attribute_description_required** rule specifies that for all instances of type **identification_role** the optional attribute **description** is present if attribute **name** of that instance has a value 'external reference'.

EXPRESS specification:

```
*)
RULE identification_role_optional_attribute_description_required
FOR (identification_role);
  WHERE
    WR1: SIZEOF(QUERY(i <* identification_role |
      ((i.name = 'external reference')
      AND NOT(EXISTS (i.description))))) = 0;
END_RULE;
(*
```

Argument definitions:

identification_role: the set of all instances of **identification_role** entities.

Formal propositions:

WR1: Optional attribute **description** of every instance of **identification_role** exists if **name** equals 'external reference'.

5.2.4.67 initiator_for_change_request

The **initiator_for_change_request** rule specifies that an instance of **versioned_action_request** with **role** 'initiator' is referenced by exactly one assignment instances of type **applied_person_and_-organization_assignment** that have the **role** 'initiator'.

EXPRESS specification:

```
*)
RULE
initiator_for_change_request
```



```
FOR(applied_person_and_organization_assignment,
  person_and_organization_role,
    versioned_action_request);
  LOCAL
    t1_set  : SET OF versioned_action_request := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* versioned_action_request |
             VALUE_IN(WHICH_CLASS(a), 'change request'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
     a_set := QUERY(b <* applied_person_and_organization_assignment |
                (VALUE_IN(b.items, t1_set[i]) AND
                (b.role.name = 'initiator')));
     violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_person_and_organization_assignment: the set of all instances of **applied_person_and_-organization_assignment** entities.
person_and_organization_role: the set of all instances of **person_and_organization_role** entities.
versioned_action_request: the set of all instances of **versioned_action_request** entities.

Formal propositions:

WR1: Every instance of **versioned_action_request** whose **role** equals 'initiator' is referenced by exactly one instance of **applied_person_and_organization_assignment** whose **role** equals 'initiator'.

5.2.4.68 library_manufacturing_definition_for_class_references

The **library_manufacturing_definition_for_class_references** rule specifies that an instance of **product_definition_shape** with class id 'library manufacturing definition' is referenced by at least one instance of **shape_aspect** with class id 'source' via **of_shape**.

EXPRESS specification:

```
*)
RULE
library_manufacturing_definition_for_class_references
FOR(shape_aspect,
  group, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET Of shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME =
                  'library manufacturing definition');
```

```
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;

  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set | 'source'
      IN WHICH_CLASS(t2_inst))) < 1;
END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'library manufacturing definition' is referenced by one or more instances of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'source' through attribute **of_shape**.

5.2.4.69 lightship_definition_has_properties

The **lightship_definition_has_properties** rule specifies that a **product_definition** with a class id 'lightship definition' is referenced by one **property_definition_representation** with the **name** 'lightship definition' via a **property_definition**.

EXPRESS specification:

```
*)
RULE lightship_definition_has_properties
FOR (product_definition, property_definition,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    t3_set    : SET OF property_definition := [];
    t4_set    : SET OF product_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id
  'lightship_definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
```



```
                    i.ASSIGNED_CLASS.NAME = 'lightship_definition');

  (* get all instances of product_definition that have class id
  'lightship_definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* get all property_definition_representation instances with name
  'lightship_definition' *)
  t2_set:= QUERY(i <* PROPERTY_DEFINITION_REPRESENTATION |
                    i.NAME = 'lightship_definition');
  (* get all property_definition instances which are the .definition of
  the
  property_definition_representation *)
  REPEAT i := 1 TO HIINDEX(t2_set);
    t3_set := t3_set + t2_set[i].definition;
  END_REPEAT;
  (* get all product_definition instances which are the .definition of the
  property_definition *)
  REPEAT i := 1 TO HIINDEX(t3_set);
    t4_set := t4_set + t3_set[i].definition;
  END_REPEAT;
  (* compare both lists with product_definition instances which have to be
  identical *)
  violation := t1_set <> t4_set;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose attribute **assigned_class.name** equals 'lightship definition' is referenced by exactly one instance of **property_definition** through attribute **definition** that in turn is referenced by an instance of **property_definition_representation** through attribute **definition** whose attribute **name** equals 'lightship definition'.

5.2.4.70 mandatory_entity_type_for_external_instance_reference

The **mandatory_entity_type_for_external_instance_reference** rule specifies that each instance of **external_source** with attribute **description** equals 'schema name' there must be at least one instance of **external_source_relationship** with **description** equals 'entity type' related to each other.

EXPRESS specification:

```
*)
RULE mandatory_entity_type_for_external_instance_reference
FOR(external_source,
    external_source_relationship);
  LOCAL
    bag_1   : BAG OF external_source := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  bag_1 := QUERY(a <* external_source | a.description = 'schema');

  REPEAT i := 1 TO SIZEOF(bag_1) WHILE (NOT violate);
     violate := (SIZEOF( QUERY(
    a <* external_source_relationship | (a.related_source :=: bag_1[i]) AND
           (a.relating_source.description = 'entity type'))) = 0 );
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

external_source: the set of all instances of **external_source** entities.
external_source_relationship: the set of all instances of **external_source_relationship** entities.

Formal propositions:

WR1: Every instance of **external_source** whose **description** equals 'schema **name**' must be the **related_source** of an instance of **external_source_relationship** whose **relating_source** is an instance of **external_source** that has a **description** equal to 'entity type'.

5.2.4.71 maximum_representation_items_for_class_library_definition

The **maximum_representation_items_for_class_library_definition** rule specifies the aggregate type attribute **items** that is an attribute in entity data type **representation** that is of class 'library definition' to have maximum one elements.

EXPRESS specification:

```
*)
RULE
maximum_representation_items_for_class_library_definition
FOR (representation, applied_classification_assignment,
  classification_role,  group);
  LOCAL
    var_representation : BAG OF representation := [];
  END_LOCAL;
  var_representation := QUERY(
           temp1 <* representation |
              SIZEOF (
                 QUERY(
                    temp2 <* bag_to_set(USEDIN(temp1,
        'SHIP_STRUCTURES_SCHEMA.applied_classification_assignment.ITEMS'))
        | ((temp2\classification_assignment.role.name =
             'class membership') AND
             (temp2\classification_assignment.assigned_class\group.name =
```



```
              'library definition'))
                   )
              ) > 0
            );
  WHERE
    WR1: SIZEOF(QUERY(temp3 <* var_representation |
                 SIZEOF(temp3.items) > 1)) = 0;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.
group: the set of all instances of **group** entities.
classification_role: the set of all instances of **classification_role** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **representation** referenced by the **items** attribute of **applied_classification_assignment** with an attribute **role** that references a **classification_role** with attribute **name** equals 'class membership' also reference by an attribute **assigned_classification** a **group** with attribute **name** equals ‘library definition’. There are at least one elements in the aggregate attribute **items**.

5.2.4.72 members_is_referenced_by_at_least_one_revision

The **members_is_referenced_by_at_least_one_revision** rule specifies that each instance of type **group** with class 'members' is referenced by at least one assignment of type **applied_group_assignment** with role 'revision' via attribute **assigned_group**.

EXPRESS specification:

```
*)
RULE members_is_referenced_by_at_least_one_revision
FOR(applied_group_assignment, group);
  LOCAL
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* group | VALUE_IN(WHICH_CLASS(a), 'members'));

  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_group_assignment |
               (b.assigned_group = t1_set[i]) AND (b.role.name =
  'revision'));
    violate := SIZEOF(a_set) < 1;
  END_REPEAT;

  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.

Formal propositions:

WR1: Every instance of **group** that has an **applied_classification_assignment** whose **assigned_classification** identifies an entity with attribute **name** equal to 'members' is referenced by one or more instances of type **applied_group_assignment** whose role equals 'revision' through attribute **assigned_group**.

5.2.4.73 no_approvals_except_in_approval_history

The **no_approvals_except_in_approval_history** rule specifies that there is no instance of type approval with class 'approval event', that is not part of a **group** of class 'approval history'. This is to ensure existence dependence for attribute **approval_event** of **approval_history**.

EXPRESS specification:

```
*)
RULE
no_approvals_except_in_approval_history
FOR (approval);
  LOCAL
    t1_set  : SET OF approval := [];
    t2_set  : SET OF APPLIED_GROUP_ASSIGNMENT := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* approval | VALUE_IN(WHICH_CLASS(a),
                                  'approval event'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
     t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS'));
     violate := (SIZEOF(t2_set) = 0);
     REPEAT k := 1 TO HIINDEX(t2_set) WHILE NOT violate;
        violate := NOT (VALUE_IN(WHICH_CLASS(t2_set[k].ASSIGNED_GROUP),
        'approval history'));
     END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

approval: the set of all instances of approval entities.

Formal propositions:

WR1: Every instance of **approval** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'approval event' is collected in the **items** of an instance of **applied_group_assignment** whose **assigned_group** has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'approval history'.

5.2.4.74 panel_system_design_definition_has_at_least_one_class

The **panel_system_design_definition_has_at_least_one_class** rule specifies that an instance of **product_definition_shape** with class id 'panel system design definition' is referenced by at least one instance of **shape_aspect** with class id 'boundary' via the **of_shape** attribute.

EXPRESS specification:

```
*)
RULE
panel_system_design_definition_has_at_least_one_class
FOR(shape_aspect,
  group, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET Of shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'panel system design definition');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set | 'boundary'
      IN WHICH_CLASS(t2_inst))) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that is referenced by the **items** attribute of an instance of **applied_classification_assignment** whose **assigned_class** attribute references an instance of **assigned_classification** whose **name** attribute equals 'panel system design definition' is referenced through attribute **of_shape** by one or more instances of **shape_aspect** that are referenced by an instance of **applied_classification_assignment** whose **assigned_class** attribute references an instance of **assigned_classification** whose **name** attribute equals 'boundary' through attribute **of_shape**.

5.2.4.75 panel_system_design_definition_has_at_most_one_class

The **panel_system_design_definition_has_at_most_one_class** rule specifies that an instance of **product_definition_shape** with class id 'panel system design definition' is referenced by at most one instance of **shape_aspect** with class id 'moulded surface' via the **of_shape** attribute.

EXPRESS specification:

```
*)
RULE
panel_system_design_definition_has_at_most_one_class
FOR(shape_aspect,
  group, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET Of shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'panel system design definition');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set |
    'moulded surface' IN WHICH_CLASS(t2_inst))) > 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that is referenced by the **items** attribute of an instance of **applied_classification_assignment** whose **assigned_class** attribute references an instance of **assigned_classification** whose **name** attribute equals 'panel system design definition' is referenced through attribute **of_shape** by one or more instances of **shape_aspect** that are referenced by an instance of **applied_classification_assignment** whose **assigned_class** attribute references an instance of **assigned_classification** whose **name** attribute equals 'moulded surface' through attribute **of_shape**.

5.2.4.76 panel_system_optionally_contains_specified_classes

The **panel_system_optionally_contains_specified_classes** rule specifies that a **product_definition** of class 'panel system' that also is a **group**, as defined by an **applied_group_assignment** with an 'equivalence' **role** and which relates the group and **product_definition** instances, may have another **applied_group_assignment** referencing the same **group** instance whose items set, if not empty, must contain at least one **product_definition** of class 'plate'.

EXPRESS specification:

```
*)
RULE
panel_system_optionally_contains_specified_classes
FOR (group, applied_classification_assignment, applied_group_assignment);
  LOCAL
    cl_ass    : SET OF applied_classification_assignment := [];
    pd        : SET OF PRODUCT_DEFINITION := [];
    gr_ass    : SET OF APPLIED_GROUP_ASSIGNMENT := [];
    groups    : SET OF GROUP := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  cl_ass := QUERY(i <* applied_classification_assignment |
                  (i.role.name = 'class membership') AND
                  (i.assigned_class.name = 'panel system'));
  REPEAT i := 1 TO HIINDEX(cl_ass);
    pd := pd + cl_ass[i].items;
  END_REPEAT;
  gr_ass := QUERY(i <* APPLIED_GROUP_ASSIGNMENT |
                  i.role.name = 'equivalence');
  REPEAT i := 1 TO HIINDEX(gr_ass);
    REPEAT j := 1 TO HIINDEX(gr_ass[i].items);  -- should always be just
  one
      IF (gr_ass[i].items[j] IN pd) THEN
        groups := groups + gr_ass[i].assigned_group;
      END_IF;
    END_REPEAT;
  END_REPEAT;
  gr_ass := QUERY(i <* APPLIED_GROUP_ASSIGNMENT |
                  (SIZEOF(i.items) <> 0) AND
                  (i.role.name = 'item structure') AND
                  (i.assigned_group IN groups));
  REPEAT i := 1 TO HIINDEX(gr_ass) WHILE NOT violation;
    violation := SIZEOF(QUERY(inst <* gr_ass[i].items |
                              'plate' IN WHICH_CLASS(inst))) = 0;
  END_REPEAT;
```

```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
classification_assignment: the set of all instances of **classification_assignment** entities.
applied_group_assignment: the set of all instances of **applied_group_assignment** entities.

Formal propositions:

WR1: For every instance of product_definition that has an **applied_classification_assignment** whose attribute **assigned_class** is a **group** with attribute **name** equal to 'panel system' and that is in the **items** set of an **applied_group_assignment** whose **role** attribute has a **name** equal to 'equivalence', the instance of **group** that is the **assigned_group** of the **applied_group_assignment** may also be the **assigned_group** of another **applied_group_assignment** whose role attribute has a **name** equal to 'item structure'. If the **group** serves both the 'equivalence' and 'item structure' **roles**, then the **applied_group_assignment** which has the 'item structure' role should either have an **items** set which is empty, or the **items** set should contain one or more instances of **product_definition** which are referenced by an **applied_classification_assignment** whose attribute **assigned_class** is a **group** with attribute **name** equal to 'plate'.

5.2.4.77 plate_design_definition_has_at_least_one_class_references

The **plate_design_definition_has_at_least_one_class_references** rule specifies that an instance of **product_definition_shape** with class id 'plate design definition' is referenced by at least one instance of **shape_aspect** with class id 'boundary' via **of_shape**.

EXPRESS specification:

```
*)
RULE
plate_design_definition_has_at_least_one_class_references
FOR(shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET Of shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'plate design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set | 'boundary'
```



```
      IN WHICH_CLASS(t2_inst))) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'plate design definition' is referenced by one or more instances of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'boundary' through attribute **of_shape**.

5.2.4.78 plate_design_definition_has_at_most_one_class_references

The **plate_design_definition_has_at_most_one_class_references** rule specifies that an instance of **product_definition_shape** with class id 'plate design definition' is referenced by at most one instance of **shape_aspect** with class id 'moulded surface' via **of_shape**.

EXPRESS specification:

```
*)
RULE
plate_design_definition_has_at_most_one_class_references
FOR(shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET Of shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME =
                  'plate design definition');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set |
      'moulded surface' IN WHICH_CLASS(t2_inst))) > 1;
END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'plate design definition' is referenced by one or less instances of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'moulded surface' through attribute **of_shape**.

5.2.4.79 plate_manufacturing_definition_has_class_references

The **plate_manufacturing_definition_has_class_references** rule specifies that an instance of **product_definition_shape** with class id 'plate manufacturing definition' is referenced by at least one instance of **shape_aspect** with class id 'outer contour' via **of_shape**.

EXPRESS specification:

```
*)
RULE
plate_manufacturing_definition_has_class_references
FOR(shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET Of shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME =
                  'plate manufacturing definition');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set | 'outer contour'
      IN WHICH_CLASS(t2_inst))) < 1;
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'plate manufacturing definition' is referenced by one or more instances of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'outer contour' through attribute **of_shape**.

5.2.4.80 principal_characteristics_has_properties

The **principal_characteristics_has_properties** rule specifies that a **product_definition** with a class id 'principal characteristics' is referenced by one **property_definition_representation** with the **name** 'principal characteristics' via a **property_definition**.

EXPRESS specification:

```
*)
RULE principal_characteristics_has_properties
FOR (property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    t3_set    : SET OF property_definition := [];
    t4_set    : SET OF product_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'principal characteristics');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  t2_set:= QUERY(i <* PROPERTY_DEFINITION_REPRESENTATION |
                   i.NAME = 'principal characteristics');
  REPEAT i := 1 TO HIINDEX(t2_set);
    t3_set := t3_set + t2_set[i].definition;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t3_set);
    t4_set := t4_set + t3_set[i].definition;
  END_REPEAT;
```



```
  violation := t1_set <> t4_set;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equal to 'principal characteristics' is referenced by exactly one instance of **property_definition** through attribute **definition** that in turn is referenced by an instance of **property_definition_representation** through attribute **definition** whose attribute **name** equals 'principal characteristics'.

5.2.4.81 product_definition_for_call_sign

The **product_definition_for_call_sign** rule specifies that an instance of **product_definition** with class id 'ship designation' is referenced by exactly one instance of **applied_identification_-assignment** via **items** whose attribute **role** has the value 'call sign'.

EXPRESS specification:

```
*)
RULE
product_definition_for_call_sign
FOR(product_definition,
   applied_identification_assignment, group,
   applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF applied_identification_assignment := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'ship designation');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
        'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.role.name =
        'call sign')) = 1);
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.
applied_identification_assignment: the set of all instances of **applied_identification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'ship designation', is referenced by exactly one **applied_identification_assignment** attribute **items**, where **applied_identification_assignment** attribute **role** equals 'call sign'.

5.2.4.82 product_definition_for_class_notation

The **product_definition_for_class_notation** rule specifies that an instance of **product_definition** with class id 'class and statutory designation' is referenced by exactly one instance of **property_-definition** with class id 'class notation' via **definition**.

EXPRESS specification:

```
*)
RULE
product_definition_for_class_notation
FOR(property_definition, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'class and statutory designation');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | 'class notation'
      IN WHICH_CLASS(t2_inst))) = 1);
END_REPEAT;
```

```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
property_definition: the set of all instances of **property_definition** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose attribute **assigned_classification** has an entity with **name** attribute equal to 'class and statutory designation' is referenced by exactly one instance of **property_definition** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'class notation' through attribute **definition**.

5.2.4.83 product_definition_for_design_still_water_shear_force_values

The **product_definition_for_design_still_water_shear_force_values** rule specifies that an instance of **product_definition** with class id 'design still water shear force' is referenced by exactly one instance of **property_definition** through the attribute **definition** whose attribute **name** has the value 'design still water shear force values'.

EXPRESS specification:

```
*)
RULE
product_definition_for_design_still_water_shear_force_values
FOR(product_definition,
    property_definition_representation,
    applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with
     id 'design still water shear  force' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME =
                  'design still water shear  force');
  (* get all instances of T1 that have class
     id 'design still water shear  force' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition in t1_set:
     get the property_definition_representation instances
     that are referencing a product_definition instance via
```



```
     definition, filter out those property_definition_representation
     instances whose attribute name has the value
     'design still water shear force values'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
    'design still water shear force values')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'design still water shear force', is referenced by exactly one **property_definition.definition**, where **property_definition.name** equals 'design still water shear force values'.

5.2.4.84 product_definition_for_flag_state

The **product_definition_for_flag_state** rule specifies that an instance of **product_definition** with class id 'ship designation' is referenced by exactly one instance of **applied_identification_-assignment** via **items** whose attribute **role** has the value 'flag state'.

EXPRESS specification:

```
*)
RULE
product_definition_for_flag_state
FOR(product_definition,
  applied_identification_assignment, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF applied_identification_assignment := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'ship designation');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
```

```
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.role.name =
    'flag state')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
product_definition: the set of all instances of **product_definition** entities.
applied_identification_assignment: the set of all instances of **applied_identification_assignment** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'ship designation', is referenced by exactly one **applied_identification_assignment** attribute **items**, where **applied_identification_assignment** attribute **role** equals 'flag state'.

5.2.4.85 product_definition_for_port_of_registration

The **product_definition_for_port_of_registration** rule specifies that an instance of **product_-definition** with class id 'ship designation' is referenced by exactly one instance of **applied_-identification_assignment** via **items** whose attribute **role** has the value 'port of registration'.

EXPRESS specification:

```
*)
RULE
product_definition_for_port_of_registration
FOR(product_definition,
  applied_identification_assignment, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF applied_identification_assignment := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'ship designation');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.role.name =
```



```
    'port of registration')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
group: the set of all instances of **group** entities.
applied_identification_assignment: the set of all instances of **applied_identification_assignment** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'ship designation', is referenced by exactly one **applied_identification_assignment** attribute **items**, where **applied_identification_assignment** attribute **role** equals 'port of registration'.

5.2.4.86 product_definition_for_regulation

The **product_definition_for_regulation** rule specifies that an instance of **product_definition** with class id 'class and statutory designation' is referenced by exactly one instance of **property_definition** with class id 'regulation' via **definition**.

EXPRESS specification:

```
*)
RULE
product_definition_for_regulation
FOR(product_definition,
  property_definition, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'class and statutory designation');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | 'regulation'
    IN WHICH_CLASS(t2_inst))) = 1);
END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
group: the set of all instances of **group** entities
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'class and statutory designation' is referenced by exactly one instance of **property_definition** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'regulation' through attribute **definition**.

5.2.4.87 product_definition_for_shipyard

The **product_definition_for_shipyard** rule specifies that an instance of **product_definition** with class id 'shipyard designation' is referenced by exactly one instance of **applied_organization_-assignment** via **items** whose attribute **role** has the value 'shipyard'.

EXPRESS specification:

```
*)
RULE
product_definition_for_shipyard
FOR(product_definition,
   applied_organization_assignment, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition := [];
     t2_set    : SET OF applied_organization_assignment := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'shipyard designation');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.role.name =
     'shipyard')) = 1);
   END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
group: the set of all instances of **group** entities
applied_organization_assignment: the set of all instances of **applied_organization_assignment** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'shipyard designation', is referenced by exactly one **applied_organization_-assignment** attribute **items**, where **applied_organization_assignment** attribute **role** equals 'shipyard'.

5.2.4.88 product_definition_has_references_with_class_loadline

The **product_definition_has_references_with_class_loadline** rule specifies that an instance of **product_definition** with class id 'freeboard characteristics' is referenced by exactly one instance of **property_definition** with class id 'loadline' via **definition**.

EXPRESS specification:

```
*)
RULE product_definition_has_references_with_class_loadline
FOR(product_definition, property_definition,
  applied_classification_assignment);
  (*
  R21<PRODUCT_DEFINITION,'freeboard
  characteristics',1,PROPERTY_DEFINITION,DEFINITION,'loadline'> *)
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'freeboard
  characteristics' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'freeboard characteristics');

  (* get all instances of product_definition that have class id 'freeboard
  characteristics' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition in t1_set:
     get the property_definition instances that are referencing a
```

```
  product_definition instance via definition,
     filter out those property_definition instances whose class id is
  'loadline'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
          'SHIP_STRUCTURES_SCHEMA.' + 'PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | 'loadline'
           IN WHICH_CLASS(t2_inst))) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'freeboard characteristics' is referenced by exactly one instance of **property_definition** that have an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'loadline' through attribute **definition**.

5.2.4.89 product_definition_local_co_ordinate_system_with_station_-reference

The **product_definition_local_co_ordinate_system_with_station_reference** rule specifies that an instance of **product_definition** with class id 'hull cross section' is referenced by exactly one instance of **property_definition** with class id 'local co_ordinate system with station reference' via **definition**.

EXPRESS specification:

```
*)
RULE
product_definition_local_coordinate_system_with_station_reference
FOR(product_definition,
property_definition, group,
applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF property_definition := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'hull cross section');

  REPEAT i := 1 TO HIINDEX(c_a_set);
```



```
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     'local coordinate system with station reference'
     IN WHICH_CLASS(t2_inst))) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
property_definition: the set of all instances of **property_definition** entities.
group: the set of all instances of **group** entities
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'hull cross section' is referenced by exactly one instance of **property_definition** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute of value 'local coordinate system with station reference' through attribute **definition**.

5.2.4.90 product_definition_relationship_with_definable_object

The **product_definition_relationship_with_definable_object** rule specifies a string type attribute **product_definition_relationship** of type 'definable object' to match the pattern '*.*' if the instance of **product_definition_relationship** is of class 'definable object'.

EXPRESS specification:

```
*)
RULE
product_definition_relationship_with_definable_object
FOR (product_definition_relationship, group,
  applied_classification_assignment);
  (*
  R7<PRODUCT_DEFINITION_RELATIONSHIP,'definable object', ID,'*.*'>*)
  LOCAL
    clfied_inst :SET OF GENERIC := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'definable object') THEN
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
END_IF;
```

```
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **product_definition_relationship** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'definable object' is to match the pattern '*.*'.

5.2.4.91 product_definition_relationship_with_identification_assignment

The **product_definition_relationship_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE product_definition_relationship_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_relationship := [];
    t2_set    : SET OF product_definition_relationship := [];
    arg_list  : LIST OF STRING := ['assembly relationship',
                'structural part penetration relationship',
                'structural system penetration relationship' ];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id  *)
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        c_a_set := QUERY(i <* applied_classification_assignment |
                     i.assigned_class.NAME =  arg_LIST[j]);
   END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
    END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_relationship** that is referenced by an **applied_classification_assignment** whose assigned_classification has a **name** attribute of value 'assembly relationship' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.92 product_definition_shape_for_assembly_manufacturing

The **product_definition_shape_for_assembly_manufacturing** rule specifies that an instance of **product_definition_shape** with class id 'assembly manufacturing definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'assembly manufacturing definition parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_assembly_manufacturing
FOR(property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'assembly manufacturing definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
              'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
                 'assembly manufacturing definition parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'assembly manufacturing definition', is referenced by exactly one **property_definition_representation** attribute **definition**, where attribute **name** equals 'assembly manufacturing definition parameters'.

5.2.4.93 product_definition_shape_for_bevel_design_parameters

The **product_definition_shape_for_bevel_design_parameters** rule specifies that an instance of **product_definition_shape** with class id 'bevel design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'bevel design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_bevel_design_parameters
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'bevel design
  definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'bevel design definition');
  (* get all instances of T1 that have class id 'bevel design definition'
  *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:   get the
  property_definition_representation instances that are referencing a
  product_definition_shape instance via definition, filter out those
  property_definition_representation instances whose attribute name has
  the value 'bevel design parameters' check if their number equals 1  *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     t2_inst.name = 'bevel design parameters')) = 1);
```



```
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'bevel design definition', is referenced by exactly one **property_definition_-representation.definition**, where **property_definition_representation.name** equals 'bevel design parameters'.

5.2.4.94 product_definition_shape_for_beveled_groove_weld_design

The **product_definition_shape_for_beveled_groove_weld_design** rule specifies that an instance of **product_definition_shape** with class id 'beveled groove weld' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'beveled groove weld design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_beveled_groove_weld_design
FOR(product_definition_shape, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'beveled groove weld');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
         'beveled groove weld design parameters')) = 1);
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'beveled groove weld', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_-representation** attribute **name** equals 'beveled groove weld design parameters'.

5.2.4.95 product_definition_shape_for_butt_groove_weld_design

The **product_definition_shape_for_butt_groove_weld_design** rule specifies that an instance of **product_definition_shape** with class id 'butt groove weld' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'butt groove weld design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_butt_groove_weld_design
FOR(product_definition_shape, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'butt groove weld');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
        'butt groove weld design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition**_shape that is referenced by an **applied_classification_assignment** with **name** equal to 'butt groove weld', is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'butt groove weld design parameters'.

5.2.4.96 product_definition_shape_for_circular_cutout_design_-parameters

The **product_definition_shape_for_circular_cutout_design_parameters** rule specifies that an instance of **product_definition_shape** with class id 'circular cutout design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'circular cutout design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_circular_cutout_design_parameters
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'circular cutout
  design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'circular cutout design
  definition');
  (* get all instances of T1 that have class id 'circular cutout design
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:    get the
  property_definition_representation instances that are referencing a
  product_definition_shape instance via definition, filter out those
  property_definition_representation instances whose attribute name has
  the value 'circular cutout design parameters' check if their number
  equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
  'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
```

```
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
'circular cutout design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'circular cutout design definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'circular cutout design parameters'.

5.2.4.97 product_definition_shape_for_class_torch_vector

The **product_definition_shape_for_class_torch_vector** rule specifies that an instance of **product_-definition_shape** with class id 'weld manufacturing definition' is referenced by exactly one instance of **shape_aspect** with class id 'torch vector' via **of_shape**.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_class_torch_vector
FOR(product_definition_shape,
  shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'weld manufacturing definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     'torch vector' IN WHICH_CLASS(t2_inst))) = 1);
```



```
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'weld manufacturing definition' is referenced by exactly one instance of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'torch vector' through attribute **of_shape**.

5.2.4.98 product_definition_shape_for_continuous_fillet_weld_design

The **product_definition_shape_for_continuous_fillet_weld_design** rule specifies that an instance of **product_definition_shape** with class id 'continuous fillet weld' is referenced by exactly one instance of **property_definition_representation** via **definition** whose **attribute name** has the value 'continuous fillet weld design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_continuous_fillet_weld_design
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME = 'continuous fillet weld');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
        'continuous fillet  weld design parameters')) = 1);
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition**_shape that is referenced by an **applied_classification_assignment** with **name** equal to 'continuous fillet weld', is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'continuous fillet weld design parameters'.

5.2.4.99 product_definition_shape_for_corrugated_part_design

The **product_definition_shape_for_corrugated_part_design** rule specifies that an instance of **product_definition_shape** with class id 'corrugated part design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose **attribute name** has the value 'corrugated part design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_corrugated_part_design
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'corrugated part design definition');
   REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'corrugated part design parameters')) = 1);
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition**_shape that is referenced by an **applied_classification_assignment** with **name** equal to 'corrugated part design definition', is referenced by exactly one **property_-definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'corrugated part design parameters'.

5.2.4.100 product_definition_shape_for_corrugated_structure_design

The **product_definition_shape_for_corrugated_structure_design** rule specifies that an instance of **product_definition_shape** with class id 'corrugated structure design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** which has an attribute **name** has the value ' corrugated structure design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_corrugated_structure_design
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'corrugated structure design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      ' corrugated structure design parameters')) = 1);
  END_REPEAT;
```

```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'corrugated structure design definition', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'corrugated structure design parameters'.

5.2.4.101 product_definition_shape_for_corrugation

The **product_definition_shape_for_corrugation** rule specifies that an instance of **product_-definition_shape** with class id 'corrugated part design definition' is referenced by exactly one instance of **shape_aspect** with class id 'corrugation' via **of_shape**.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_corrugation
FOR(product_definition_shape,
  shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'corrugated part design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     'corrugation' IN WHICH_CLASS(t2_inst))) = 1);
END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
group: the set of all instances of **group** entities
shape_aspect: the set of all instances of **shape_aspect** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'corrugated part design definition' is referenced by exactly one instance of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'corrugation' through attribute **of_shape**.

5.2.4.102 product_definition_shape_for_drain_hole_cutout_design

The **product_definition_shape_for_drain_hole_cutout_design** rule specifies that an instance of **product_definition_shape** with class id 'drain hole cutout design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'drain hole cutout design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_drain_hole_cutout_design
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'drain hole cutout
  design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'drain hole cutout design
  definition');
  (* get all instances of T1 that have class id 'drain hole cutout design
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
     get the property_definition_representation instances that are
  referencing a product_definition_shape instance via definition,
```

```
   filter out those property_definition_representation instances whose
   attribute name has the value 'drain hole cutout design parameters'
   check if their number equals 1   *)
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
 'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     t2_inst.name = 'drain hole cutout design parameters')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'drain hole cutout design definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'drain hole cutout design parameters'.

5.2.4.103 product_definition_shape_for_elliptical_cutout_design

The **product_definition_shape_for_elliptical_cutout_design** rule specifies that an instance of **product_definition_shape** with class id 'elliptical cutout design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'elliptical cutout design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_elliptical_cutout_design
FOR(product_definition_shape,
   property_definition_representation,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   (* get all classification_assignment instances with id 'elliptical cutout
   design definition' *)
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.ASSIGNED_CLASS.NAME = 'elliptical cutout design
   definition');
   (* get all instances of T1 that have class id 'elliptical cutout design
   definition' *)
```



```
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
     get the property_definition_representation instances that are
  referencing a product_definition_shape instance via definition,
     filter out those property_definition_representation instances
   whose attribute name has the value 'elliptical cutout design parameters'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
   'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
    t2_inst.name = 'elliptical cutout design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'elliptical cutout design definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'elliptical cutout design parameters'.

5.2.4.104 product_definition_shape_for_elongated_oval_cutout_design

The **product_definition_shape_for_elongated_oval_cutout_design** rule specifies that an instance of **product_definition_shape** with class id 'elongated oval cutout one design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'elongated oval cutout design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_elongated_oval_cutout_design
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
```

```
  END_LOCAL;

  (* get all classification_assignment instances with id 'elongated oval
  cutout l design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'elongated oval cutout l design
  definition');
  (* get all instances of T1 that have class id 'elongated oval cutout l
  design definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
     get the property_definition_representation instances that are
    referencing a product_definition_shape instance via definition,
     filter out those property_definition_representation instances
    whose attribute name has the value 'elongated oval cutout design
  parameters'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
          'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
      t2_inst.name = 'elongated oval cutout design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'elongated oval cutout l design definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'elongated oval cutout design parameters'.

5.2.4.105 product_definition_shape_for_fillet_weld_design_parameters

The **product_definition_shape_for_fillet_weld_design_parameters** rule specifies that an instance of **product_definition_shape** with class id 'fillet weld' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'fillet weld design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_fillet_weld_design_parameters
FOR(product_definition_shape,
   property_definition_representation, group,
   applied_classification_assignment);
   (*
   R19<PRODUCT_DEFINITION_SHAPE,'fillet weld',1,
   PROPERTY_DEFINITION_REPRESENTATION, DEFINITION,NAME,'fillet  weld design
   parameters'> *)
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME = 'fillet weld');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
   'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
       'fillet  weld design parameters')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to ‘fillet weld’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_-representation** attribute **name** equals ‘fillet weld design parameters’.

5.2.4.106 product_definition_shape_for_free_form_interior_cutout_design

The **product_definition_shape_for_free_form_interior_cutout_design** rule specifies that an instance of **product_definition_shape** with class id 'free form interior cutout design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'free form interior cutout design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_free_form_interior_cutout_design
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id
    'free form interior cutout design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'free form interior cutout
  design definition');
  (* get all instances of T1 that have class id
      'free form interior cutout design definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
     get the property_definition_representation instances that
     are referencing a product_definition_shape instance via definition,
     filter out those property_definition_representation instances
     whose attribute name has the value 'free form interior cutout design
  parameters'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
    'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
      t2_inst.name = 'free form interior cutout design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'free form interior cutout design definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'free form interior cutout design parameters'.

5.2.4.107 product_definition_shape_for_groove_weld_design

The **product_definition_shape_for_groove_weld_design** rule specifies that an instance of **product_definition_shape** with class id 'groove weld' is referenced by exactly one instance of **property_definition_representation** via **definition** whose **attribute name** has the value 'groove weld design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_groove_weld_design
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  (*
  R19<PRODUCT_DEFINITION_SHAPE,'groove
  weld',1,PROPERTY_DEFINITION_REPRESENTATION,DEFINITION,NAME,'groove weld
  design parameters'> *)
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'groove weld');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'groove weld design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'groove weld', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_-representation** attribute **name** equals 'groove weld design parameters'.

5.2.4.108 product_definition_shape_for_hull_cross_section_design_-definition

The **product_definition_shape_for_hull_cross_section_design_definition** rule specifies that for each instance of **product_definition** of class 'hull cross section' that is referenced through attribute **definition** from an instance **product_definition_shape** with class 'hull cross section design definition' all those **product_definition_shape** is collected in the same **group_assignment** of **role** 'versions' with assigned **group** having the class 'version history'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_hull_cross_section_design_definition
FOR(product_definition);
  LOCAL
    t1_set1    : SET OF product_definition_shape := [];
    t1_set2    : SET OF product_definition_shape := [];
    t2_set     : SET OF product_definition := [];
    gr_ass_set : SET OF applied_group_assignment := [];
    violation  : LOGICAL := FALSE;
  END_LOCAL;
  t2_set := QUERY(all_t2_inst <* product_definition |
                  'hull cross section' IN which_class(all_t2_inst));
  REPEAT i := 1 TO HIINDEX(t2_set) WHILE NOT violation;
    t1_set1 := QUERY (pd <* bag_to_set(USEDIN(t2_set[i],
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'))|
    'SHIP_STRUCTURES_SCHEMA.PRODUCT_DEFINITION_SHAPE' IN TYPEOF(pd));
    t1_set2 := QUERY(t1_inst1 <* t1_set1 |
             'hull cross section design definition'
             IN which_class(t1_inst1));
    REPEAT j := 1 TO HIINDEX(t1_set2);
      gr_ass_set := QUERY(aga <* bag_to_set(USEDIN(t1_set2[j],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS')) |
       (aga.role.name = 'versions')
       AND
      ('version history' IN WHICH_CLASS(aga.assigned_group)));
    END_REPEAT;
    violation := SIZEOF(gr_ass_set) > 1;
```



```
    gr_ass_set := [];
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'hull cross section design definition' and that references through its attribute **definition** an instance of **product_definition** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'hull cross section' is **items** in the same instance of **applied_group_assignment** whose **role** equals 'versions' and whose **assigned_group** has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'version history'.

5.2.4.109 product_definition_shape_for_intermittent_fillet_weld_design

The **product_definition_shape_for_intermittent_fillet_weld_design** rule specifies that an instance of **product_definition_shape** with class id 'intermittent fillet weld' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'intermittent fillet weld design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_intermittent_fillet_weld_design
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'intermittent fillet weld');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
    'intermittent fillet  weld design parameters')) = 1);
  END_REPEAT;
```

```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: **Every product_definition_shape** that is referenced **by an applied_classification_assignment** with **name** equal to 'intermittent fillet weld', is referenced by exactly one **property_definition_representation** with attribute **definition**, where **property_-definition_representation** has an **attribute name** equals 'intermittent fillet weld design parameters'.

5.2.4.110 product_definition_shape_for_inward_round_corner_design

The **product_definition_shape_for_inward_round_corner_design** rule specifies that an instance of **product_definition_shape** with class id 'inward round corner design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'inward round corner design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_inward_round_corner_design
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'inward round
  corner design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
         i.ASSIGNED_CLASS.NAME = 'inward round corner design definition');
  (* get all instances of T1 that have class id 'inward round corner design
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
   get the property_definition_representation instances that are
  referencing
```



```
   a product_definition_shape instance via definition,
   filter out those property_definition_representation instances
    whose attribute name has the value 'inward round corner design
  parameters'
   check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
    'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     t2_inst.name = 'inward round corner design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'inward round corner design definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'inward round corner design parameters'.

5.2.4.111 product_definition_shape_for_library_definition_parameters

The **product_definition_shape_for_library_definition_parameters** rule specifies that an instance of **product_definition_shape** with class id 'library definition' is referenced by exactly one instance of **shape_aspect** via **of_shape** whose attribute **name** has the value 'library definition parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_library_definition_parameters
FOR(product_definition_shape,
  shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'library definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
```



```
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
       'library definition parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'library definition', is referenced by exactly one **shape_aspect** with attribute **of_shape**, where **shape_aspect** attribute **name** equals 'library definition parameters'.

5.2.4.112 product_definition_shape_for_library_design_parameters

The **product_definition_shape_for_library_design_parameters** rule specifies that an instance of **product_definition_shape** with class id 'library design definition' is referenced by exactly one instance of **shape_aspect** via **of_shape** whose attribute **name** has the value 'library design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_library_design_parameters
FOR(product_definition_shape,
  shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'library design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
```



```
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
       'library design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'library design definition', is referenced by exactly one **shape_aspect** with attribute **of_shape**, where **shape_aspect** attribute **name** equals 'library design parameters'.

5.2.4.113 product_definition_shape_for_library_manufacturing_-defiinition

The **product_definition_shape_for_library_manufacturing_defiinition** rule specifies that an instance of **product_definition_shape** with class id 'library manufacturing definition' is referenced by exactly one instance of **shape_aspect** via **of_shape** whose attribute **name** has the value 'library manufacturing defiinition parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_library_manufacturing_defiinition
FOR(product_definition_shape,
  shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'library manufacturing definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
```

```
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;

   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
       'library manufacturing defiinition parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'library manufacturing definition', is referenced by exactly one **shape_aspect** with attribute **of_shape**, where **shape_aspect** attribute **name** equals 'library manufacturing definition parameters'.

5.2.4.114 product_definition_shape_for_manufacturing_definition

The **product_definition_shape_for_manufacturing_definition** rule specifies that an instance of **product_definition_shape** with class id 'manufacturing definition' is referenced by exactly one instance of **shape_aspect** via **of_shape** whose attribute **name** has the value 'manufacturing definition parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_manufacturing_definition
FOR(product_definition_shape,
shape_aspect, group,
applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'manufacturing definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
```



```
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
       'manufacturing definition parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'manufacturing definition', is referenced by exactly one **shape_aspect** attribute **of_shape**, where **shape_aspect** attribute **name** equals 'manufacturing definition parameters'.

5.2.4.115 product_definition_shape_for_name

The **product_definition_shape_for_name** rule specifies that an instance of **product_definition_-shape** with class id 'library design definition' is referenced by exactly one instance of **shape_aspect** with class id name via **of_shape**.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_name
FOR(product_definition_shape,
  shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'library design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
```

```
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     'name' IN WHICH_CLASS(t2_inst))) = 1);
END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'library design definition' is referenced by exactly one instance of **shape_aspect** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with attribute **name** equal to 'name' through attribute **of_shape**.

5.2.4.116 product_definition_shape_for_outer_contour_representation

The **product_definition_shape_for_outer_contour_representation** rule specifies that an instance of **product_definition_shape** with class id 'plate manufacturing definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'outer contour representation'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_outer_contour_representation
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'plate manufacturing definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
```



```
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
              'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
                   'outer contour representation')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'plate manufacturing definition', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'outer contour representation'.

5.2.4.117 product_definition_shape_for_outward_round_corner_design

The **product_definition_shape_for_outward_round_corner_design** rule specifies that an instance of **product_definition_shape** with class id 'outward round corner design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'outward round corner design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_outward_round_corner_design
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id
     'outward round corner design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
          i.ASSIGNED_CLASS.NAME = 'outward round corner design
  definition');
  (* get all instances of T1 that have class id
     'outward round corner design definition' *)
```

```
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
  get the property_definition_representation instances that are
  referencing a product_definition_shape instance via definition,
  filter out those property_definition_representation instances
  whose attribute name has the value 'outward round corner design
  parameters'
  check if their number equals 1
  *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
   'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
   t2_inst.name = 'outward round corner design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'outward round corner design definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'outward round corner design parameters'.

5.2.4.118 product_definition_shape_for_panel_system_design_parameters

The **product_definition_shape_for_panel_system_design_parameters** rule specifies that an instance of **product_definition_shape** with class id 'panel system design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** which has an attribute **name** has the value 'panel system design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_panel_system_design_parameters
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   :SET OF applied_classification_assignment := [];
```



```
    t1_set    :SET OF product_definition_shape := [];
    t2_set    :SET OF property_definition_representation := [];
    violation :LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                    'panel system design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],

  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'))
  ;
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'panel system design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to ‘panel system design definition’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘panel system design parameters’.

5.2.4.119 product_definition_shape_for_part_edge_cutout_design

The **product_definition_shape_for_part_edge_cutout_design** rule specifies that an instance of **product_definition_shape** with class id 'part edge cutout design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'part edge cutout design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_part_edge_cutout_design
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
```

```
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'part edge cutout
  design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
             i.ASSIGNED_CLASS.NAME = 'part edge cutout design definition');
  (* get all instances of T1 that have class id 'part edge cutout design
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
     get the property_definition_representation instances that
    are referencing a product_definition_shape instance via definition,
     filter out those property_definition_representation instances
    whose attribute name has the value 'part edge cutout design parameters'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
  'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
   t2_inst.name = 'part edge cutout design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'part edge cutout design definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'part edge cutout design parameters'.

5.2.4.120 product_definition_shape_for_plate_design_parameters

The **product_definition_shape_for_plate_design_parameters** rule specifies that an instance of **product_definition_shape** with class id 'plate design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'plate design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_plate_design_parameters
FOR(product_definition_shape,
   property_definition_representation, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                     'plate design definition');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],

   'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'))
   ;
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
     'plate design parameters')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to ‘plate design definition’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_-representation** attribute **name** equals ‘plate design parameters’.

5.2.4.121 product_definition_shape_for_profile_cross_section

The **product_definition_shape_for_profile_cross_section** rule specifies that an instance of **product_definition_shape** with class id 'profile design definition' is referenced by exactly one instance of **shape_aspect** with class id 'profile cross section' via **of_shape**.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_profile_cross_section
FOR(product_definition_shape,
   shape_aspect, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF shape_aspect := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'profile design definition');

   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
      'profile cross section' IN WHICH_CLASS(t2_inst))) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'profile design definition' is referenced by exactly one instance of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'profile cross section' through attribute **of_shape**.

5.2.4.122 product_definition_shape_for_profile_design_parameters

The **product_definition_shape_for_profile_design_parameters** rule specifies that an instance of **product_definition_shape** with class id 'profile design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'profile design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_profile_design_parameters
FOR(product_definition_shape,
   property_definition_representation, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'profile design definition');

   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
   'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
     'profile design parameters')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to ‘profile design definition’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_-representation** attribute **name** equals ‘profile design parameters’.

5.2.4.123 product_definition_shape_for_rectangular_cutout_design

The **product_definition_shape_for_rectangular_cutout_design** rule specifies that an instance of **product_definition_shape** with class id 'rectangular cutout design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'rectangular cutout design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_rectangular_cutout_design
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'rectangular
  cutout design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
             i.ASSIGNED_CLASS.NAME = 'rectangular cutout design
  definition');
  (* get all instances of T1 that have class id 'rectangular cutout design
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
     get the property_definition_representation instances that are
    referencing a product_definition_shape instance via definition,
     filter out those property_definition_representation instances
    whose attribute name has the value 'rectangular cutout design
  parameters'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
   'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
   t2_inst.name = 'rectangular cutout design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'rectangular cutout design definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'rectangular cutout design parameters'.

5.2.4.124 product_definition_shape_for_round_corner_rectangular_-cutout

The **product_definition_shape_for_round_corner_rectangular_cutout** rule specifies that an instance of **product_definition_shape** with class id 'round corner rectangular cutout design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'round corner rectangular cutout design parameters'.

EXPRESS specification:

```
*)
RULE
 product_definition_shape_for_round_corner_rectangular_cutout
FOR(product_definition_shape,
   property_definition_representation,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   (* get all classification_assignment instances with id
     'round corner rectangular cutout design definition' *)
    c_a_set := QUERY(i <* applied_classification_assignment |
   i.ASSIGNED_CLASS.NAME = 'round corner rectangular cutout design
   definition');
   (* get all instances of T1 that have class id
      'round corner rectangular cutout design definition' *)
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   (* for all instances of product_definition_shape in t1_set:
      get the property_definition_representation instances that are
     referencing a product_definition_shape instance via definition,
      filter out those property_definition_representation instances
     whose attribute name has the value
     'round corner rectangular cutout design parameters'
      check if their number equals 1
    *)
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     t2_inst.name = 'round corner rectangular cutout design parameters')) =
   1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'round corner rectangular cutout design definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'round corner rectangular cutout design parameters'.

5.2.4.125 product_definition_shape_for_round_edge_rectangular_cutout

The **product_definition_shape_for_round_edge_rectangular_cutout** rule specifies that an instance of **product_definition_shape** with class id 'round edge rectangular cutout design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'round edge rectangular cutout design parameters'.

EXPRESS specification:

```
*)
RULE
 product_definition_shape_for_round_edge_rectangular_cutout
FOR(product_definition_shape,
   property_definition_representation,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;

   (* get all classification_assignment instances with id
     'round edge rectangular cutout design definition' *)
   c_a_set := QUERY(i <* applied_classification_assignment |
   i.ASSIGNED_CLASS.NAME = 'round edge rectangular cutout design
   definition');
   (* get all instances of T1 that have class id
    'round edge rectangular cutout design definition' *)
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   (* for all instances of product_definition_shape in t1_set:
      get the property_definition_representation instances that
    are referencing a product_definition_shape instance via definition,
      filter out those property_definition_representation instances
    whose attribute name has the value
     'round edge rectangular cutout design parameters'
      check if their number equals 1
```



```
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
   'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
   'round edge rectangular cutout design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'round edge rectangular cutout design definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'round edge rectangular cutout design parameters'.

5.2.4.126 product_definition_shape_for_shear_bevel_design_parameters

The **product_definition_shape_for_shear_bevel_design_parameters** rule specifies that an instance of **product_definition_shape** with class id 'shear bevel design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'shear bevel design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_shear_bevel_design_parameters
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'shear bevel
  design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'shear bevel design
  definition');
  (* get all instances of T1 that have class id 'shear bevel design
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
```

```
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
     get the property_definition_representation instances that are
   referencing a product_definition_shape instance via definition,
     filter out those property_definition_representation instances whose
    attribute name has the value 'shear bevel design parameters'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
    'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'shear bevel design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'shear bevel design definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'shear bevel design parameters'.

5.2.4.127 product_definition_shape_for_source

The **product_definition_shape_for_source** rule specifies that an instance of **product_definition_-shape** with class id 'library definition' is referenced by exactly one instance of **shape_aspect** with class id 'source' via **of_shape**.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_source
FOR(product_definition_shape,
  shape_aspect, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
```



```
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'library definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
      'source' IN WHICH_CLASS(t2_inst))) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'library definition' is referenced by exactly one instance of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'source' through attribute **of_shape**.

5.2.4.128 product_definition_shape_for_spot_seam_weld_design

The **product_definition_shape_for_spot_seam_weld_design** rule specifies that an instance of **product_definition_shape** with class id 'spot seam weld' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'spot seam weld design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_spot_seam_weld_design
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
```

```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'spot seam weld');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
       'spot seam weld design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equals ‘spot seam weld’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_-representation** attribute **name** equals ‘spot seam weld design parameters’.

5.2.4.129 product_definition_shape_for_structural_part_joint_design

The **product_definition_shape_for_structural_part_joint_design** rule specifies that an instance of **product_definition_shape** with class id 'structural part joint design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'structural part joint design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_structural_part_joint_design
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
```



```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'structural part joint design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'structural part joint design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equals ‘structural part joint design definition’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘structural part joint design parameters’.

5.2.4.130 product_definition_shape_for_structural_system_design

The **product_definition_shape_for_structural_system_design** rule specifies that an instance of **product_definition_shape** with class id 'structural system design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** which has an attribute **name** has the value 'structural system design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_structural_system_design
FOR(product_definition_shape,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
```



```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'structural system design definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],

  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'))
  ;
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'structural system design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equals ‘structural system design definition’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘structural system design parameters’.

5.2.4.131 product_definition_shape_for_triangular_cutout_design

The **product_definition_shape_for_triangular_cutout_design** rule specifies that an instance of **product_definition_shape** with class id 'triangular cutout design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'triangular cutout design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_for_triangular_cutout_design
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
```



```
  END_LOCAL;
  (* get all classification_assignment instances with id 'triangular cutout
  design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'triangular cutout design
  definition');
  (* get all instances of T1 that have class id 'triangular cutout design
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
     get the property_definition_representation instances that
    are referencing a product_definition_shape instance via definition,
     filter out those property_definition_representation instances
    whose attribute name has the value 'triangular cutout design
  parameters'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
    'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
    t2_inst.name = 'triangular cutout design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'triangular cutout design definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'triangular cutout design parameters'.

5.2.4.132 product_definition_shape_for_welded_joint_design_parameters

The **product_definition_shape_for_welded_joint_design_parameters** rule specifies that an instance of **product_definition_shape** with class id 'welded joint design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'welded joint design parameters'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_for_welded_joint_design_parameters
FOR(product_definition_shape,
   property_definition_representation, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'welded joint design definition');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
   'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
       'welded joint design parameters')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equals ‘welded joint design definition’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_-definition_representation** attribute **name** equals ‘welded joint design parameters’.

5.2.4.133 product_definition_shape_has_references_for_centre_of_gravity

The **product_definition_shape_has_references_for_centre_of_gravity** rule specifies that an instance of **product_definition_shape** with class id 'assembly manufacturing definition' is referenced by exactly one instance of **shape_aspect** with class id 'centre of gravity' via **of_shape**.

EXPRESS specification:

```
*)
RULE
product_definition_shape_has_references_for_centre_of_gravity
FOR(product_definition_shape,
   shape_aspect, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF shape_aspect := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'assembly manufacturing definition');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
      'centre of gravity' IN WHICH_CLASS(t2_inst))) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'assembly manufacturing definition' is referenced by exactly one instance of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'centre of gravity' through attribute **of_shape**.

5.2.4.134 product_definition_shape_has_references_for_weld_design

The **product_definition_shape_has_references_for_weld_design** rule specifies that an instance of **product_definition_shape** with class id 'weld design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'weld design parameters'.

EXPRESS specification:

```
*)
RULE  product_definition_shape_has_references_for_weld_design
FOR(product_definition_shape,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition_shape := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'weld design
  definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'weld design definition');
  (* get all instances of T1 that have class id 'weld design definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition_shape in t1_set:
     get the property_definition_representation instances that are
  referencing a product_definition_shape instance via definition,
     filter out those property_definition_representation instances
  whose attribute name has the value 'weld design parameters'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
        'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
              t2_inst.name = 'weld design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'weld design definition', is referenced by exactly one **property_definition_-representation.definition**, where **property_definition_representation.name** equals 'weld design parameters'.

5.2.4.135 product_definition_shape_has_references_with_class_border

The **product_definition_shape_has_references_with_class_border** rule specifies that an instance of **product_definition_shape** with class id 'corrugated part design definition' is referenced by exactly two instances of **shape_aspect** with class id 'boundary' via **of_shape**.

EXPRESS specification:

```
*)
RULE product_definition_shape_has_references_with_class_border
FOR(product_definition_shape,
   shape_aspect, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF shape_aspect := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'corrugated part design definition');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     'boundary' IN WHICH_CLASS(t2_inst))) = 2);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances of **shape_aspect** entities.
group: the set of all instances of **group** entities
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'corrugated part design definition' is referenced by exactly two instances of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'boundary' through attribute **of_shape**.

5.2.4.136 product_definition_shape_of_rectangular_cutback_corner_-design

The **product_definition_shape_of_rectangular_cutback_corner_design** rule specifies that an instance of **product_definition_shape** with class id 'rectangular cutback corner design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'rectangular cutback corner design parameters'.

EXPRESS specification:

```
*)
RULE
 product_definition_shape_of_rectangular_cutback_corner_design
FOR(product_definition_shape,
   property_definition_representation,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   (* get all classification_assignment instances with id 'rectangular
   cutback corner design definition' *)
   c_a_set := QUERY(i <* applied_classification_assignment |
      i.ASSIGNED_CLASS.NAME = 'rectangular cutback corner design
   definition');
   (* get all instances of T1 that have class id 'rectangular cutback corner
   design definition' *)
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   (* for all instances of product_definition_shape in t1_set:
      get the property_definition_representation instances that
     are referencing a product_definition_shape instance via definition,
      filter out those property_definition_representation instances
     whose attribute name has the value
     'rectangular cutback corner design parameters'
      check if their number equals 1
    *)
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
     t2_inst.name = 'rectangular cutback corner design parameters')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition_shape** that is referenced by an **applied_classification_assignment** with **name** equal to 'rectangular cutback corner design definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'rectangular cutback corner design parameters'.

5.2.4.137 product_definition_shape_with_definition_matches_pattern

The **product_definition_shape_with_definition_matches_pattern** rule specifies a string type attribute **product_definition_shape** of type 'definition' to match the pattern '*.*' if the instance of **product_definition_shape** is of class 'definition'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_with_definition_matches_pattern
FOR (property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    clfied_inst :SET OF GENERIC := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'DEFINITION') THEN
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
  END_IF;
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
property_definition_representation: the set of all instances of **property_definition_representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **product_definition_shape** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'definition' is to match the pattern '*.*'.

5.2.4.138 product_definition_shape_with_identification_assignment

The **product_definition_shape_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE product_definition_shape_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set  : SET OF applied_classification_assignment := [];
    t1_set   : SET OF product_definition_shape := [];
    t2_set   : SET OF product_definition_shape := [];
    arg_list : LIST OF STRING := ['explicit feature design definition',
  'assembly manufacturing definition',
  'structural added material feature design definition',
  'structural part manufacturing definition',
  'corrugated part design definition',
  'plate',
  'plate design definition',
  'profile design definition',
  'weld manufacturing definition',
  'bevel design definition',
  'circular cutout design definition',
  'drain hole cutout design definition',
  'elliptical cutout design definition',
  'elongated oval cutout design definition',
  'free form interior cutout design definition',
  'inward round corner design definition',
  'outward round corner design definition',
  'part edge cutout design definition',
  'rectangular cutback corner design definition',
  'rectangular cutout design definition',
  'round edge rectangular cutout design definition',
  'seam design definition',
  'shear bevel design definition',
  'triangular cutout design definition'];
   violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id  *)
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =  arg_LIST[j]);
   END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
```



```
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
   'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
   END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'explicit feature design definition' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.139 product_definition_shape_with_versionable_object

The **product_definition_shape_with_versionable_object** rule specifies a string type attribute **product_definition_shape** of type 'versionable object' to match the pattern '*.*' if the instance of **product_definition_shape** is of class 'versionable object'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_with_versionable_object
FOR (group, applied_classification_assignment);
  LOCAL
    clfied_inst :SET OF GENERIC := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'versionable object') THEN
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
END_IF;
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **product_definition_shape** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'versionable object' is to match the pattern '*.*'.

5.2.4.140 product_definition_shape_with_versionable_object_pattern

The **product_definition_shape_with_versionable_object_pattern** rule specifies a string type attribute **product_definition_shape** of type 'versionable object' to match the pattern '*.*' if the instance of **product_definition_shape** is of class 'versionable object'.

EXPRESS specification:

```
*)
RULE
product_definition_shape_with_versionable_object_pattern
FOR (group, applied_classification_assignment);
  LOCAL
    clfied_inst :SET OF GENERIC := [];
  END_LOCAL;

  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'versionable object') THEN
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
END_IF;
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **product_definition_shape** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'versionable object' is to match the pattern '*.*'.

5.2.4.141 product_definition_with_date_freeboard_assigned

The **product_definition_with_date_freeboard_assigned** rule specifies that an instance of **product_definition** with class id 'freeboard characteristics' is referenced by exactly one instance of **applied_date_and_time_assignment** via **items** whose attribute **role** has the value 'date freeboard assigned'.

EXPRESS specification:

```
*)
RULE  product_definition_with_date_freeboard_assigned
FOR(product_definition,
   applied_date_and_time_assignment,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition := [];
     t2_set    : SET OF applied_date_and_time_assignment := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   (* get all classification_assignment instances with id 'freeboard
   characteristics' *)
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.ASSIGNED_CLASS.NAME = 'freeboard characteristics');
   (* get all instances of T1 that have class id 'freeboard characteristics'
   *)
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   (* for all instances of product_definition in t1_set:
      get the applied_date_and_time_assignment instances
   that are referencing a product_definition instance via items,
      filter out those applied_date_and_time_assignment instances
   whose attribute role has the value 'date freeboard assigned'
      check if their number equals 1
   *)
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
          'APPLIED_DATE_AND_TIME_ASSIGNMENT.ITEMS'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.role.name =
                         'date freeboard assigned')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
applied_date_and_time_assignment: the set of all instances of **applied_date_and_time_-assignment** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'freeboard characteristics', is referenced by exactly one **applied_date_and_time_-assignment.items**, where **applied_date_and_time_assignment.role** equals 'date freeboard assigned'.

5.2.4.142 product_definition_with_definable_object_matches_pattern

The **product_definition_with_definable_object_matches_pattern** rule specifies a string type attribute **product_definition** of type 'definable object' to match the pattern '*.*' if the instance of **product_definition** is of class 'definable object'.

EXPRESS specification:

```
*)
RULE
product_definition_with_definable_object_matches_pattern
FOR (product_definition, group, applied_classification_assignment);
  LOCAL
    clfied_inst :SET OF GENERIC := [];
  END_LOCAL;

  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'definable object') THEN
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
  END_IF;
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **product_definition** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'definable object' is to match the pattern '*.*'.

5.2.4.143 product_definition_with_freeboard_assigned_by

The **product_definition_with_freeboard_assigned_by** rule specifies that an instance of **product_-definition** with class id 'freeboard characteristics' is referenced by exactly one instance of **applied_-organization_assignment** via **items** whose attribute role has the value 'freeboard assigned by'.

EXPRESS specification:

```
*)
RULE  product_definition_with_freeboard_assigned_by
FOR(product_definition,
  applied_organization_assignment,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
```



```
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF applied_organization_assignment := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'freeboard
  characteristics' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'freeboard characteristics');
  (* get all instances of T1 that have class id 'freeboard characteristics'
  *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of product_definition in t1_set:
     get the applied_organization_assignment instances that are referencing
  a product_definition instance via items,
     filter out those applied_organization_assignment instances
  whose attribute role has the value 'freeboard assigned by'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.' + 'APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.role.name =
                       'freeboard assigned by')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition: the set of all instances of **product_definition** entities.
applied_organization_assignment: the set of all instances of **applied_organization_assignment** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **product_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'freeboard characteristics', is referenced by exactly one **applied_organization_-assignment.items**, where **applied_organization_assignment.role** equals 'freeboard assigned by'.

5.2.4.144 product_definition_with_identification_assignment

The **product_definition_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE product_definition_with_identification_assignment
FOR (applied_classification_assignment);
```

```
  LOCAL
    c_a_set  : SET OF applied_classification_assignment := [];
    t1_set   : SET OF product_definition := [];
    t2_set   : SET OF product_definition := [];
    arg_list : LIST OF STRING := [ 'principal characteristics',
                                   'class and statutory designation',
                                   'freeboard characteristics',
                                   'lightship definition',
                                   'ship designation',
                                   'shipyard designation',
                                   'owner designation',
                                   'assembly',
                                   'corrugated part',
                                   'plate boundary relationship',
                                   'profile',
                                   'versionable object'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id  *)
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =  arg_LIST[j]);
   END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
    END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'principal characteristics', 'class and statutory designation', 'ship designation', 'shipyard designation', 'owner designation' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.145 product_related_product_category_with_identification_assignment

The **product_related_product_category_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE product_related_product_category_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_related_product_category := [];
    t2_set    : SET OF product_related_product_category := [];
    arg_list  : LIST OF STRING := [ 'ship type'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id  *)
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =  arg_LIST[j]);
   END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
    END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_related_product_category** that is referenced by an **applied_classification_assignment** whose assigned_classification has a **name** attribute of value 'ship type' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.146 product_with_identification_assignment

The **product_with_identification_assignment** rule specifies a string type attribute **product** of type 'definable object' to match the pattern 'definable object' if the instance of **product** is of class 'definable object'.

EXPRESS specification:

```
*)
RULE product_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product := [];
    t2_set    : SET OF product := [];
    arg_list  : LIST OF STRING := ['definable object'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'definable object'
  *)
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =  arg_LIST[j]);
   END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
    END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'definable object' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.147 profile_cross_section_has_at_most_one_named_references

The **profile_cross_section_has_at_most_one_named_references** rule specifies that an instance of **shape_aspect** with class id 'profile cross section' is referenced by at most one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'section properties'.

EXPRESS specification:

```
*)
RULE
profile_cross_section_has_at_most_one_named_references
FOR(shape_aspect,
   property_definition_representation, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET Of shape_aspect := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'profile cross section');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
   'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
     violation := SIZEOF(QUERY(t2_inst <* t2_set |
       t2_inst.name = 'section properties')) > 1;
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances **of shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:
WR1: Every instance of **shape_aspect** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'profile cross section' is referenced by one or less instances of **property_definition_representation** whose attribute **name** has a value of 'section properties' through attribute **definition**.

5.2.4.148 profile_design_definition_has_at_most_one_class_references

The **profile_design_definition_has_at_most_one_class_references** rule specifies that an instance of **product_definition_shape** with class id 'profile design definition' is referenced by at most one instance of **shape_aspect** with class id 'profile curve trace line' via **of_shape**.

EXPRESS specification:

```
*)
RULE
profile_design_definition_has_at_most_one_class_references
FOR(product_definition_shape,
   shape_aspect, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET Of shape_aspect := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'profile design definition');

   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
     violation := SIZEOF(QUERY(t2_inst <* t2_set |
       'profile curve trace line' IN WHICH_CLASS(t2_inst))) > 1;
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with attribute **name** equal to 'profile design definition' is referenced by one or less instances of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with attribute **name** equal to 'profile curve trace line' through attribute **of_shape**.

5.2.4.149 profile_design_definition_has_zero_or_two_class_references

The **profile_design_definition_has_zero_or_two_class_references** rule specifies that an instance of **product_definition_shape** with class id 'profile design definition' is referenced by exactly zero or two instances of **shape_aspect** with class id 'boundary' via **of_shape**.

EXPRESS specification:

```
*)
RULE
profile_design_definition_has_zero_or_two_class_references
FOR(product_definition_shape,
   shape_aspect, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF product_definition_shape := [];
     t2_set    : SET Of shape_aspect := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'profile design definition');

   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
       'boundary' IN WHICH_CLASS(t2_inst))) = 0) AND
                  NOT (SIZEOF(QUERY(t2_inst <* t2_set |
                    'boundary' IN WHICH_CLASS(t2_inst))) = 2);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

product_definition_shape: the set of all instances of **product_definition_shape** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition_shape** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'profile design definition' is referenced by either zero or two instances of **shape_aspect** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'boundary' through attribute **of_shape**.

5.2.4.150 property_definition_for_class_notation

The **property_definition_for_class_notation** rule specifies that an instance of **property_definition** with class id 'class notation' is referenced by exactly one instance of **property_definition_-representation** via **definition** whose attribute **name** has the value 'class notation'.

EXPRESS specification:

```
*)
RULE
property_definition_for_class_notation
FOR(property_definition,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'class notation');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'class notation')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities
property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'class notation', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'class notation'.

5.2.4.151 property_definition_for_class_society

The **property_definition_for_class_society** rule specifies that an instance of **property_definition** with class id 'class notation' is referenced by exactly one instance of **applied_organization_-assignment** via **items** whose attribute **role** has the value 'class society'.

EXPRESS specification:

```
*)
RULE
property_definition_for_class_society
FOR(property_definition,
  applied_organization_assignment, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF applied_organization_assignment := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME = 'class notation');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.role.name =
      'class society')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
property_definition: the set of all instances of **property_definition** entities.
applied_organization_assignment: the set of all instances of **applied_organization_assignment** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'class notation', is referenced by exactly one **applied_organization_assignment** attribute **items**, where **applied_organization_assignment** attribute **role** equals 'class society'.

5.2.4.152 property_definition_for_corrugated_structure_function

The **property_definition_for_corrugated_structure_function** rule specifies that an instance of **property_definition** with class id 'corrugated structure functional definition' is referenced by exactly one instance of **property_definition_representation** via **definition** which has an attribute **name** has the value 'corrugated structure function parameters'.

EXPRESS specification:

```
*)
RULE
property_definition_for_corrugated_structure_function
FOR(property_definition,
   property_definition_representation, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF property_definition := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME =
                    'corrugated structure functional definition');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.'+
         'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
       'corrugated structure function parameters')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with attribute **name** equals ‘corrugated structure functional definition’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_-representation** attribute **name** equals ‘corrugated structure function parameters’.

5.2.4.153 property_definition_for_edge_cutout_function_parameters

The **property_definition_for_edge_cutout_function_parameters** rule specifies that an instance of **property_definition** with class id 'edge cutout functional definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'edge cutout function parameters'.

EXPRESS specification:

```
*)
RULE  property_definition_for_edge_cutout_function_parameters
FOR(property_definition,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'edge cutout
  functional definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'edge cutout functional
  definition');
  (* get all instances of T1 that have class id 'edge cutout functional
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances that are
    referencing a property_definition instance via definition,
     filter out those property_definition_representation instances
     whose attribute name has the value 'edge cutout function parameters'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
     'edge cutout function parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'edge cutout functional definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'edge cutout function parameters'.

5.2.4.154 property_definition_for_edge_feature_function_parameters

The **property_definition_for_edge_feature_function_parameters** rule specifies that an instance of **property_definition** with class id 'edge feature functional definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'edge feature function parameters'.

EXPRESS specification:

```
*)
RULE  property_definition_for_edge_feature_function_parameters
FOR(property_definition,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'edge feature
  functional definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
             i.ASSIGNED_CLASS.NAME = 'edge feature functional definition');
  (* get all instances of T1 that have class id 'edge feature functional
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances that are
  referencing
    a property_definition instance via definition,
     filter out those property_definition_representation instances
    whose attribute name has the value 'edge feature function parameters'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
    t2_inst.name = 'edge feature function parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'edge feature functional definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'edge feature function parameters'.

5.2.4.155 property_definition_for_homogeneous_ship_material_property

The **property_definition_for_homogeneous_ship_material_property** rule specifies that an instance of **property_definition** with class id 'homogeneous ship material property' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'homogeneous ship material property parameters'.

EXPRESS specification:

```
*)
RULE
property_definition_for_homogeneous_ship_material_property
FOR(property_definition,
   property_definition_representation, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF property_definition := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME = 'homogeneous ship material
   property');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;

   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i],
   'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
       'homogeneous ship  material property parameters')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'homogeneous ship material property', is referenced by exactly one **property_-definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'homogeneous ship material property parameters'.

5.2.4.156 property_definition_for_interior_cutout_function_parameters

The **property_definition_for_interior_cutout_function_parameters** rule specifies that an instance of **property_definition** with class id 'interior cutout functional definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'interior cutout function parameters'.

EXPRESS specification:

```
*)
RULE  property_definition_for_interior_cutout_function_parameters
FOR(property_definition,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'interior cutout
  functional definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
          i.ASSIGNED_CLASS.NAME = 'interior cutout functional definition');
  (* get all instances of T1 that have class id 'interior cutout functional
  definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances that
     are referencing a property_definition instance via definition,
     filter out those property_definition_representation instances
     whose attribute name has the value 'interior cutout function
  parameters'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
```



```
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
  'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
    t2_inst.name = 'interior cutout function parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'interior cutout functional definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'interior cutout function parameters'.

5.2.4.157 property_definition_for_local_co_ordinate_system

The **property_definition_for_local_co_ordinate_system** rule specifies that an instance of **property_definition** with class id 'local coordinate system' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'local coordinate system'.

EXPRESS specification:

```
*)
RULE  property_definition_for_local_coordinate_system
FOR(property_definition,
  property_definition_representation, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'local coordinate
  system' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'local coordinate system');
  (* get all instances of T1 that have class id 'local coordinate system'
  *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
```

```
   (* for all instances of property_definition in t1_set:       get the
   property_definition_representation instances that are referencing a
   property_definition instance via definition, filter out those
   property_definition_representation instances whose attribute name has
   the value 'local coordinate system'
      check if their number equals 1
    *)
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
   'PROPERTY_DEFINITION_REPRESENTATION' + '.' + 'DEFINITION'));
     violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
   'local coordinate system')) = 1);
   END_REPEAT;
   WHERE
     WR1: NOT violation;
   END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'local coordinate system', is referenced by exactly one **property_definition_-representation.definition**, where **property_definition_representation.name** equals 'local coordinate system'.

5.2.4.158 property_definition_for_local_co_ordinate_system_with_position

The **property_definition_for_local_co_ordinate_system_with_position** rule specifies that an instance of **property_definition** with class id 'local coordinate system with position reference' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'local coordinate system with position reference'.

EXPRESS specification:

```
*)
RULE
property_definition_for_local_co_ordinate_system_with_position
FOR(property_definition,
   property_definition_representation, group,
   applied_classification_assignment);
   LOCAL
     c_a_set   : SET OF applied_classification_assignment := [];
     t1_set    : SET OF property_definition := [];
     t2_set    : SET OF property_definition_representation := [];
     violation : LOGICAL := FALSE;
   END_LOCAL;
```



```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'local coordinate system with position reference');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'local coordinate system with position reference')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every property_definition that is referenced by an **applied_classification_assignment** with **name** equal to 'local coordinate system with position reference', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_-representation** attribute **name** equals 'local coordinate system with position reference'.

5.2.4.159 property_definition_for_panel_system_function_parameters

The **property_definition_for_panel_system_function_parameters** rule specifies that an instance of **property_definition** with class id 'panel system functional definition' is referenced by exactly one instance of **property_definition_representation** via **definition** which has an attribute **name** has the value 'panel system function parameters'.

EXPRESS specification:

```
*)
RULE
property_definition_for_panel_system_function_parameters
FOR(property_definition,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
```

```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'panel system functional definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'panel system function parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equals ‘panel system functional definition’, is referenced by exactly one **property_-definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘panel system function parameters’.

5.2.4.160 property_definition_for_plate_function_parameters

The **property_definition_for_plate_function_parameters** rule specifies that an instance of **property_definition** with class id 'plate functional definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'plate function parameters'.

EXPRESS specification:

```
*)
RULE
property_definition_for_plate_function_parameters
FOR(property_definition,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
```



```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'plate functional definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'plate function parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'plate functional definition', is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'plate function parameters'.

5.2.4.161 property_definition_for_plate_strake_function_parameters

The **property_definition_for_plate_strake_function_parameters** rule specifies that an instance of **property_definition** with class id 'plate strake functional definition' is referenced by exactly one instance of **property_definition_representation** via **definition** which has an attribute **name** has the value 'plate strake function parameters'.

EXPRESS specification:

```
*)
RULE
property_definition_for_plate_strake_function_parameters
FOR(property_definition,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
```

```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'plate strake functional definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'plate strake function parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with attribute **name** equals ‘plate strake functional definition’, is referenced by exactly one **property_-definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘plate strake function parameters’.

5.2.4.162 property_definition_for_profile_function_parameters

The **property_definition_for_profile_function_parameters** rule specifies that an instance of **property_definition** with class id 'profile functional definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'profile function parameters'.

EXPRESS specification:

```
*)
RULE
property_definition_for_profile_function_parameters
FOR(property_definition,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
```



```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'profile functional definition');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'profile function parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with attribute **name** equal to 'profile functional definition', is referenced by exactly one **property_-definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'profile function parameters'.

5.2.4.163 property_definition_for_ship_material_property_parameters

The **property_definition_for_ship_material_property_parameters** rule specifies that an instance of **property_definition** with class id 'ship material property' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'ship material property parameters'.

EXPRESS specification:

```
*)
RULE
property_definition_for_ship_material_property_parameters
FOR(property_definition,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
```

```
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'ship material property');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;

  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'ship material property parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with attribute **name** equal to 'ship material property', is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'ship material property parameters'.

5.2.4.164 property_definition_for_structural_system_function_parameters

The **property_definition_for_structural_system_function_parameters** rule specifies that an instance of **property_definition** with class id 'structural system functional definition' is referenced by exactly one instance of **property_definition_representation** via **definition** that has an attribute **name** with the value 'structural system function parameters'.

EXPRESS specification:

```
*)
RULE
property_definition_for_structural_system_function_parameters
FOR(property_definition,
  property_definition_representation,
  applied_classification_assignment);
  (*
  R19<PROPERTY_DEFINITION, 'structural system functional definition', 1,
  PROPERTY_DEFINITION_REPRESENTATION, DEFINITION, NAME, 'structural system
  function parameters'> *)
  LOCAL
```



```
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with
       id 'structural system functional definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'structural system functional
  definition');
  (* get all instances of T1 that have class id 'structural system
  functional definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances
    that are referencing a property_definition instance via definition,
     filter out those property_definition_representation instances
     which has an attribute name has the value 'structural system function
  parameters'
     check if their number equals 1   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
          t2_inst.name = 'structural system function parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'structural system functional definition', is referenced by exactly one **property_-definition_representation.definition**, where **property_definition_representation.name** equals 'structural system function parameters'.

5.2.4.165 property_definition_for_weld_filler_material_design

The **property_definition_for_weld_filler_material_design** rule specifies that an instance of **product_definition_shape** with class id 'weld filler material' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'weld filler material design parameters'.

EXPRESS specification:

```
*)
RULE  property_definition_for_weld_filler_material_design
FOR(property_definition,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances *)
  (* with id 'weld filler material' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'weld filler material');
  (* get all instances of T1 that have class id 'weld filler material' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances that are
     referencing a property_definition instance via definition,
     filter out those property_definition_representation instances
     whose attribute name has the value 'weld filler material
     design parameters' check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
    'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
     'weld filler material design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **product_definition_shape** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'weld filler material', is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'weld filler material design parameters'.

5.2.4.166 property_definition_has_references_with_name_loadline

The **property_definition_has_references_with_name_loadline** rule specifies that an instance of **property_definition** with class id 'loadline' is referenced by exactly one instance of **property_-definition_representation** via **definition** whose attribute **name** has the value 'loadline'.

EXPRESS specification:

```
*)
RULE  property_definition_has_references_with_name_loadline
FOR(property_definition,
   property_definition_representation,
   applied_classification_assignment);
    (*
R19<PROPERTY_DEFINITION,'loadline',1,PROPERTY_DEFINITION_REPRESENTATION,DEF
INITION,NAME,'loadline'> *)
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'loadline' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'loadline');
  (* get all instances of T1 that have class id 'loadline' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances that
    are referencing a property_definition instance via definition,
     filter out those property_definition_representation instances
    whose attribute name has the value 'loadline'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
  'loadline')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with attribute **name** equal to 'load line', is referenced by exactly one **property_definition_-representation.definition**, where **property_definition_representation.name** equals 'load line'.

5.2.4.167 property_definition_structural_weld_shrinkage_allowance_-feature

The **property_definition_structural_weld_shrinkage_allowance_feature** rule specifies that an instance of **property_definition** with class id 'structural weld shrinkage allowance feature design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'structural weld shrinkage allowance feature design parameters'.

EXPRESS specification:

```
*)
RULE
 property_definition_structural_weld_shrinkage_allowance_feature
FOR(property_definition,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'structural weld
  shrinkage allowance feature design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'structural weld shrinkage
  allowance feature design definition');
  (* get all instances of T1 that have class id 'structural weld shrinkage
  allowance feature design definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances that are
  referencing a property_definition instance via definition,
  filter out those property_definition_representation instances whose
  attribute name has the value 'structural weld shrinkage allowance
  feature design parameters'
  check if their number equals 1
```



```
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
  'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
  'structural weld shrinkage allowance feature design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with attribute **name** equal to 'structural weld shrinkage allowance feature design definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'structural weld shrinkage allowance feature design parameters'.

5.2.4.168 property_definition_to_structural_added_material_feature_-design

The **property_definition_to_structural_added_material_feature_design** rule specifies that an instance of **property_definition** with class id 'structural added material feature design definition' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'structural added material feature design parameters'.

EXPRESS specification:

```
*)
RULE  property_definition_to_structural_added_material_feature_design
FOR(property_definition,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL  := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'structural added
  material feature design definition' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'structural added material
  feature design definition');
```

```
  (* get all instances of T1 that have class id 'structural added material
  feature design definition' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances that are
  referencing a property_definition instance via definition,
     filter out those property_definition_representation instances whose
  attribute name has the value 'structural added material feature design
  parameters'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
  'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
  'structural added material feature design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equal to 'structural added material feature design definition', is referenced by exactly one **property_definition_representation.definition**, where **property_definition_representation.name** equals 'structural added material feature design parameters'.

5.2.4.169 property_definition_with_definition_matches_pattern

The **property_definition_with_definition_matches_pattern** rule specifies a string type attribute **property_definition** of type 'definition' to match the pattern '*.*' if the instance of **property_-definition** is of class 'definition'.

EXPRESS specification:

```
*)
RULE
property_definition_with_definition_matches_pattern
FOR (group, applied_classification_assignment);
  LOCAL
    clfied_inst :SET OF GENERIC := [];
  END_LOCAL;
```



```
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'DEFINITION') THEN
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
   END_IF;
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **property_definition** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'definition' is to match the pattern '*.*'.

5.2.4.170 property_definition_with_identification_assignment

The **property_definition_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE property_definition_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set  : SET OF applied_classification_assignment := [];
    t1_set   : SET OF property_definition := [];
    t2_set   : SET OF property_definition := [];
    arg_list : LIST OF STRING := [ 'moulded form characteristics
   definition',
                                 'ship overall dimensions',
                                 'local coordinate system',
                                 'spacing table',
                                 'global axis placement',
                                 'hydrostatic definition',
                                 'stability definition',
                                 'plate functional definition',
                                 'profile functional definition',
                                 'edge cutout functional definition',
                                 'edge feature functional definition',
                                 'interior cutout functional definition',
        'structural weld shrinkage allowance  feature design definition'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
```

```
        c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =  arg_LIST[j]);
   END_REPEAT;
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
    END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **property_definition** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'moulded form characteristics definition', 'ship overall dimensions', 'local coordinate system', 'spacing table', 'global axis placement' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.171 property_definition_has_references_lightship_weight_item

The **property_definition_has_references_lightship_weight_item** rule specifies that an instance of **property_definition** with class id 'lightship weight item' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'lightship weight item'.

EXPRESS specification:

```
*)
RULE  property_definition_has_references_lightship_weight_item
FOR(property_definition,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'lightship weight
  item' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'lightship weight item');
  (* get all instances of T1 that have class id 'lightship weight item' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
```



```
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances
  that are referencing a property_definition instance via definition,
     filter out those property_definition_representation instances
  whose attribute name has the value 'lightship weight item'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
                        'lightship weight item')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'lightship weight item', is referenced by exactly one **property_definition_-representation.definition**, where **property_definition_representation.name** equals 'lightship weight item'.

5.2.4.172 property_definition_with_versionable_object_matches_pattern

The **property_definition_with_versionable_object_matches_pattern** rule specifies a string type attribute **property_definition** of type 'versionable object' to match the pattern '*.*' if the instance of **property_definition** is of class 'versionable object'.

EXPRESS specification:

```
*)
RULE
property_definition_with_versionable_object_matches_pattern
FOR (group, applied_classification_assignment);
  LOCAL
    clfied_inst :SET OF GENERIC := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'versionable object') THEN
```

```
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
  END_IF;
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **property_definition** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'versionable object' is to match the pattern '*.*'.

5.2.4.173 property_definition_with_weight_and_centre_of_gravity

The **property_definition_with_weight_and_centre_of_gravity** rule specifies that an instance of **property_definition** with class id 'weight and centre of gravity' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'weight and centre of gravity'.

EXPRESS specification:

```
*)
RULE  property_definition_with_weight_and_centre_of_gravity
FOR(property_definition,
  property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'weight and centre
  of gravity' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'weight and centre of gravity');
  (* get all instances of T1 that have class id 'weight and centre of
  gravity' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the property_definition_representation instances
     that are referencing a property_definition instance via definition,
     filter out those property_definition_representation instances
     whose attribute name has the value 'weight and centre of gravity'
```



```
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.' +
    'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
                   'weight and centre of gravity')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equal to ‘weight and centre of gravity’, is referenced by exactly one **property_definition_-representation.definition**, where **property_definition_representation.name** equals ‘weight and centre of gravity’.

5.2.4.174 representation_design_vertical_wave_bending_moment

The **representation_design_vertical_wave_bending_moment** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'design vertical wave bending moment parameters'.

EXPRESS specification:

```
*)
RULE
representation_design_vertical_wave_bending_moment
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['hogging', 'sagging'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
     to 'design vertical wave bending moment parameters' *)
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
```

```
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'+
  '.USED_REPRESENTATION'))
| (temp_prop_def_rep.name = 'design vertical wave bending moment
  parameters')
                    )
              ) > 0
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘design vertical wave bending moment parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'hogging' or 'sagging'.

5.2.4.175 representation_for_angle_bar_cross_section_dimensions

The **representation_for_angle_bar_cross_section_dimensions** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list, exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'angle bar cross section dimensions'.

EXPRESS specification:

```
*)
RULE representation_for_angle_bar_cross_section_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'width', 'radius', 'thk'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,

  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
```



```
              'USED_REPRESENTATION'))
            | (temp_prop_def_rep.name = 'angle bar cross section
  dimensions')
                    )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'angle bar cross section dimensions', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'depth', 'width', 'radius', or 'thk'.

5.2.4.176 representation_for_assembly_manufacturing_definition

The **representation_for_assembly_manufacturing_definition** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'assembly manufacturing definition parameters'.

EXPRESS specification:

```
*)
RULE
representation_for_assembly_manufacturing_definition
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING :=
              ['assembly method','assembly stage','mass','center of
  gravity'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
```

```
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
                'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
    | (temp_prop_def_rep.name = 'assembly manufacturing definition
  parameters')
                      )
                ) > 0
            );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with attribute **name** equal to 'assembly manufacturing definition parameters', is to use a **representation** that has exactly one **item** in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'assembly method', 'assembly stage', 'mass', or 'center of gravity'.

5.2.4.177 representation_for_bulbflat_cross_section_dimensions

The **representation_for_bulbflat_cross_section_dimensions** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'bulbflat cross section dimensions'.

EXPRESS specification:

```
*)
RULE representation_for_bulbflat_cross_section_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'bulb width', 'bulb radius',
  'tw'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
```



```
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,

  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                                   'bulbflat cross section dimensions')
                  )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘bulbflat cross section dimensions’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'depth', 'bulb width', 'bulb radius', or 'tw'.

5.2.4.178 representation_for_channel_cross_section_dimensions

The **representation_for_channel_cross_section_dimensions** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** that has a **name** attribute has a value 'channel cross section dimensions'.

EXPRESS specification:

```
*)
RULE representation_for_channel_cross_section_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'width', 'web thk',
                                   'flange thk', 'radius', 'k'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
```

```
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,

  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                                          'channel cross section
  dimensions')
                  )
            ) > 0
         );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with attribute **name** equal to 'channel cross section dimensions', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'depth', 'width', 'web thk', 'flange thk', 'radius', or 'k'.

5.2.4.179 representation_for_circular_cutout_design_parameters

The **representation_for_circular_cutout_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'circular cutout design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_circular_cutout_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['radius'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
```



```
     by a property_definition_representation with name equal
    to 'circular cutout design parameters' *)
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'+
    '.USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name = 'circular cutout design
  parameters')
                )
             ) > 0
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘circular cutout design parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'radius'.

5.2.4.180 representation_for_circular_hollow_profile_cross_section

The **representation_for_circular_hollow_profile_cross_section** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'circular hollow profile cross section dimensions'.

EXPRESS specification:

```
*)
RULE representation_for_circular_hollow_profile_cross_section
FOR (representation, representation_item,
  property_definition_representation);
  (*  R10<'circular hollow profile cross section dimensions', ['outer
  diameter', 'inner diameter']> *)
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'width', 'web thk',
```

```
                                    'flange thk', 'radius', 'k'];
    violation : LOGICAL := FALSE;
  END_LOCAL;

  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                                        'channel cross section
  dimensions')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with attribute **name** equal to 'channel cross section dimensions', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'depth', 'width', 'web thk', 'flange thk', 'radius', or 'k'.

5.2.4.181 representation_for_class_and_statutory_designation

The **representation_for_class_and_statutory_designation** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'class and statutory designation'.

EXPRESS specification:

```
*)
RULE
representation_for_class_and_statutory_designation
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['class number'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
              'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                                    'class and statutory designation')
                 )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with attribute **name** equal to 'class and statutory designation', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'class number'.

5.2.4.182 representation_for_corrugated_part_design_parameters

The **representation_for_corrugated_part_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'corrugated part design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_corrugated_part_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['mirrored', 'offset', 'repetition',
                                   'thickness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                                        'corrugated part design
  parameters')
                 )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every property_**definition_representation** with **name** equal to ‘corrugated part design parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'mirrored', 'offset', 'repetition', or 'thickness'.

5.2.4.183 representation_for_design_vertical_wave_shear_force_-parameters

The **representation_for_design_vertical_wave_shear_force_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_-item** whose **name** attribute has the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** whose **name** attribute has a value 'design vertical wave shear force parameters'.

EXPRESS specification:

```
*)
RULE
representation_for_design_vertical_wave_shear_force_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['negative value', 'positive value'];
    violation : LOGICAL := FALSE;
  END_LOCAL;

  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
     to 'design vertical wave shear force parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'+
  '.USED_REPRESENTATION'))
  | (temp_prop_def_rep.name = 'design vertical wave shear force
  parameters')
                  )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with attribute name equal to ‘design vertical wave shear force parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'negative value’ or 'positive value'.

5.2.4.184 representation_for_drain_hole_cutout_design_parameters

The **representation_for_drain_hole_cutout_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'drain hole cutout design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_drain_hole_cutout_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'drain hole radius', 'gap',
   'gap radius', 'width'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
 'drain hole cutout design parameters' *)
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'+
'.USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name = 'drain hole cutout design
parameters')
                )
             ) > 0
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with name equal to 'drain hole cutout design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'depth', 'drain hole radius', 'gap', 'gap radius', or 'width'.

5.2.4.185 representation_for_edge_cutout_function_parameters

The **representation_for_edge_cutout_function_parameters** rule specifies that the items attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'edge cutout function parameters'.

EXPRESS specification:

```
*)
RULE representation_for_edge_cutout_function_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['function'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
  to 'edge cutout function parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
    +'.USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'edge cutout function
  parameters')
                 )
              ) > 0
         );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘edge cutout function parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'function'.

5.2.4.186 representation_for_edge_feature_function_parameters

The **representation_for_edge_feature_function_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'edge feature function parameters'.

EXPRESS specification:

```
*)
RULE representation_for_edge_feature_function_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['function'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
    'edge feature function parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
    +'.USED_REPRESENTATION'))
                  | (temp_prop_def_rep.name = 'edge feature function
parameters')
                    )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'edge feature function parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'function'.

5.2.4.187 representation_for_electrode_chemical_composition_design

The **representation_for_electrode_chemical_composition_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'electrode chemical composition design parameters'.

EXPRESS specification:

```
*)
RULE
representation_for_electrode_chemical_composition_design
FOR (representation, representation_item,
property_definition_representation);
(*
R10X<'electrode chemical
composition',['carbon','silicon','manganese','phosphorus','sulphur','nickel
','molybdenum','aluminium','titanium','zirconium','chromium','copper',
'vanadium']> *)
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['carbon', 'silicon', 'manganese',
                           'phosphorus', 'sulphur', 'nickel', 'molydenum',
                          'aliminium', 'titanium', 'zirconium', 'chromium',
                             'copper', 'vanadium'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,

'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                           'electrode chemical composition design
parameters')
                  )
               ) > 0
            );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'electrode chemical composition design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'carbon', 'silicon', 'manganese', 'phosphorus', 'sulphur', 'nickel', 'molydenum', 'aliminium', 'titanium', 'zirconium', 'chromium', 'copper', or 'vanadium'.

5.2.4.188 representation_for_elliptical_cutout_design_parameters

The **representation_for_elliptical_cutout_design_parameters** rule specifies that the items attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'elliptical cutout design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_elliptical_cutout_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['half axis a', 'half axis b'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
    'elliptical cutout design parameters' *)
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
    +'.USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name = 'elliptical cutout design
parameters')
                  )
               ) > 0
            );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
```



```
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘elliptical cutout design parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'half axis a', or 'half axis b'.

5.2.4.189 representation_for_elongated_oval_cutout_design_parameters

The **representation_for_elongated_oval_cutout_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'elongated oval cutout design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_elongated_oval_cutout_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['distance', 'r1', 'r2'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
   'elongated oval cutout design parameters' *)
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
     +'.USED_REPRESENTATION'))
       | (temp_prop_def_rep.name = 'elongated oval cutout design
parameters')
                  )
               ) > 0
            );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
```

```
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'elongated oval cutout design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'distance', 'r1', or 'r2'.

5.2.4.190 representation_for_explicit_profile_cross_section_shape

The **representation_for_explicit_profile_cross_section_shape** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition**_representation whose **name** attribute has a value 'explicit profile cross section shape'.

EXPRESS specification:

```
*)
RULE
representation_for_explicit_profile_cross_section_shape
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['cross section geometry',
                                   'local coordinate system'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                                    'explicit profile cross section
  shape')
                 )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'explicit profile cross section shape', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'cross section geometry', or 'local coordinate system'.

5.2.4.191 representation_for_flanged_plate_cross_section_dimensions

The **representation_for_flanged_plate_cross_section_dimensions** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'flanged plate cross section dimensions'.

EXPRESS specification:

```
*)
RULE
representation_for_flanged_plate_cross_section_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'width', 'radius', 'thk'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,

  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
              'USED_REPRESENTATION'))
                        | (temp_prop_def_rep.name =
                                    'flanged plate cross section
  dimensions')
                  )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
```

```
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘flanged plate cross section dimensions’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'depth', 'width', 'radius', or 'thk'.

5.2.4.192 representation_for_flat_bar_cross_section_dimensions

The **representation_for_flat_bar_cross_section_dimensions** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'flat bar cross section dimensions'.

EXPRESS specification:

```
*)
RULE
representation_for_flat_bar_cross_section_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'width'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                                  'flat bar cross section dimensions')
                 )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
```



```
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'flat bar cross section dimensions', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'depth', or 'width'.

5.2.4.193 representation_for_global_axis_placement

The **representation_for_global_axis_placement** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'global axis placement'.

EXPRESS specification:

```
*)
RULE representation_for_global_axis_placement
FOR (representation, representation_item,
property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['global axes and origin',
                          'after perpendicular offset', 'orientation'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'global axis placement')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
```

```
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'global axis placement', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_-item** with attribute **name** equal to 'global axes and origin', 'after perpendicular offset', or 'orientation'.

5.2.4.194 representation_for_interior_cutout_function_parameters

The **representation_for_interior_cutout_function_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'interior cutout function parameters'.

EXPRESS specification:

```
*)
RULE representation_for_interior_cutout_function_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['function'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
    to 'interior cutout function parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
     +'.USED_REPRESENTATION'))
         | (temp_prop_def_rep.name = 'interior cutout function parameters')
                    )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
```



```
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘interior cutout function parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'function'.

5.2.4.195 representation_for_inward_round_corner_design_parameters

The **representation_for_inward_round_corner_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'inward round corner design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_inward_round_corner_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['radius', 'x offset', 'y offset'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
   to 'inward round corner design parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
     +'.USED_REPRESENTATION'))
         | (temp_prop_def_rep.name = 'inward round corner design
  parameters')
                  )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
```

```
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘inward round corner design parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'radius', 'x offset', or 'y offset'.

5.2.4.196 representation_for_local_co_ordinate_system

The **representation_for_local_co_ordinate_system** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'local coordinate system'.

EXPRESS specification:

```
*)
RULE representation_for_local_coordinate_system
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['local axes and origin'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name = 'local coordinate system')
                 )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'local coordinate system', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_-item** with attribute **name** equal to 'local axes and origin'.

5.2.4.197 representation_for_outward_round_corner_design_parameters

The **representation_for_outward_round_corner_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'outward round corner design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_outward_round_corner_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['radius', 'x offset', 'y offset'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
    'outward round corner design parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
    +'.USED_REPRESENTATION'))
       | (temp_prop_def_rep.name = 'outward round corner design
  parameters')
                 )
              ) > 0
         );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
```

```
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'outward round corner design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'radius', 'x offset', or 'y offset'.

5.2.4.198 representation_for_part_edge_cutout_design_parameters

The **representation_for_part_edge_cutout_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'part edge cutout design parameters' .

EXPRESS specification:

```
*)
RULE representation_for_part_edge_cutout_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['r left', 'r right', 'x a', 'x b','x c',
                'x d', 'y b', 'y c'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
    to 'part edge cutout design parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
   'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
   +'.USED_REPRESENTATION'))
           | (temp_prop_def_rep.name = 'part edge cutout design parameters')
                 )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
```



```
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                        rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'part edge cutout design parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'r left', 'r right', 'x a', 'x b','x c', 'x d', 'y b', or 'y c'.

5.2.4.199 representation_for_profile_curve_trace_line_design_parameters

The **representation_for_profile_curve_trace_line_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'profile curve trace line design parameters'.

EXPRESS specification:

```
*)
RULE
representation_for_profile_curve_trace_line_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['curve'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'profile curve trace line design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                        rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
```

```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'profile curve trace line design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'curve'.

5.2.4.200 representation_for_profile_function_parameters

The **representation_for_profile_function_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'profile function parameters'.

EXPRESS specification:

```
*)
RULE representation_for_profile_function_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['function'];
    violation : LOGICAL := FALSE;
  END_LOCAL;

  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'profile function parameters')
                  )
              ) > 0
         );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'profile function parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'function'.

5.2.4.201 representation_for_rectangular_cutback_corner_design

The **representation_for_rectangular_cutback_corner_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'rectangular cutback corner design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_rectangular_cutback_corner_design
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['radius', 'x depth', 'y depth'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
     'rectangular cutback corner design parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
    +'.USED_REPRESENTATION'))
  | (temp_prop_def_rep.name = 'rectangular cutback corner design
  parameters')
                 )
              ) > 0
         );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
```

```
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘rectangular cutback corner design parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'radius', 'x depth', or 'y depth'.

5.2.4.202 representation_for_rectangular_cutout_design_parameters

The **representation_for_rectangular_cutout_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'rectangular cutout design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_rectangular_cutout_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['length of', 'width'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
   to 'rectangular cutout design parameters' *)
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
     +'.USED_REPRESENTATION'))
   | (temp_prop_def_rep.name = 'rectangular cutout design parameters')
                  )
               ) > 0
            );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
```



```
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘rectangular cutout design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'length of', or 'width'.

5.2.4.203 representation_for_round_bar_cross_section_dimensions

The **representation_for_round_bar_cross_section_dimensions** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'round bar cross section dimensions'.

EXPRESS specification:

```
*)
RULE representation_for_round_bar_cross_section_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['radius'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'round bar cross section dimensions')
                 )
              ) > 0
         );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘round bar cross section dimensions’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'radius'.

5.2.4.204 representation_for_round_corner_rectangular_cutout_design

The **representation_for_round_corner_rectangular_cutout_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'round corner rectangular cutout design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_round_corner_rectangular_cutout_design
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['corner radius', 'length of', 'width'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
    'round corner rectangular cutout design parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
  +'.USED_REPRESENTATION'))| (temp_prop_def_rep.name =
    'round corner rectangular cutout design parameters')
                  )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
```



```
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘round corner rectangular cutout design parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'corner radius', 'length of', or 'width'.

5.2.4.205 representation_for_round_edge_rectangular_cutout_design

The **representation_for_round_edge_rectangular_cutout_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'round edge rectangular cutout design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_round_edge_rectangular_cutout_design
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['distance', 'edge radius', 'length of',
  'width'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
     'round edge rectangular cutout design parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
  +'.USED_REPRESENTATION'))       | (temp_prop_def_rep.name =
      'round edge rectangular cutout design parameters')
                    )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
```

```
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘round edge rectangular cutout design parameters’, is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'distance', 'edge radius', 'length of', or 'width'.

5.2.4.206 representation_for_shear_bevel_design_parameters

The **representation_for_shear_bevel_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'shear bevel design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_shear_bevel_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['offset', 'x y angle', 'x z angle'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
    'shear bevel design parameters' *)
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
  +'.USED_REPRESENTATION'))
                  | (temp_prop_def_rep.name = 'shear bevel design
  parameters')
                    )
               ) > 0
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
```



```
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'shear bevel design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'offset', 'x y angle', or 'x z angle'.

5.2.4.207 representation_for_square_tube_cross_section_dimensions

The **representation_for_square_tube_cross_section_dimensions** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'square tube cross section dimensions'.

EXPRESS specification:

```
*)
RULE representation_for_square_tube_cross_section_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'width', 'wall thk'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'square tube cross section dimensions')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
```

```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'square tube cross section dimensions', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'depth', 'width', or 'wall thk'.

5.2.4.208 representation_for_structural_added_material_feature_design

The **representation_for_structural_added_material_feature_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'structural added material feature design parameter'.

EXPRESS specification:

```
*)
RULE
representation_for_structural_added_material_feature_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['added_material_length'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                      'structural added material feature '+
                      'design parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
```



```
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'structural added material feature design definition parameter representation', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'added_-material_length'.

5.2.4.209 representation_for_structural_part_joint_design

The **representation_for_structural_part_joint_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'structural part joint design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_structural_part_joint_design
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['tightness','joint orientation'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
     'structural part joint design parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'structural part joint
  design parameters')
                  )
              ) > 0
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
```

```
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'structural part joint design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'tightness', or 'joint orientation'.

5.2.4.210 representation_for_structural_part_symmetry_relationship

The **representation_for_structural_part_symmetry_relationship** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'structural part symmetry relationship design parameters'.

EXPRESS specification:

```
*)
RULE
representation_for_structural_part_symmetry_relationship
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['mirroring plane'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'structural part symmetry relationship design
  parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
```



```
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'structural part symmetry relationship design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'mirroring plane'.

5.2.4.211 representation_for_structural_system_symmetry_relationship

The **representation_for_structural_system_symmetry_relationship** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute has the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute has a value 'structural system symmetry relationship design parameters'.

EXPRESS specification:

```
*)
RULE
representation_for_structural_system_symmetry_relationship
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['mirroring plane'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                 'structural system symmetry relationship design
  parameters')
                  )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
```

```
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'structural system symmetry relationship design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'mirroring plane'.

5.2.4.212 representation_for_t_bar_cross_section_dimensions

The **representation_for_t_bar_cross_section_dimensions** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 't bar cross section dimensions'.

EXPRESS specification:

```
*)
RULE representation_for_t_bar_cross_section_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'width', 'web thk',
                                   'flange thk', 'radius', 'k'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   't bar cross section dimensions')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
```



```
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                      rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘t bar cross section dimensions’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'depth', 'width', 'web thk', 'flange thk', 'radius', or 'k'.

5.2.4.213 representation_for_triangular_cutout_design_parameters

The **representation_for_triangular_cutout_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'triangular cutout design parameters'.

EXPRESS specification:

```
*)
RULE representation_for_triangular_cutout_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['c1 radius', 'c2 radius', 'c3 radius',
                'c2 x', 'c3 x', 'c3 y'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to
     'triangular cutout design parameters' *)
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
+'.USED_REPRESENTATION'))
          | (temp_prop_def_rep.name = 'triangular cutout design parameters')
                  )
               ) > 0
            );
```



```
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'triangular cutout design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'c1 radius', 'c2 radius', 'c3 radius', 'c2 x', 'c3 x', or 'c3 y'.

5.2.4.214 representation_for_twist_location_design_parameters

The **representation_for_twist_location_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'twist location design parameters'.

EXPRESS specification:

```
*)
RULE
representation_for_twist_location_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['location', 'direction'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'twist location design parameters')
                 )
              ) > 0
           );
```



```
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'twist location design parameters', is to use a **representation** that has exactly one item in its set of **item**s that is a type of **representation_item** with attribute **name** equal to 'location', or 'direction'.

5.2.4.215 representation_has_global_unit_assigned_context

The **representation_has_global_unit_assigned_context** rule specifies that a **representation** has a **global_unit_assigned_context** if it has **representation_items** of type **value_representation_item** or **geometric_representation_item**.

EXPRESS specification:

```
*)
RULE representation_has_global_unit_assigned_context
FOR (representation);
  LOCAL
    has_guac :LOGICAL := TRUE;
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(representation) WHILE has_guac;
    REPEAT j := 1 TO SIZEOF(representation[i].ITEMS) WHILE has_guac;
      IF (('SHIP_STRUCTURES_SCHEMA.VALUE_representation_ITEM'
          IN TYPEOF(representation[i].ITEMS[j])) OR
          ('SHIP_STRUCTURES_SCHEMA.GEOMETRIC_representation_ITEM'
           IN TYPEOF(representation[i].ITEMS[j]))) THEN
        has_guac := ('SHIP_STRUCTURES_SCHEMA.GLOBAL_UNIT_ASSIGNED_CONTEXT'
         IN TYPEOF(representation[i].CONTEXT_OF_ITEMS));
      END_IF;
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: has_guac;
END_RULE;
(*
```


Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **representation** that contains either **value_representation_items** or **geometric_-representation_items** has a **representation_context** that is a **global_unit_assigned_context**.

5.2.4.216 representation_has_references_with_name_moment_3d

The **representation_has_references_with_name_moment_3d** rule specifies that an instance of **representation** with class id 'weight and centre of gravity' is referenced by exactly one instance of **property_definition_representation** via **used_representation** whose attribute **name** has the value 'moment 3d'.

EXPRESS specification:

```
*)
RULE  representation_has_references_with_name_moment_3d
FOR(representation, property_definition_representation,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF representation := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'weight and centre
  of gravity' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'weight and centre of gravity');
  (* get all instances of T1 that have class id 'weight and centre of
  gravity' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of representation in t1_set:
     get the property_definition_representation instances
    that are referencing a representation instance via used_representation,
     filter out those property_definition_representation instances
    whose attribute name has the value 'moment 3d'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.USED_REPRESENTATION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
'moment 3d')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

representation: the set of all instances of **representation** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **representation** that is referenced by an **applied_classification_assignment** with **name** equals 'weight and centre of gravity', is referenced by exactly one **property_definition_-representation.used_representation**, where **property_definition_representation.name** equals 'moment 3d'.

5.2.4.217 representation_item_corrugated_structure_boundary_-relationship

The **representation_item_corrugated_structure_boundary_relationship** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'corrugated structure boundary relationship design parameters'.

EXPRESS specification:

```
*)
RULE
representation_item_corrugated_structure_boundary_relationship
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['displacement'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
             'USED_REPRESENTATION'))
             | (temp_prop_def_rep.name =
             'corrugated structure boundary relationship design
  parameters')
                    )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
               rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
```

```
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'corrugated structure boundary relationship design parameters', the value of 'displacement' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.218 representation_item_for_panel_system_curve_boundary_design

The **representation_item_for_panel_system_curve_boundary_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'panel system curve boundary design parameters'.

EXPRESS specification:

```
*)
RULE
representation_item_for_panel_system_curve_boundary_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['displacement'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'panel system curve boundary design parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
```



```
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'panel system curve boundary design parameters', the value of 'displacement' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.219 representation_item_for_structural_weld_shrinkage_allowance

.
The **representation_item_for_structural_weld_shrinkage_allowance** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'structural weld shrinkage allowance feature design parameters'.

EXPRESS specification:

```
*)
RULE
representation_item_for_structural_weld_shrinkage_allowance
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['longitudinal allowance',
                                  'transverse allowance',
                                  'vertical allowance'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                    'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'structural weld shrinkage allowance feature '+
                    'design parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
```

```
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'structural weld shrinkage allowance feature design definition parameter representation', the value of 'longitudinal allowance', 'transverse allowance', 'vertical allowance' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.220 representation_item_for_transformation_to_parent

The **representation_item_for_transformation_to_parent** rule specifies that an instance of **property_definition** with class id 'local coordinate system' is referenced by exactly one instance of **representation_map** via **mapped_representation**.

EXPRESS specification:

```
*)
RULE
representation_item_for_transformation_to_parent
FOR(property_definition,
  representation, mapped_item, group,
  applied_classification_assignment);
  LOCAL
    c_a_set    : SET OF applied_classification_assignment := [];
    t1_set     : SET OF property_definition := [];
    t2_set     : SET OF property_definition_representation := [];
    t3_set     : SET OF representation := [];
    t4_set     : SET OF representation_map := [];
    t5_set     : SET OF mapped_item := [];
    arg_list   : LIST OF STRING :=
     ['local coordinate system position in global coordinate system',
     'local coordinate system position in parent local coordinate system',
     'local coordinate system position in parent local coordinate system
  with position reference'];
    violation1 : LOGICAL := FALSE;
    violation2 : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
             i.assigned_class.NAME =  'local coordinate system');
```



```
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation1;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation1 := NOT (SIZEOF(QUERY(t2_inst <* t2_set |
  t2_inst.used_representation.name =
      'local axis representation')) = 1);
    t3_set := t3_set + t2_set[i].used_representation;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation1;
    t4_set := bag_to_set(USEDIN(t3_set[i],
      'SHIP_STRUCTURES_SCHEMA.REPRESENTATION_MAP.MAPPED_REPRESENTATION'));
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation1;
    t5_set := bag_to_set(USEDIN(t4_set[i],
      'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM.MAPPING_SOURCE'));
      REPEAT j := 1 TO 3WHILE NOT violation2;
      violation2 := NOT (SIZEOF(QUERY(t2_inst <* t5_set | t2_inst.name =
      ARG_LIST[j])) = 1);
      END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation1;
    WR2: NOT violation2;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities
property_definition: the set of all instances of **property_definition** entities.
representation: the set of all instances of **representation** entities.
mapped_item: the set of all instances of **mapped_item** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **property_definition** that is referenced by an **applied_classification_assignment** with **name** equals 'local coordinate system', is referenced by exactly one **property_definition_-representation.definition**, where **property_definition_representation.name** equals 'local coordinate system', and a **used_representation** attribute which is a **representation** with attribute **name** equals 'local axis representation'.

WR2: Every **mapped_item** with a **name** of 'local coordinate system position in global coordinate system', 'local coordinate system position in parent local coordinate system', or 'local coordinate system position in parent local coordinate system with position reference' has an attribute **mapping_source** that references a **representation_map** that has a **mapped_representation** attribute that references a **representation** with attribute **name** equal to 'local axis representation'.

5.2.4.221 representation_item_to_structural_system_adjacency_-relationship

The **representation_item_to_structural_system_adjacency_relationship** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'structural system adjacency relationship design parameters'.

EXPRESS specification:

```
*)
RULE
representation_item_to_structural_system_adjacency_relationship
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['displacement'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                 'USED_REPRESENTATION'))
                 | (temp_prop_def_rep.name =
                 'structural system adjacency relationship design
  parameters')
                   )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'structural system adjacency relationship design parameters', the value of 'displacement' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.222 representation_items_for_corrugated_part_design_parameters

The **representation_items_for_corrugated_part_design_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'corrugated part design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_for_corrugated_part_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['mass'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'corrugated part design parameters')
                  )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'corrugated part design parameters', the value of 'mass' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.223 representation_items_for_corrugated_structure_design

The **representation_items_for_corrugated_structure_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'corrugated structure design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_for_corrugated_structure_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['corrugation width',
                                  'user defined tightness'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'corrugated structure design parameters')
                 )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```


Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'corrugated structure design parameters', the value of 'corrugation width',or 'user defined tightness' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.224 representation_items_for_corrugated_structure_function

The **representation_items_for_corrugated_structure_function** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'corrugated structure function parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_for_corrugated_structure_function
FOR (representation, representation_item,
  property_definition_representation);

  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['user def function'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'corrugated structure function parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
```



```
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'corrugated structure function parameters', the value of 'user def function' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.225 representation_items_for_edge_cutout_function_parameters

The **representation_items_for_edge_cutout_function_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'edge cutout function parameters'.

EXPRESS specification:

```
*)
RULE representation_items_for_edge_cutout_function_parameters
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['user def function'];
    found    : LOGICAL := FALSE;
  END_LOCAL;

  (* Find all instances of representation  which are used
     by a property_definition_representation with name equal to
    'edge cutout function parameters'   *)
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'+
    '.USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name = 'edge cutout function
  parameters')
                )
             ) > 0
          );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
```



```
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'edge cutout function parameters', the value of set 'user def function' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.226 representation_items_for_edge_feature_function_parameters

The **representation_items_for_edge_feature_function_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'edge feature function parameters'.

EXPRESS specification:

```
*)
RULE representation_items_for_edge_feature_function_parameters
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['user def function'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  (* Find all instances of representation  which are used
     by a property_definition_representation with name equal to
   'edge feature function parameters'
   *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
    +'.USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'edge feature function
  parameters')
                  )
              ) > 0
          );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
```

```
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'edge feature function parameters', the value of set 'user def function' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.227 representation_items_for_elongated_oval_cutout_design

The **representation_items_for_elongated_oval_cutout_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'elongated oval cutout design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_for_elongated_oval_cutout_design
FOR (representation);
   (*
   R12<'elongated oval cutout design parameters', ['r3', 'r3 x', 'r3 y']>
   *)
   LOCAL
     reps     : BAG OF REPRESENTATION := [];
     arg_list : LIST OF STRING := ['r3', 'r3 x', 'r3 y'];
     found    : LOGICAL := FALSE;
   END_LOCAL;
   (* Find all instances of representation  which are used
      by a property_definition_representation with name equal to
      'elongated oval cutout design parameters'   *)
   reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
      +'.USED_REPRESENTATION'))
       | (temp_prop_def_rep.name = 'elongated oval cutout design
   parameters')
                     )
               ) > 0
            );
```



```
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                 rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'elongated oval cutout design parameters', the value of set 'r3', 'r3 x', or 'r3 y' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.228 representation_items_for_interior_cutout_function_parameters

The **representation_items_for_interior_cutout_function_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the representation is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'interior cutout function parameters'.

EXPRESS specification:

```
*)
RULE representation_items_for_interior_cutout_function_parameters
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['user def function'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  (* Find all instances of representation  which are used
     by a property_definition_representation with name equal
     to 'interior cutout function parameters'   *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
     +'.USED_REPRESENTATION'))
         | (temp_prop_def_rep.name = 'interior cutout function parameters')
                 )
              ) > 0
           );
```



```
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                 rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'interior cutout function parameters', the value of set 'user def function' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.229 representation_items_for_inward_round_corner_design

The **representation_items_for_inward_round_corner_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'inward round corner design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_for_inward_round_corner_design
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['z depth'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  (* Find all instances of representation  which are used
     by a property_definition_representation with name equal
   to 'inward round corner design parameters'
   *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
     +'.USED_REPRESENTATION'))
        | (temp_prop_def_rep.name = 'inward round corner design
  parameters')
                  )
```



```
            ) > 0
          );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'inward round corner design parameters', the value of set 'z depth' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.230 representation_items_for_maximum_permissible_swbm_values

The **representation_items_for_maximum_permissible_swbm_values** rule specifies the **item_-element** attribute of the **compound_representation_item** of which inherited **name** attribute from **representation_item** has the value of 'maximum permissible swbm values', to have a **representation_item** of which **name** attribute has the value given in the list.

EXPRESS specification:

```
*)
RULE representation_items_for_maximum_permissible_swbm_values
FOR (compound_representation_item);
   LOCAL
    arg_list  : LIST OF STRING := ['hogging', 'sagging','position'];
    creps     : BAG OF compound_representation_item := [];
    rep_item  : SET OF representation_item  := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of compound_representation_item
      whose inherited .name equal to 'maximum permissible swbm values' *)
  creps := QUERY( temp_comp_rep_item <* compound_representation_item |
                    temp_comp_rep_item.name = 'maximum permissible swbm
  values');
  (* iterate over all compound_representation_item found above; stop, if
  one of
     them has not exactly one rep_item with for each name of the arg_list
  *)
  REPEAT i:=1 TO HIINDEX(creps) WHILE(NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE(NOT violation);
    REPEAT k:=1 TO HIINDEX(creps[i].item_element);
```

```
        rep_item := creps[i].item_element;
        violation:= (SIZEOF(QUERY(items <* rep_item |
                    items.name = arg_list[j])) =1);
      END_REPEAT;
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

compound_representation_item: the set of all instances of **compound_representation_item** entities.

Formal propositions:

WR1: Every **compound_representation_item** with **name** equal to 'maximum permissible swbm values', is to have an **item_element** that is a type of **representation_item** with attribute **name** equal to 'hogging', 'sagging', or 'position'.

5.2.4.231 representation_items_for_maximum_permissible_swsf_values

The **representation_items_for_maximum_permissible_swsf_values** rule specifies the **item_-element** attribute of the **compound_representation_item** of which inherited **name** attribute from **representation_item** has the value of 'maximum permissible swsf values', to have a **representation_item** of which **name** attribute has the value given in the list.

EXPRESS specification:

```
*)
RULE representation_items_for_maximum_permissible_swsf_values
FOR (compound_representation_item);
  LOCAL
    arg_list  : LIST OF STRING := ['negative value', 'positive value',
  'position'];
    creps     : BAG OF compound_representation_item := [];
    rep_item  : SET OF representation_item  := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of compound_representation_item whose inherited
  .name equal to
  'maximum permissible swsf values' *)
  creps := QUERY( temp_comp_rep_item <* compound_representation_item |
                  temp_comp_rep_item.name = 'maximum permissible swsf
  values');
  (* iterate over all compound_representation_item found above; stop, if
  one of
     them has not exactly one rep_item with for each name of the arg_list
  *)
  REPEAT i:=1 TO HIINDEX(creps) WHILE(NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE(NOT violation);
    REPEAT k:=1 TO HIINDEX(creps[i].item_element);
        rep_item := creps[i].item_element;
        violation:= (SIZEOF(QUERY(items <* rep_item |
                    items.name = arg_list[j])) =1);
```



```
      END_REPEAT;
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

compound_representation_item: the set of all instances of **compound_representation_item** entities.

Formal propositions:

WR1: Every **compound_representation_item** with **name** equal to 'maximum permissible swsf values', is to have an **item_element** that is a type of **representation_item** with attribute **name** equal to 'negative value', 'positive value', or 'position'.

5.2.4.232 representation_items_for_outward_round_corner_design

The **representation_items_for_outward_round_corner_design** rule specifies the items attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'outward round corner design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_for_outward_round_corner_design
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['z depth'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  (* Find all instances of representation  which are used
     by a property_definition_representation with name equal to
     'outward round corner design parameters'
   *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
    +'.USED_REPRESENTATION'))
       | (temp_prop_def_rep.name = 'outward round corner design
  parameters')
                  )
              ) > 0
          );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
```

```
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'outward round corner design parameters', the value of set 'z depth' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.233 representation_items_for_panel_system_boundary_relationship

The **representation_items_for_panel_system_boundary_relationship** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'panel system boundary relationship design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_for_panel_system_boundary_relationship
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['displacement'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'panel system boundary relationship design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
```



```
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'panel system boundary relationship design parameters', the value of 'displacement' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.234 representation_items_for_panel_system_function_parameters

The **representation_items_for_panel_system_function_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'panel system function parameters'.

EXPRESS specification:

```
*)
RULE representation_items_for_panel_system_function_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['user def function'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'panel system function parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
```

```
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'panel system function parameters', the value of 'user def function' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.235 representation_items_for_profile_boundary_relationship_design

The **representation_items_for_profile_boundary_relationship_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'profile boundary relationship design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_for_profile_boundary_relationship_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['displacement'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'profile boundary relationship design parameters')
                 )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
```



```
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'profile boundary relationship design parameters', the value of 'displacement' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.236 representation_items_for_profile_curve_trace_line_design

The **representation_items_for_profile_curve_trace_line_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** that has a **name** attribute with the value 'profile curve trace line design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_for_profile_curve_trace_line_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['displacement'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'profile curve trace line design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
```

```
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'profile curve trace line design parameters', the value of 'displacement' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.237 representation_items_for_profile_trace_line_relationship_design

The **representation_items_for_profile_trace_line_relationship_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'profile trace line relationship design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_for_profile_trace_line_relationship_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['displacement', 'auxiliary line'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'profile trace line relationship design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
```



```
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'profile trace line relationship design parameters', the value of 'displacement', 'auxiliary line' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.238 representation_items_for_rectangular_cutback_corner_design

The **representation_items_for_rectangular_cutback_corner_design** rule specifies the items attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the representation is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'rectangular cutback corner design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_for_rectangular_cutback_corner_design
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['z depth'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  (* Find all instances of representation  which are used
     by a property_definition_representation with name equal
     to 'rectangular cutback corner design parameters'
   *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
   +'.USED_REPRESENTATION'))
  | (temp_prop_def_rep.name = 'rectangular cutback corner design
  parameters')
                  )
```



```
              ) > 0
            );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'rectangular cutback corner design parameters', the value of set 'z depth' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.239 representation_items_for_seam_curve_relationship_parameter

The **representation_items_for_seam_curve_relationship_parameter** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'seam curve relationship parameter'.

EXPRESS specification:

```
*)
RULE
representation_items_for_seam_curve_relationship_parameter
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['displacement'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'seam curve relationship parameter')
                 )
```



```
            ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'seam curve relationship parameter', the value of 'displacement' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.240 representation_items_for_shear_bevel_design_parameters

The **representation_items_for_shear_bevel_design_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'shear bevel design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_for_shear_bevel_design_parameters
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['z angle'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  (* Find all instances of representation  which are used
     by a property_definition_representation with name equal to
    'shear bevel design parameters'
   *)
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
+'.USED_REPRESENTATION'))
```

```
                | (temp_prop_def_rep.name = 'shear bevel design parameters')
                  )
              ) > 0
          );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'shear bevel design parameters', the value of set 'z angle' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.241 representation_items_optional_for_bevel_design_parameters

The **representation_items_optional_for_bevel_design_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'bevel design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_bevel_design_parameters
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['z angle'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  (* Find all instances of representation  which are used by a
  property_definition_representation with name equal to 'bevel design
  parameters'   *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
      +'.USED_REPRESENTATION'))
```



```
                    | (temp_prop_def_rep.name = 'bevel design parameters')
                  )
              ) > 0
          );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'bevel design parameters', the value of set ['z angle'] is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.242 representation_items_optional_for_beveled_groove_weld_design

The **representation_items_optional_for_beveled_groove_weld_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute has a value 'beveled groove weld design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_optional_for_beveled_groove_weld_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['weld backing type', 'taper angle'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,

  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                   'USED_REPRESENTATION'))
```

```
                    | (temp_prop_def_rep.name =
                    'beveled groove weld design parameters')
                    )
                ) > 0
            );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'beveled groove weld design parameters', the value 'weld backing type', or 'taper angle' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.243 representation_items_optional_for_butt_groove_weld_design

The **representation_items_optional_for_butt_groove_weld_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'butt groove weld design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_optional_for_butt_groove_weld_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['weld backing type'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
```



```
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'butt groove weld design parameters')
                  )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'butt groove weld design parameters', the value of 'weld backing type' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.244 representation_items_optional_for_class_notation

The **representation_items_optional_for_class_notation** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** that has a **name** attribute with the value 'class notation'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_class_notation
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['ice class notation','service factor',
                                  'approval required for heavy cargo'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
```

```
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'class notation')
                  )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'class notation', the values of 'ice class notation', 'service factor' or 'approval required for heavy cargo' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.245 representation_items_optional_for_corrugation_dimensions

The **representation_items_optional_for_corrugation_dimensions** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** that has a **name** attribute with the value 'corrugation dimensions'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_corrugation_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['radius 1', 'radius 2'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
```



```
                    | (temp_prop_def_rep.name =
                    'corrugation dimensions')
                  )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'corrugation dimensions', the value of 'radius 1',or 'radius 2' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.246 representation_items_optional_for_groove_weld_design

The **representation_items_optional_for_groove_weld_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** that has a **name** attribute with the value 'groove weld design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_groove_weld_design
FOR (representation, representation_item,
  property_definition_representation);
  (*
  R12<'groove weld design parameters',['weld backing type']> *)
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['weld backing type'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
```

```
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'groove weld design parameters')
                  )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'groove weld design parameters', the value of 'weld backing type' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.247 representation_items_optional_for_owner_designation

The **representation_items_optional_for_owner_designation** rule specifies the **items** attribute of a **representation** is to have for each entry in the list, zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'owner designation'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_owner_designation
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['owner approval'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  (* Find all instances of representation  which are used
     by a property_definition_representation with name equal to 'owner
  designation'
   *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
```



```
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
        'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
        'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name = 'owner designation')
                 )
             ) > 0
          );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'owner designation', the value of set ['owner approval'] is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.248 representation_items_optional_for_panel_system_design

The **representation_items_optional_for_panel_system_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'panel system design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_optional_for_panel_system_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['stiffener spacing',
                                  'user defined tightness'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
```

```
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'panel system design parameters')
                  )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'panel system design parameters', the value of 'stiffener spacing', or 'user defined tightness' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.249 representation_items_optional_for_plate_design_parameters

The **representation_items_optional_for_plate_design_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'plate design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_plate_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['mass'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
```



```
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'plate design parameters')
                  )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'plate design parameters', the value of 'mass' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.250 representation_items_optional_for_plate_function_parameters

The **representation_items_optional_for_plate_function_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'plate function parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_optional_for_plate_function_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['user def function'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
```

```
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'plate function parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'plate function parameters', the value of 'user def function' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.251 representation_items_optional_for_plate_strake_function

The **representation_items_optional_for_plate_strake_function** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'plate strake function parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_optional_for_plate_strake_function
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['user def function'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
```



```
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'plate strake function parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'plate strake function parameters', the value of 'user def function' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.252 representation_items_optional_for_principal_characteristics

The **representation_items_optional_for_principal_characteristics** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'principal characteristics'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_principal_characteristics
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['block coefficient'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
```

```
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'principal characteristics')
                 )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'principal characteristics', the value of 'block coefficient' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.253 representation_items_optional_for_profile_design_parameters

The **representation_items_optional_for_profile_design_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'profile design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_profile_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['mass'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
```



```
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'profile design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'profile design parameters', the value of 'mass' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.254 representation_items_optional_for_profile_function_parameters

The **representation_items_optional_for_profile_function_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'profile function parameters'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_profile_function_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['user def function'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
```

```
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'profile function parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'profile function parameters', the value of 'user def function' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.255 representation_items_optional_for_seam_design_parameters

The **representation_items_optional_for_seam_design_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'seam design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_seam_design_parameters
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['seam curve'];
    found    : LOGICAL  := FALSE;
  END_LOCAL;
```



```
  (* Find all instances of representation  which are used
     by a property_definition_representation with name equal to 'seam
  design parameters'
   *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'+
  '.USED_REPRESENTATION'))
     | (temp_prop_def_rep.name = 'seam design parameters')
                    )
              ) > 0
           );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'seam design parameters', the value of set 'seam curve' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.256 representation_items_optional_for_structural_system_design

The **representation_items_optional_for_structural_system_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'structural system design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_optional_for_structural_system_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
```

```
    arg_list : LIST OF STRING := ['user def  tightness'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'structural system design parameters')
                  )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equals 'structural system design parameters', the value of 'user def tightness' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.257 representation_items_optional_for_structural_system_function

The **representation_items_optional_for_structural_system_function** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a property_definition_representation which has a **name** attribute with the value 'structural system function parameters'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_structural_system_function
FOR (representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['user def function'];
```



```
    found    : LOGICAL := FALSE;
  END_LOCAL;
  (* Find all instances of representation  which are used
     by a property_definition_representation with name
     equal to 'structural system function parameters'
   *)
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
   'USED_REPRESENTATION'))
       | (temp_prop_def_rep.name = 'structural system function parameters')
                   )
             ) > 0
          );
  (* iterate over all representations found above.  Stop, if for one of
     them the names of its representation_items are duplicated.
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'structural system function parameters', the value of set ['user def function'] is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.258 representation_items_optional_for_weld_design_parameters

The **representation_items_optional_for_weld_design_parameters** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'weld design parameters'.

EXPRESS specification:

```
*)
RULE representation_items_optional_for_weld_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
```

```
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['start suspension',
                                  'end suspension',
                                  'weld cross section area'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'weld design parameters')
                 )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
                rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'weld design parameters', the value of 'start suspension', 'end suspension', 'weld cross section area' is not to occur more than once as values of the **representation_item** attribute **name**.

5.2.4.259 representation_items_optional_for_weld_filler_material_design

The **representation_items_optional_for_weld_filler_material_design** rule specifies the **items** attribute of a **representation** is to have for each entry in the list zero or one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'weld filler material design parameters'.

EXPRESS specification:

```
*)
RULE
representation_items_optional_for_weld_filler_material_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['chemical composition id',
                                  'electrode id', 'material strength id',
                                  'notch impact work id'];
    found    : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'weld filler material design parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT found);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT found);
      found := (SIZEOF(QUERY(rep_item <* reps[i].items |
               rep_item.name=arg_list[j])) > 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT found;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: For all **representation_item** referenced by the **items** attribute of every **representation** referenced by the **definition** attribute of a **property_definition_representation** with **name** equal to 'weld filler material design parameters', the value of 'chemical composition id', 'electrode id', 'material strength id', 'notch impact work id' is not to occur more than once as values of the **representation_-item** attribute **name**.

5.2.4.260 representation_items_specified_for_design_swsf_values

The **representation_items_specified_for_design_swsf_values** rule specifies the **item_element** attribute of the **compound_representation_item** of which inherited **name** attribute from **representation_item** has the value of 'design swsf values', to have a **representation_item** of which **name** attribute has the value given in the list.

EXPRESS specification:

```
*)
RULE representation_items_specified_for_design_swsf_values
FOR (compound_representation_item,
     representation_item);
  LOCAL
    arg_list  : LIST OF STRING := ['negative value', 'positive value',
                                   'position'];
    creps     : BAG OF compound_representation_item := [];
    rep_item  : SET OF representation_item  := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of compound_representation_item
      whose inherited .name equal to 'design swsf values' *)
  creps := QUERY( temp_comp_rep_item <* compound_representation_item |
                     temp_comp_rep_item.name = 'design swsf values');

  (* iterate over all compound_representation_item found above; stop, if
  one of
     them has not exactly one rep_item with for each name of the arg_list
  *)
  REPEAT i:=1 TO HIINDEX(creps) WHILE(NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE(NOT violation);
    REPEAT k:=1 TO HIINDEX(creps[i].item_element);
         rep_item := creps[i].item_element;
         violation:= (SIZEOF(QUERY(items <* rep_item |
                      items.name = arg_list[j])) =1);
      END_REPEAT;
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

compound_representation_item: the set of all instances of **compound_representation_item** entities.

Formal propositions:

WR1: Every **compound_representation_item** with **name** 'design swsf values', is to have an **item_element** that is a type of **representation_item** with attribute **name** equal to 'negative value', 'positive value', or 'position'.

5.2.4.261 representation_local_co_ordinate_system_with_position_-reference

The **representation_local_co_ordinate_system_with_position_reference** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_-item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'local coordinate system with position reference'.

EXPRESS specification:

```
*)
RULE
representation_local_coordinate_system_with_position_reference
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps     : BAG OF REPRESENTATION := [];
    arg_list : LIST OF STRING := ['local axes and origin'];
    violation: LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                          'local coordinate system with position
  reference')
                 )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'local coordinate system with position reference', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'local axes and origin'.

5.2.4.262 representation_maximum_permissible_still_water_bending_-moment

The **representation_maximum_permissible_still_water_bending_moment** rule specifies the **items** attribute of a **representation**, that has a **name** attribute is 'maximum permissible swbm values', to have exactly one **representation_item** that has a **name** attribute has the value given in 'maximum permissible swbm values'.

EXPRESS specification:

```
*)
RULE
representation_maximum_permissible_still_water_bending_moment
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  whose name equal
     to 'maximum permissible still water bending moment' *)
  reps := QUERY(
                temp_rep <* representation | temp_rep.name =
                'maximum permissible still water bending moment'
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of
     the 'maximum permissible swbm values' *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = 'maximum permissible swbm values')) <>
  1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **representation** with **name** equal to 'maximum permissible still water bending moment' is to have exactly one item in its set of **items** that is a type of **representation_item** with **name** equal to 'maximum permissible swbm values'.

5.2.4.263 representation_of_design_still_water_bending_moment_-parameters

The **representation_of_design_still_water_bending_moment_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_-item** that has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** that has a **name** attribute with the value 'design still water bending moment parameters'.

EXPRESS specification:

```
*)
RULE
representation_of_design_still_water_bending_moment_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['hogging amidship', 'sagging amidship'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
     to 'design still water bending moment parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'
  +'.USED_REPRESENTATION'))
  | (temp_prop_def_rep.name = 'design still water bending moment
  parameters')
                       )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                      rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘design still water bending moment parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'hogging amidship' or 'sagging amidship'.

5.2.4.264 representation_of_maximum_permissible_still_water_shear_-force

The **representation_of_maximum_permissible_still_water_shear_force** rule specifies the **items** attribute of a **representation**, whose **name** attribute is 'maximum permissible swsf values', to have exactly one **representation_item** whose **name** attribute has the value given in 'maximum permissible swsf values'.

EXPRESS specification:

```
*)
RULE
representation_of_maximum_permissible_still_water_shear_force
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  whose name
     equal to 'maximum permissible still water shear force' *)
  reps := QUERY(
                temp_rep <* representation | temp_rep.name =
                'maximum permissible still water shear force'
          );

  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of
     the 'maximum permissible swsf values' *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name =
                    'maximum permissible swsf values')) <> 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **representation** with **name** equal to 'maximum permissible still water shear force' is to have exactly one item in its set of **items** that is a type of **representation_item** with **name** equal to 'maximum permissible swsf values'.

5.2.4.265 representation_restricted_by_name_bevel_design_parameters

The **representation_restricted_by_name_bevel_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'bevel design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_bevel_design_parameters
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['offset', 'x y angle'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to 'bevel
  design parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
            'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION'+
           '.USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name = 'bevel design parameters')
                  )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of them has
  not exactly one rep_item with for each name of the arg_list  *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘bevel design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_-item** with attribute **name** equal to 'offset' or 'x y angle'.

5.2.4.266 representation_restricted_by_name_class_notation

The **representation_restricted_by_name_class_notation rule** specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the representation is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'class notation'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_class_notation
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['service area',
                            'approval required for oil cargo',
                            'approval required for loading unloading
  aground',
                            'approval required for unloading grabs'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'class notation')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'class notation', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'service area', 'approval required for oil cargo', 'approval required for loading unloading aground', or 'approval required for unloading grabs'.

5.2.4.267 representation_restricted_by_name_class_parameters

The **representation_restricted_by_name_class_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'class parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_class_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['length class', 'length solas',
                                   'scantlings draught', 'block coefficient
                                      class',
                                'design speed ahead','design speed astern'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'class parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'class parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'length class', 'length solas', 'scantlings draught', 'block coefficient class', 'design speed ahead', or 'design speed astern'.

5.2.4.268 representation_restricted_by_name_corrugation_dimensions

The **representation_restricted_by_name_corrugation_dimensions** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'corrugation dimensions'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_corrugation_dimensions
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'flat width 1', 'flat width 2',
                                   'slope width'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'corrugation dimensions')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘corrugation dimensions’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_-item** with attribute **name** equal to 'depth', 'flat width 1', 'flat width 2', or 'slope width'.

5.2.4.269 representation_restricted_by_name_flare_area_parameters

The **representation_restricted_by_name_flare_area_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a name attribute with the value 'flare area parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_flare_area_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['area', 'height'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name = 'flare area parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘flare area parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_-item** with attribute **name** equal to 'area', or 'height'.

5.2.4.270 representation_restricted_by_name_freeboard_characteristics

The **representation_restricted_by_name_freeboard_characteristics** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list, exactly one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'freeboard characteristics'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_freeboard_characteristics
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['assigned code','freeboard'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to 'freeboard
  characteristics' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                     'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'freeboard
  characteristics')
                  )
              ) > 0
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'freeboard characteristics', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_-item** with attribute **name** equal to 'assigned code', or 'freeboard'.

5.2.4.271 representation_restricted_by_name_lightship_definition

The **representation_restricted_by_name_lightship_definition** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list, exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'lightship_definition'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_lightship_definition
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['lightship weight',
                                   'lightship centre of gravity'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name
     equal to 'lightship_definition' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
        'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
        'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name = 'lightship_definition')
                  )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'lightship_definition', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'lightship weight', or 'lightship centre of gravity'.

5.2.4.272 representation_restricted_by_name_lightship_weight_item

The **representation_restricted_by_name_lightship_weight_item** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list, exactly one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'lightship weight item'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_lightship_weight_item
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['aft weight extent','fwd weight
   extent'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
     to 'lightship weight item' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
      'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name = 'lightship weight item')
                  )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘lightship weight item’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_-item** with attribute **name** equals 'aft weight extent' or 'fwd weight extent'.

5.2.4.273 representation_restricted_by_name_loadline

The **representation_restricted_by_name_loadline** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list, exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'load line'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_loadline
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['load line length',
                                   'load line depth',
                                   'load line displacement',
                                   'load line block coefficient',
                                   'load line regulation'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal to 'loadline'
  *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
     'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name = 'loadline')
                  )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: every **property_definition_representation** with **name** equal to ‘load line’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'load line length','load line depth','load line displacement','load line block coefficient', or 'load line regulation'.

5.2.4.274 representation_restricted_by_name_plate_design_parameters

The **representation_restricted_by_name_plate_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'plate design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_plate_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['material offset', 'thickness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                     'plate design parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'plate design parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'material offset' or 'thickness'.

5.2.4.275 representation_restricted_by_name_plate_function_parameters

The **representation_restricted_by_name_plate_function_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'plate function parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_plate_function_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['function'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'plate function parameters')
                 )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘plate function parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'function'.

5.2.4.276 representation_restricted_by_name_principal_characteristics

The **representation_restricted_by_name_principal_characteristics** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'principal characteristics'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_principal_characteristics
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['length between perpendiculars',
                                   'moulded breadth', 'moulded depth',
                                   'design draught', 'design deadweight',
                                   'min draught at fp','max draught at fp',
                                   'min draught at ap','max draught at ap'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                     'principal characteristics')
                  )
               ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'principal characteristics', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_-item** with attribute **name** equal to 'length between perpendiculars', 'moulded breadth', 'moulded depth', 'design draught', 'design deadweight','min draught at fp','max draught at fp', 'min draught at ap', or 'max draught at ap'.

5.2.4.277 representation_restricted_by_name_profile_design_parameters

The **representation_restricted_by_name_profile_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'profile design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_profile_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['mirrored'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'profile design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'profile design parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'mirrored'.

5.2.4.278 representation_restricted_by_name_section_properties

The **representation_restricted_by_name_section_properties** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'section properties'.

EXPRESS specification:

```
*)
RULE representation_restricted_by_name_section_properties
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['nominal mass per length', 'area',
                                   'na u', 'na v', 'moi u', 'moi v',
                                   'moi uv', 'tr'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                     'section properties')
                  )
              ) > 0
           );

  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘section properties’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'nominal mass per length', 'area', 'na u', 'na v', 'moi u', 'moi v', 'moi uv', or 'tr'.

5.2.4.279 representation_restricted_by_name_table_of_shear_force_-values

The **representation_restricted_by_name_table_of_shear_force_values** rule specifies the **items** attribute of a **representation**, that has a **name** attribute equal to 'design swsf values', to have exactly one **representation_item** that has a **name** attribute with the value given in 'design swsf values'.

EXPRESS specification:

```
*)
RULE
representation_restricted_by_name_table_of_shear_force_values
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  whose name
  equal to 'table of shear force values' *)
  reps := QUERY(
   temp_rep <* representation | temp_rep.name = 'table of shear force
  values'
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name
     of the 'design swsf values' *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = 'design swsf values')) <> 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **representation** with **name** equal to 'table of shear force values' is to have exactly one item in its set of **items** that is a type of **representation_item** with **name** equal to 'design swsf values'.

5.2.4.280 representation_restricted_by_name_w_shape_cross_section

The **representation_restricted_by_name_w_shape_cross_section** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'w shape cross section'.

EXPRESS specification:

```
*)
RULE
representation_restricted_by_name_w_shape_cross_section
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['depth', 'width', 'web thk',
                                   'flange thk', 'radius', 'k'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'w shape cross section')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to‘w shape cross section’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'depth', 'width', 'web thk', 'flange thk', 'radius', or 'k'.

5.2.4.281 representation_restricted_for_assembly_manufacturing_position

The **representation_restricted_for_assembly_manufacturing_position** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_-item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'assembly manufacturing position parameters'.

EXPRESS specification:

```
*)
RULE
 representation_restricted_for_assembly_manufacturing_position
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['assembly footprint','orientation'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
     | (temp_prop_def_rep.name = 'assembly manufacturing position
  parameters')
                    )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'assembly manufacturing position parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'assembly footprint' or 'orientation'.

5.2.4.282 representation_restricted_weight_and_centre_of_gravity

The **representation_restricted_weight_and_centre_of_gravity** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list, exactly one **representation_item** that has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** that has a **name** attribute with the value 'weight and centre of gravity'.

EXPRESS specification:

```
*)
RULE representation_restricted_weight_and_centre_of_gravity
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['mass','centre of gravity'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name
     equal to 'weight and centre of gravity' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
          'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
         'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name = 'weight and centre of
  gravity')
                  )
              ) > 0
           );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'weight and centre of gravity', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'mass' or 'centre of gravity'.

5.2.4.283 representation_restricted_for_beveled_groove_weld_design

The **representation_restricted_for_beveled_groove_weld_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'beveled groove weld design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_beveled_groove_weld_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING :=
                ['endcut shape', 'taper', 'sideness', 'weld joint spacer',
           'configuration', 'penetration', 'joint_orientation',
  'tightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
               'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                 'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                                  'beveled groove weld design parameters')
                  )
               ) > 0
            );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'beveled groove weld design parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'endcut shape'; 'taper', 'sideness', 'weld joint spacer', 'configuration', 'penetration', 'joint_orientation', or 'tightness'.

5.2.4.284 representation_restricted_for_butt_groove_weld_design

The **representation_restricted_for_butt_groove_weld_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'butt groove weld design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_butt_groove_weld_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING :=
              ['face shape', 'sideness', 'weld joint spacer',
  'configuration',
              'penetration', 'joint_orientation', 'tightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                                   'butt groove weld design parameters')
                  )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'butt groove weld design parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'face shape', 'sideness', 'weld joint spacer', 'configuration', 'penetration', 'joint_orientation', or 'tightness'.

5.2.4.285 representation_restricted_for_composite_feature_design

The **representation_restricted_for_composite_feature_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'composite feature design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_composite_feature_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['composed of', 'description', 'name'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                                    'composite feature design
  parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'composite feature design parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'composed of', 'description', or 'name'.

5.2.4.286 representation_restricted_for_continuous_fillet_weld_design

The **representation_restricted_for_continuous_fillet_weld_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'continuous fillet weld design parameters'.

EXPRESS specification:

```
*)
RULE
 representation_restricted_for_continuous_fillet_weld_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['endcut shape type', 'sideness',
           'configuration', 'penetration', 'joint_orientation',
  'tightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                                'continuous fillet weld design
  parameters')
                  )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'continuous fillet weld design parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'endcut shape type', 'sideness', 'configuration', 'penetration', 'joint_orientation', or 'tightness'.

5.2.4.287 representation_restricted_for_corrugated_structure_design

The **representation_restricted_for_corrugated_structure_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'corrugated structure design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_corrugated_structure_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['thickness', 'tightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
              'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                                  'corrugated structure design
  parameters')
                 )
              ) > 0
         );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
```

```
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘corrugated structure design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'thickness', or 'tightness'.

5.2.4.288 representation_restricted_for_corrugated_structure_function

The **representation_restricted_for_corrugated_structure_function** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'corrugated structure function parameters'.

EXPRESS specification:

```
*)
RULE
 representation_restricted_for_corrugated_structure_function
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['function'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                            'corrugated structure function parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
```



```
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘corrugated structure function parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'function'.

5.2.4.289 representation_restricted_for_fillet_weld_design_parameters

The **representation_restricted_for_fillet_weld_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'fillet weld design parameters'.

EXPRESS specification:

```
*)
RULE
representation_restricted_for_fillet_weld_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['endcut shape type', 'sideness',
          'configuration', 'penetration', 'joint_orientation',
  'tightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                                  'fillet weld design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
```

```
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'fillet weld design parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'endcut shape type', 'sideness', 'configuration', 'penetration', 'joint_orientation', or 'tightness'.

5.2.4.290 representation_restricted_for_groove_weld_design

The **representation_restricted_for_groove_weld_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'groove weld design parameters'.

EXPRESS specification:

```
*)
RULE
representation_restricted_for_groove_weld_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['sideness', 'weld joint spacer',
          'configuration', 'penetration', 'joint_orientation',
  'tightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                                          'groove weld design
  parameters')
                )
             ) > 0
```



```
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘groove weld design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'sideness', 'weld joint spacer', 'configuration', 'penetration', 'joint_orientation', or 'tightness'.

5.2.4.291 representation_for_homogeneous_ship_material_property

The **representation_for_homogeneous_ship_material_property** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'homogeneous ship material property parameters'.

EXPRESS specification:

```
*)
RULE
representation_for_homogeneous_ship_material_property
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['density', 'poisson ratio',
                                   'stress of fracture',
                                   'thermal expansion coefficient',
                                   'yield point', 'youngs module'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
               'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                 'USED_REPRESENTATION'))
```

```
                    | (temp_prop_def_rep.name =
                            'homogeneous ship material property
  parameters')
                  )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘homogeneous ship material property parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'density', 'poisson ratio', 'stress of fracture', 'thermal expansion coefficient', 'yield point', or 'youngs module'.

5.2.4.292 representation_restricted_for_hull_cross_section_design_-definition

The **representation_restricted_for_hull_cross_section_design_definition** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_-item** whose **name** attribute has the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** whose **name** attribute has a value 'hull cross section design definition parameters'.

EXPRESS specification:

```
*)
RULE
representation_restricted_for_hull_cross_section_design_definition
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['mirrored symmetry'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
```



```
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
                'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
                  'USED_REPRESENTATION'))
                      | (temp_prop_def_rep.name =
                            'hull cross section design definition
  parameters')
                    )
                ) > 0
            );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'hull cross section design definition parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'mirrored symmetry'.

5.2.4.293 representation_restricted_for_intermittent_fillet_weld_design

The **representation_restricted_for_intermittent_fillet_weld_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'intermittent fillet weld design parameters'.

EXPRESS specification:

```
*)
RULE
representation_restricted_for_intermittent_fillet_weld_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['end rules', 'cutout rules',
           'fillet alignment', 'fillet weld length', 'fillet weld spacing',
           'penetration rules', 'endcut shape type', 'sideness',
           'configuration', 'penetration', 'joint_orientation',
  'tightness'];
    violation : LOGICAL := FALSE;
```

```
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                             'intermittent fillet weld design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘intermittent fillet weld design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'end rules', 'cutout rules', 'fillet alignment', 'fillet weld length', 'fillet weld spacing', 'penetration rules', 'endcut shape type', 'sideness', 'configuration', 'penetration', 'joint_orientation', or 'tightness'.

5.2.4.294 representation_restricted_for_panel_system_curve_boundary

The **representation_restricted_for_panel_system_curve_boundary** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'panel system curve boundary design parameters'.

EXPRESS specification:

```
*)
RULE
representation_restricted_for_panel_system_curve_boundary
FOR (representation, representation_item,
property_definition_representation);

  LOCAL
```



```
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['curve'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
             'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
               'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                            'panel system curve boundary design
  parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘panel system curve boundary design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'curve'.

5.2.4.295 representation_restricted_for_panel_system_design_parameters

The **representation_restricted_for_panel_system_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'panel system design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_panel_system_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
```

```
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['material offset', 'thickness',
  'tightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'panel system design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘panel system design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'material offset', 'thickness', or 'tightness'.

5.2.4.296 representation_restricted_for_panel_system_function

The **representation_restricted_for_panel_system_function** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'panel system function parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_panel_system_function
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
```



```
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['function'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'panel system function parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘panel system function parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'function'.

5.2.4.297 representation_restricted_for_panel_system_plane_boundary

The **representation_restricted_for_panel_system_plane_boundary** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'panel system plane boundary design parameters'.

EXPRESS specification:

```
*)
RULE
representation_restricted_for_panel_system_plane_boundary
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
```

```
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['plane'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'panel system plane boundary design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘panel system plane boundary design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'plane'.

5.2.4.298 representation_restricted_for_plate_renewal_definition_-attributes

The **representation_restricted_for_plate_renewal_definition_attributes** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_-item** whose **name** attribute has the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** whose **name** attribute has a value 'plate renewal definition attributes'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_plate_renewal_definition_attributes
FOR (representation, representation_item,
  property_definition_representation);
```



```
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['rule thickness', 'renewal thickness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'plate renewal definition attributes')
                  )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘plate renewal definition attributes’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'rule thickness' or 'renewal thickness'.

5.2.4.299 representation_restricted_for_plate_strake_function

The **representation_restricted_for_plate_strake_function** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** which has a **name** attribute with the value 'plate strake function parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_plate_strake_function
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
```

```
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['function'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'plate strake function parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘plate strake function parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'function'.

5.2.4.300 representation_restricted_for_ship_material_property

The **representation_restricted_for_ship_material_property** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'ship material property parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_ship_material_property
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
```



```
    arg_list  : LIST OF STRING := ['density'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'ship material property parameters')
                  )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘ship material property parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'density'.

5.2.4.301 representation_restricted_for_spot_seam_weld_design

The **representation_restricted_for_spot_seam_weld_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'spot seam weld design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_spot_seam_weld_design
FOR (representation, representation_item,
  property_definition_representation);
  (*R10<'spot seam weld design parameters', ['weld
  context','configuration','penetration','joint_orientation','tightness']>
  *)
```

```
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['weld context', 'configuration',
                                   'penetration', 'joint_orientation',
                                   'tightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'spot seam weld design parameters')
                 )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘spot seam weld design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'weld context', 'configuration', 'penetration', 'joint_orientation', or 'tightness'.

5.2.4.302 representation_restricted_for_structural_part_joint_design

The **representation_restricted_for_structural_part_joint_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'structural part joint design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_structural_part_joint_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['tightness', 'joint orientation'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                     'structural part joint design parameters')
                   )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘structural part joint design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'tightness' or 'joint orientation'.

5.2.4.303 representation_restricted_for_structural_system_design

The **representation_restricted_for_structural_system_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute with the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute with the value 'structural system design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_structural_system_design
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['thightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'structural system design parameters')
                 )
              ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘structural system design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'thightness'.

5.2.4.304 representation_restricted_for_structural_system_function

The **representation_restricted_for_structural_system_function** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** which has a **name** attribute has the value given in the list if the **representation** is the **used_-representation** in a **property_definition_representation** which has a **name** attribute has a value 'structural system function parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_structural_system_function
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['function'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name
   equal to 'structural system function parameters' *)
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
       | (temp_prop_def_rep.name = 'structural system function parameters')
                   )
             ) > 0
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘structural system function parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'function'.

5.2.4.305 representation_restricted_for_weld_design_parameters

The **representation_restricted_for_weld_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'weld design parameters'.

EXPRESS specification:

```
*)
RULE
representation_restricted_for_weld_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['shape of weld surface',
                                   'weld geometry', 'connection angle',
                                   'rotation angle', 'inclination angle',
                                   'shrinkage', 'weld size'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                     'weld design parameters')
                   )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘weld design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'shape of weld surface', 'weld geometry', 'connection angle', 'rotation angle', 'inclination angle', 'shrinkage', or 'weld size'.

5.2.4.306 representation_restricted_for_weld_filler_material_design

The **representation_restricted_for_weld_filler_material_design** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'weld filler material design parameters'.

EXPRESS specification:

```
*)
RULE representation_restricted_for_weld_filler_material_design
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['density'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation  which are used
     by a property_definition_representation with name equal
  to 'weld filler material design parameters' *)
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                   'weld filler material design parameters')
                   )
              ) > 0
          );
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item with for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘weld filler material design parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'density'.

5.2.4.307 representation_restricted_for_weld_manufacturing_definition

The **representation_restricted_for_weld_manufacturing_definition** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'weld manufacturing definition parameters'.

EXPRESS specification:

```
*)
RULE
 representation_restricted_for_weld_manufacturing_definition
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['position','process',
                                   'degree of automations',
                                   'number of weld passes',
                                   'welding deposition sequences',
                                   'welding environment',
                                   'tack weld used'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
            temp_rep <* representation |
               SIZEOF (
                  QUERY(
                     temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                     | (temp_prop_def_rep.name =
                     'weld manufacturing definition parameters')
                  )
               ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘weld manufacturing definition parameters’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'position', 'process', 'degree of automations', 'number of weld passes', 'welding deposition sequences', 'welding environment', or 'tack weld used'.

5.2.4.308 representation_restricted_for_welded_joint_design_parameters

The **representation_restricted_for_welded_joint_design_parameters** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'welded joint design parameters'.

EXPRESS specification:

```
*)
RULE
representation_restricted_for_welded_joint_design_parameters
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['configuration', 'penetration',
                                   'joint_orientation', 'tightness'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
          temp_rep <* representation |
             SIZEOF (
                QUERY(
                   temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                   | (temp_prop_def_rep.name =
                   'welded joint design parameters')
                )
             ) > 0
          );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                   rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'welded joint design parameters', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'configuration', 'penetration', 'joint_orientation', or 'tightness'.

5.2.4.309 representation_restricted_for_welding_sequence_representation

The **representation_restricted_for_welding_sequence_representation** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'welding sequence representation'.

EXPRESS specification:

```
*)
RULE
representation_restricted_for_welding_sequence_representation
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['name','start offset',
                                   'end offset', 'direction'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'welding sequence representation')
                 )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
```



```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to 'welding sequence representation', is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'name', 'start offset', 'end offset', or 'direction'.

5.2.4.310 representation_restricts_profile_renewal_definition_attributes

The **representation_restricts_profile_renewal_definition_attributes** rule specifies that the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list if the **representation** is the **used_representation** in a **property_definition_representation** whose **name** attribute has a value 'profile renewal definition attributes'.

EXPRESS specification:

```
*)
RULE representation_restricts_profile_renewal_definition_attributes
FOR (representation, representation_item,
  property_definition_representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['renewal modulus',
                                   'rule inertia', 'rule section modulus'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  reps := QUERY(
           temp_rep <* representation |
              SIZEOF (
                 QUERY(
                    temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.'+
  'USED_REPRESENTATION'))
                    | (temp_prop_def_rep.name =
                    'profile renewal definition attributes')
                 )
              ) > 0
           );
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
```

```
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition_representation: the set of all instances of **property_definition_-representation** entities.
representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **property_definition_representation** with **name** equal to ‘profile renewal definition attributes’, is to use a **representation** that has exactly one item in its set of **items** that is a type of **representation_item** with attribute **name** equal to 'renewal modulus', 'rule inertia', or 'rule section modulus'.

5.2.4.311 representation_for_moment_3d_restricts_representation_item

The **representation_for_moment_3d_restricts_representation_item** rule specifies the **items** attribute of a **representation** with the **name** 'moment 3d' is to have for each entry in the list, exactly one **representation_item** that has a **name** attribute with the value given in the list.

EXPRESS specification:

```
*)
RULE representation_for_moment_3d_restricts_representation_item
FOR (representation);
  LOCAL
    reps      : BAG OF REPRESENTATION := [];
    arg_list  : LIST OF STRING := ['longitudinal moment',
                                   'transverse moment',
                                   'vertical moment','origin'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* find all instances of representation with name equal to 'moment 3d'
   *)
  reps := QUERY(i <* representation |
       i.NAME = 'moment 3d');

  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item for each name in the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(reps) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* reps[i].items |
                     rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```


Argument definitions:

representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every instance of **representation** that has an **applied_classification_assignment** whose attribute **assigned_class.name** equals 'moment 3d' is to collect each value out of ['longitudinal moment', 'transverse moment', 'vertical moment', 'origin'] exactly and only one instance of **representation_item** in its attribute **items** whose **name** attribute equals that value.

5.2.4.312 representation_with_versionable_object_matches_pattern

The **representation_with_versionable_object_matches_pattern** rule specifies a string type attribute **representation** of type 'versionable object' to match the pattern '*.*' if the instance of **representation** is of class 'versionable object'.

EXPRESS specification:

```
*)
RULE
representation_with_versionable_object_matches_pattern
FOR (group, applied_classification_assignment);
  LOCAL
    clfied_inst :SET OF GENERIC := [];
  END_LOCAL;

  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'versionable object') THEN
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
  END_IF;
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **representation** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'versionable object' is to match the pattern '*.*'.

5.2.4.313 restrict_name_for_known_source

The **restrict_name_for_known_source** rule ensures that each instance of **known_source** has a name of 'ISO 13584 library'.

EXPRESS specification:

```
*)
RULE restrict_name_for_known_source
FOR (known_source);
  WHERE
    WR1: SIZEOF ( QUERY ( ks <* known_source | ( ks.  name<>
    'ISO 13584 library' ) ) ) =0;
END_RULE;
(*
```

Argument definitions:

known_source: the set of all instances of **known_source** entities.

Formal propositions:

WR1: Each instance of **known_source** is to have a name of 'ISO 13584 library'.

5.2.4.314 revision_has_mandatory_attribute_description

The **revision_has_mandatory_attribute_description** rule specifies that for an instance of **group** with class id 'revision' the optional attribute **description** is instantiated.

EXPRESS specification:

```
*)
RULE
revision_has_mandatory_attribute_description
FOR (group);
  LOCAL
    t1_set  : SET OF group := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* group | VALUE_IN(WHICH_CLASS(i), 'revision'));
  violate := (SIZEOF(QUERY(k <* t1_set |
  NOT EXISTS (k.description))) > 0);
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.

Formal propositions:

WR1: The optional attribute **description** is to exist for every instance of **group** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'revision'.

5.2.4.315 revision_with_context_referenced_for_context_of_revision

The **revision_with_context_referenced_for_context_of_revision** rule specifies that each instance of type **group** with class 'revision with context' is referenced by exactly one assignment of type **applied_group_assignment** with role 'context of revision' via attribute **assigned_group**.

EXPRESS specification:

```
*)
RULE
revision_with_context_referenced_for_context_of_revision
FOR(applied_group_assignment, group);
  LOCAL
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* group |
         VALUE_IN(WHICH_CLASS(a), 'revision with context'));

  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_group_assignment |
               (b.assigned_group = t1_set[i]) AND
               (b.role.name = 'context of revision'));
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.

Formal propositions:

WR1: Every instance of **group** that has an **applied_classification_assignment** whose **assigned_-classification** is a **group** with attribute **name** equal to 'revision with context' is referenced by exactly one instance of type **applied_group_assignment** whose **role** equals 'context of revision' through attribute **assigned_group**.

5.2.4.316 seam_has_at_most_one_class_references

The **seam_has_at_most_one_class_references** rule specifies that an instance of **shape_aspect** with class id 'seam' is referenced by at most one instance of **shape_aspect_relationship** with class id 'seam curve relationship' via **relating_shape_aspect.**

EXPRESS specification:

```
*)
RULE
seam_has_at_most_one_class_references
FOR(shape_aspect,
  shape_aspect_relationship, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET Of shape_aspect := [];
    t2_set    : SET Of shape_aspect_relationship := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'seam');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT_RELATIONSHIP.RELATING_SHAPE_ASPECT'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set |
      'seam curve relationship' IN WHICH_CLASS(t2_inst))) > 1;
  END_REPEAT;
    WHERE
      WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
shape_aspect: the set of all instances **of shape_aspect** entities.
shape_aspect_relationship: the set of all instances **of shape_aspect_relationship** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **shape_aspect** that has an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'seam' is referenced by one or less instances of **shape_aspect_relationship** that have an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'seam curve relationship' through attribute **relating_shape_aspect**.

5.2.4.317 shape_aspect_for_angle_bar_cross_section_dimensions

The **shape_aspect_for_angle_bar_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'angle bar cross section' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'angle bar cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_angle_bar_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'angle bar cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'angle bar cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to 'angle bar cross section', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'angle bar cross section dimensions'.

5.2.4.318 shape_aspect_for_assembly_manufacturing_position_parameters

The **shape_aspect_for_assembly_manufacturing_position_parameters** rule specifies that an instance of **shape_aspect** with class id 'assembly manufacturing position' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'assembly manufacturing position parameters'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_assembly_manufacturing_position_parameters
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'assembly manufacturing position');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'assembly manufacturing position parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to 'assembly manufacturing position', is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'assembly manufacturing position parameters'.

5.2.4.319 shape_aspect_for_border_corner_cutout_boundary_relationship

The **shape_aspect_for_border_corner_cutout_boundary_relationship** specifies that an instance of **shape_aspect** with class id 'border' is referenced by exactly one **group_assignments** that have the role 'boundary index 2' and that reference a **group** of class 'corner cutout boundary relationship'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_border_corner_cutout_boundary_relationship
FOR(shape_aspect,
  applied_group_assignment, group,
  applied_classification_assignment);
  LOCAL
    c_a_set    : SET OF applied_classification_assignment := [];
    t1_set     : SET Of shape_aspect := [];
    gr_ass_set : BAG OF applied_group_assignment := [];
    violation  : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'border');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     gr_ass_set := USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS');

    violation := SIZEOF(QUERY(aga <* gr_ass_set |
       (aga.role.name = 'boundary index 2') AND
       ('corner cutout boundary relationship'
       IN WHICH_CLASS(aga.assigned_group))
       )) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances **of shape_aspect** entities.
applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **shape_aspect** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'border' is to have one **applied_-group_assignments** whose **role** equals 'boundary index 2' and whose **assigned_group** has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'corner cutout boundary relationship'.

5.2.4.320 shape_aspect_for_bulbflat_cross_section_dimensions

The **shape_aspect_for_bulbflat_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'bulbflat cross section' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'bulbflat cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_bulbflat_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'bulbflat cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'bulbflat cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘bulbflat cross section’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘bulbflat cross section dimensions’.

5.2.4.321 shape_aspect_for_channel_cross_section_dimensions

The **shape_aspect_for_channel_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'channel cross section' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'channel cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_channel_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'channel cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'channel cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to 'channel cross section', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'channel cross section dimensions'.

5.2.4.322 shape_aspect_for_circular_hollow_cross_section_dimensions

The **shape_aspect_for_circular_hollow_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'circular hollow cross section' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'circular hollow cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_circular_hollow_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'circular hollow cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'circular hollow cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to 'circular hollow cross section', is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'circular hollow cross section dimensions'.

5.2.4.323 shape_aspect_for_composite_feature_design_parameters

The **shape_aspect_for_composite_feature_design_parameters** rule specifies that an instance of **shape_aspect** with class id 'composite feature' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'composite feature design parameters'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_composite_feature_design_parameters
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'composite feature');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'composite feature design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to 'composite feature', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'composite feature design parameters'.

5.2.4.324 shape_aspect_for_composite_feature_has_at_least_two_instances

The **shape_aspect_for_composite_feature_has_at_least_two_instances** rule specifies that an instance of shape_aspect with class id 'composite feature' is referenced by exactly two instances of **shape_aspect_relationship** via **relating_shape_aspect**.

EXPRESS specification:

```
*)
RULE shape_aspect_for_composite_feature_has_at_least_two_instances
FOR (shape_aspect, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    classed_s_a_set: SET OF shape_aspect := [];
    t2_set    : SET OF shape_aspect := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with class name
  'composite feature' *)
  c_a_set := QUERY(i <* applied_classification_assignment
  |i.assigned_class.name = 'composite feature');
  (* get all instances of shape_aspect that are of class 'composite
  feature'
   assumed that all items that are classed like this are shape_aspects *)
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
           classed_s_a_set:= classed_s_a_set + c_a_set[i].items[j];
     END_REPEAT;
  END_REPEAT;
  c_a_set := [];
  REPEAT i := 1 TO HIINDEX(classed_s_a_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN (classed_s_a_set[1],
    'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT_RELATIONSHIP.'+
    'RELATING_SHAPE_ASPECT'));
    violation := NOT (SIZEOF(T2_SET) = 2);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equals 'composite feature', is referenced by the **relating_shape_aspect** attribute of exactly two **shape_aspect_relationships**.

5.2.4.325 shape_aspect_for_corner_cutout

The **shape_aspect_for_corner_cutout** specifies that an instance of **shape_aspect** with class id 'corner cutout' is referenced by exactly one **group_assignments** that has the **role** 'item 1' and that reference a **group** of class 'structural cutout boundary relationship'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_corner_cutout
FOR(shape_aspect,
   applied_group_assignment, group,
   applied_classification_assignment);
   LOCAL
     c_a_set    : SET OF applied_classification_assignment := [];
     t1_set     : SET Of shape_aspect := [];
     gr_ass_set : BAG OF applied_group_assignment := [];
     violation  : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME = 'corner cutout');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     gr_ass_set := USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS');
     violation := SIZEOF(QUERY(aga <* gr_ass_set |
       (aga.role.name = 'item 1') AND
       ('structural cutout boundary relationship'
       IN WHICH_CLASS(aga.assigned_group)))) <> 1;
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances **of shape_aspect** entities.
applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **shape_aspect** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'corner cutout' is to have one **applied_group_assignment** whose **role** equals 'item 1' and whose **assigned_group** has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'structural cutout boundary relationship'.

5.2.4.326 shape_aspect_for_corner_cutout_boundary_relationship

The **shape_aspect_for_corner_cutout_boundary_relationship** specifies that an instance of **shape_aspect** with class id 'corner cutout' is referenced by exactly one **group_assignments** that has the **role** 'item 1' and that reference a **group** of class 'corner cutout boundary relationship'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_corner_cutout_boundary_relationship
FOR(shape_aspect,
   applied_group_assignment, group,
   applied_classification_assignment);
   LOCAL
     c_a_set    : SET OF applied_classification_assignment := [];
     t1_set     : SET Of shape_aspect := [];
     gr_ass_set : BAG OF applied_group_assignment := [];
     violation  : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME = 'corner cutout');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     gr_ass_set := USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS');
     violation := SIZEOF(QUERY(aga <* gr_ass_set |
          (aga.role.name = 'item 1') AND
          ('corner cutout boundary relationship'
          IN WHICH_CLASS(aga.assigned_group)))) <> 1;
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances **of shape_aspect** entities.
applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **shape_aspect** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'corner cutout' is to have one **applied_group_assignment** whose **role** equals 'item 1' and whose **assigned_group** has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'corner cutout boundary relationship'.

5.2.4.327 shape_aspect_for_corrugation_dimensions

The **shape_aspect_for_corrugation_dimensions** rule specifies that an instance of **shape_aspect** with class id 'corrugation' is referenced by exactly one instance of **property_definition_-representation** via **definition** whose attribute **name** has the value 'corrugation dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_corrugation_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'corrugation');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'corrugation dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘corrugation’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘corrugation dimensions’.

5.2.4.328 shape_aspect_for_explicit_profile_cross_section_dimensions

The **shape_aspect_for_explicit_profile_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'explicit profile cross section' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value ' explicit profile cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_explicit_profile_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'explicit profile cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      ' explicit profile cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘explicit profile cross section’, is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘explicit profile cross section dimensions’.

5.2.4.329 shape_aspect_for_flanged_plate_cross_section_dimensions

The **shape_aspect_for_flanged_plate_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'flanged plate cross section' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'flanged plate cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_flanged_plate_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'flanged plate cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'flanged plate cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘flanged plate cross section’, is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘flanged plate cross section dimensions’.

5.2.4.330 shape_aspect_for_flare_area_parameters

The **shape_aspect_for_flare_area_parameters** rule specifies that an instance of **shape_aspect** with class id 'flare area' is referenced by exactly two instances of **property_definition_representation** via **definition** whose attribute **name** has the value 'flare area parameters'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_flare_area_parameters
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.NAME = 'flare area');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'flare area parameters')) = 2);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘flare area’, is referenced by exactly two **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘flare area parameters’.

5.2.4.331 shape_aspect_for_flat_bar_cross_section_dimensions

The **shape_aspect_for_flat_bar_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'flat bar cross section' is referenced by exactly one instance of **property_-definition_representation** via **definition** whose attribute **name** has the value 'flat bar cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_flat_bar_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'flat bar cross section');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'flat  bar cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘flat bar cross section’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘flat bar cross section dimensions’.

5.2.4.332 shape_aspect_for_panel_system_curve_boundary_design

The **shape_aspect_for_panel_system_curve_boundary_design** rule specifies that an instance of **shape_aspect** with class id 'panel system curve boundary' is referenced by exactly one instance of **property_definition_representation** via **definition** which has an attribute **name** with the value 'panel system curve boundary design parameters'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_panel_system_curve_boundary_design
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'panel system curve boundary');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'panel system curve boundary design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to 'panel system curve boundary', is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'panel system curve boundary design parameters'.

5.2.4.333 shape_aspect_for_panel_system_plane_boundary_design

The **shape_aspect_for_panel_system_plane_boundary_design** rule specifies that an instance of **shape_aspect** with class id 'panel system plane boundary' is referenced by exactly one instance of **property_definition_representation** via **definition** which has an attribute **name** with the value 'panel system plane boundary design parameters'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_panel_system_plane_boundary_design
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'panel system plane boundary');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'panel system plane boundary design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:
WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to 'panel system plane boundary', is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'panel system plane boundary design parameters'.

5.2.4.334 shape_aspect_for_profile_curve_trace_line_design_parameters

The **shape_aspect_for_profile_curve_trace_line_design_parameters** rule specifies that an instance of **shape_aspect** with class id 'profile curve trace line' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'profile curve trace line design parameters'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_profile_curve_trace_line_design_parameters
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'profile curve trace line');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'profile curve trace line design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘profile curve trace line’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘profile curve trace line design parameters’.

5.2.4.335 shape_aspect_for_round_bar_cross_section_dimensions

The **shape_aspect_for_round_bar_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'round bar cross section' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'round bar cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_round_bar_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'round bar cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'round bar cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to 'round bar cross section', is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals 'round bar cross section dimensions'.

5.2.4.336 shape_aspect_for_square_tube_cross_section_dimensions

The **shape_aspect_for_square_tube_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'square tube cross section' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'square tube cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_square_tube_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'square tube cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'square tube cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘square tube cross section’, is referenced by exactly one **property_definition_-representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘square tube cross section dimensions’.

5.2.4.337 shape_aspect_for_structural_cutout_boundary_relationship

The **shape_aspect_for_structural_cutout_boundary_relationship** specifies that an instance of **shape_aspect** with class id 'structural cutout' is referenced by exactly one **group_assignments** that has the **role** 'item 1' and that reference a **group** of class 'structural cutout boundary relationship'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_structural_cutout_boundary_relationship
FOR(shape_aspect,
   applied_group_assignment, group,
   applied_classification_assignment);
   LOCAL
     c_a_set    : SET OF applied_classification_assignment := [];
     t1_set     : SET Of shape_aspect := [];
     gr_ass_set : BAG OF applied_group_assignment := [];
     violation  : LOGICAL := FALSE;
   END_LOCAL;
   c_a_set := QUERY(i <* applied_classification_assignment |
                    i.assigned_class.NAME = 'structural cutout');
   REPEAT i := 1 TO HIINDEX(c_a_set);
     REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
       t1_set := t1_set + c_a_set[i].items[j];
     END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
     gr_ass_set := USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS');
     violation := SIZEOF(QUERY(aga <* gr_ass_set |
       (aga.role.name = 'item 1') AND
       ('structural cutout boundary relationship'
       IN WHICH_CLASS(aga.assigned_group)))) <> 1;
   END_REPEAT;
   WHERE
     WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances **of shape_aspect** entities.
applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **shape_aspect** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'structural cutout' is to have one **applied_group_assignment** whose **role** equals 'item 1' and whose **assigned_group** has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'structural cutout boundary relationship'.

5.2.4.338 shape_aspect_for_t_bar_cross_section_dimensions

The **shape_aspect_for_t_bar_cross_section_dimensions** rule specifies that an instance of **shape_-aspect** with class id 't bar cross section' is referenced by exactly one instance of **property_-definition_representation** via **definition** whose attribute **name** has the value 't bar cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_t_bar_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   't bar cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      't bar cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘t bar cross section’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘t bar cross section dimensions’.

5.2.4.339 shape_aspect_for_twist_location_design_parameters

The **shape_aspect_for_twist_location_design_parameters** rule specifies that an instance of **shape_aspect** with class id 'twist location' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'twist location design parameters'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_twist_location_design_parameters
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'twist location');

  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'twist location design parameters')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘twist location’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘twist location design parameters’.

5.2.4.340 shape_aspect_for_w_shape_cross_section_dimensions

The **shape_aspect_for_w_shape_cross_section_dimensions** rule specifies that an instance of **shape_aspect** with class id 'w shape cross section' is referenced by exactly one instance of **property_definition_representation** via **definition** whose attribute **name** has the value 'w shape cross section dimensions'.

EXPRESS specification:

```
*)
RULE
shape_aspect_for_w_shape_cross_section_dimensions
FOR(shape_aspect,
  property_definition_representation, group,
  applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF property_definition_representation := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'w shape cross section');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.name =
      'w shape cross section dimensions')) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

shape_aspect: the set of all instances of **shape_aspect** entities.
property_definition_representation: the set of all instances of **property_definition_-representation** entities.
group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **shape_aspect** that is referenced by an **applied_classification_assignment** with **name** equal to ‘w shape cross section’, is referenced by exactly one **property_definition_representation** attribute **definition**, where **property_definition_representation** attribute **name** equals ‘w shape cross section dimensions’.

5.2.4.341 shape_aspect_relationship_attributes_references_are_distinct

The **shape_aspect_relationship_attributes_references_are_distinct** rule specifies the instances that are related by the attributes **relating_shape_aspect** and **related_shape_aspect** of the relationship **shape_aspect_relationship** is different.

EXPRESS specification:

```
*)
RULE shape_aspect_relationship_attributes_references_are_distinct
FOR (shape_aspect_relationship);
  LOCAL
    cyclic_relationship : LOGICAL := FALSE;
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(shape_aspect_relationship)
                  WHILE NOT cyclic_relationship;
    cyclic_relationship:=
shape_aspect_relationship[i].relating_shape_aspect :=:
      shape_aspect_relationship[i].related_shape_aspect;
  END_REPEAT;
  WHERE
    WR1: NOT cyclic_relationship;
END_RULE;
(*
```

Argument definitions:

shape_aspect_relationship: the set of all instances **of shape_aspect_relationship** entities.

Formal propositions:

WR1: The entity instances referenced by attributes **relating_shape_aspect** and **related_shape_-aspect** instance of a **shape_aspect_relationship** must not be identical instances.

5.2.4.342 shape_aspect_relationship_with_definable_object

The **shape_aspect_relationship_with_definable_object** rule specifies a string type attribute **shape_aspect_relationship** of type 'definable object' to match the pattern '*.*' if the instance of **shape_aspect_relationship** is of class 'definable object'.

EXPRESS specification:

```
*)
RULE
shape_aspect_relationship_with_definable_object
FOR (group, applied_classification_assignment);
  LOCAL
    clfied_inst :SET OF GENERIC := [];
  END_LOCAL;

  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'definable object') THEN
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
  END_IF;
  END_REPEAT;
```



```
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **shape_aspect_relationship** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'definable object' is to match the pattern '*.*'.

5.2.4.343 shape_aspect_relationship_with_definition_matches_pattern

The **shape_aspect_relationship_with_definition_matches_pattern** rule specifies a string type attribute **shape_aspect_relationship** of type 'definition' to match the pattern '*.*' if the instance of **shape_aspect_relationship** is of class 'definition'.

EXPRESS specification:

```
*)
RULE
shape_aspect_relationship_with_definition_matches_pattern
FOR (group, applied_classification_assignment);
  LOCAL
    clfied_inst : SET OF GENERIC := [];
  END_LOCAL;

  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'DEFINITION') THEN
      clfied_inst := clfied_inst +
         applied_classification_assignment[i].items;
   END_IF;
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Attribute id of every instance of **shape_aspect_relationship** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'definition' is to match the pattern '*.*'.

5.2.4.344 shape_aspect_relationship_with_identification_assignment

The **shape_aspect_relationship_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE shape_aspect_relationship_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect_relationship := [];
    t2_set    : SET OF shape_aspect_relationship := [];
    arg_list  : LIST OF STRING := ['composite feature',
        'seam curve relationship'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id  *)
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =  arg_LIST[j]);
   END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
   'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **shape_aspect_relationship** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'composite feature' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.345 shape_aspect_with_definable_object_matches_pattern

The **shape_aspect_with_definable_object_matches_pattern** rule specifies a string type attribute **shape_aspect** of type 'definable object' to match the pattern '*.*' if the instance of **shape_aspect** is of class 'definable object'.

EXPRESS specification:

```
*)
RULE
shape_aspect_with_definable_object_matches_pattern
FOR (group, applied_classification_assignment);
  LOCAL
    clfied_inst : SET OF GENERIC := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment);
    IF (applied_classification_assignment[i].assigned_class.name =
    'definable object') THEN
      clfied_inst := clfied_inst +
      applied_classification_assignment[i].items;
  END_IF;
  END_REPEAT;
  WHERE
    WR1: SIZEOF(QUERY(i <* clfied_inst | NOT(i.id LIKE '*.*')) )= 0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:
WR1: Attribute id of every instance of **shape_aspect** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'definable object' is to match the pattern '*.*'.

5.2.4.346 shape_aspect_with_identification_assignment

The **shape_aspect_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE shape_aspect_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF shape_aspect := [];
    t2_set    : SET OF shape_aspect := [];
    arg_list  : LIST OF STRING := ['composite feature',
                                   'seam',
                                   'structural cutout'];
    violation : LOGICAL := FALSE;
```



```
  END_LOCAL;
  (* get all classification_assignment instances with id  *)
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =  arg_LIST[j]);
   END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
  'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
    END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Eery instance of **shape_aspect** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'composite feature' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.347 ship_designation_has_one_specified_names

The **ship_designation_has_one_specified_names** rule specifies that an instance of **product_-definition** with class id 'ship designation' is referenced by exactly one instance of **applied_-identification_assignment** via **items** whose attribute **role** that defines an entity with an attribute **name** has either the value 'imo number' or 'pennant hull number'.

EXPRESS specification:

```
*)
RULE ship_designation_has_one_specified_names
FOR(product_definition,
    applied_identification_assignment, group,
    applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF product_definition := [];
    t2_set    : SET OF applied_identification_assignment := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'ship designation');
  REPEAT i := 1 TO HIINDEX(c_a_set);
```

```
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(QUERY(t2_inst <* t2_set | (t2_inst.role.name =
      'imo number')OR (t2_inst.role.name = 'pennant hull number') )) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
product_definition: the set of all instances of **product_definition** entities.
applied_identification_assignment: the set of all instances of **applied_identification_assignment** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equal to 'ship designation' is referenced by exactly one instance of **applied_identification_assignment** through attribute **items** whose attribute **role** is an **identification_role** with attribute **name** equal to either 'imo number' or 'pennant hull number'.

5.2.4.348 ship_material_property_has_at_most_one_named_references

The **ship_material_property_has_at_most_one_named_references** specifies that an instance of **property_definition** with class id 'ship material property' is referenced by at most one instance of **applied_document_reference** via attribute **items** whose attribute **role.name** has the value 'material reference' **property_definition** equal to referenced instance.

EXPRESS specification:

```
*)
RULE ship_material_property_has_at_most_one_named_references
FOR(property_definition,
    applied_document_reference,
    applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF property_definition := [];
    t2_set    : SET OF applied_document_reference := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'ship material
  property' *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.ASSIGNED_CLASS.NAME = 'ship material property');
```



```
  (* get all instances of property_definition that have class id 'ship
  material property' *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* for all instances of property_definition in t1_set:
     get the applied_document_reference instances that are referencing a
  property_definition instance via items,
     filter out those applied_document_reference instances whose attribute
  role.name has the value 'material reference'
     check if their number equals 1
   *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i], 'SHIP_STRUCTURES_SCHEMA.' +
'APPLIED_DOCUMENT_REFERENCE.ITEMS'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set | t2_inst.role.name =
'material reference')) > 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

property_definition: the set of all instances of **property_definition** entities.
applied_document_reference: the set of all instances of **applied_document_reference** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **property_definition** that has an **applied_classification_assignment** whose **assigned_classification** has a **name** attribute of value 'ship material property' is referenced by one or less instances of **applied_document_reference** whose attribute **role.name** has a value of 'material reference' through attribute **items**.

5.2.4.349 source_for_library_definition

The **source_for_library_definition** rule specifies the **items** attribute of a **representation** is to have for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list, if the **representation** is of class 'source'. There is not to be **representation_items** with other names than those in the list of names.

EXPRESS specification:

```
*)
RULE source_for_library_definition
FOR (representation, representation_item, group,
     applied_classification_assignment);
  LOCAL
    c_a_set         : SET OF applied_classification_assignment := [];
    classed_rep_set : SET OF representation := [];
    arg_list        : LIST OF STRING := ['library definition'];
```

```
    violation        :LOGICAL := FALSE;
  END_LOCAL;
  (* find all classification_assignment instances with class name 'source'
  *)
  c_a_set := QUERY(i <* applied_classification_assignment |
                  i.ASSIGNED_CLASS.NAME = 'source');
  (* get all instances of representation that are of class 'source'
   assumed that all items that are classed like this are representations *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      classed_rep_set:= classed_rep_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  (* iterate over all representations found above; stop, if one of
     them has not exactly one rep_item for each name of the arg_list
   *)
  REPEAT i:=1 TO HIINDEX(classed_rep_set) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
      violation := (SIZEOF(QUERY(rep_item <* classed_rep_set[i].items |
                    rep_item.name = arg_list[j])) <> 1);
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **representation_item** entities.
representation_item: the set of all instances of **group** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every instance of **representation** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equal to 'source' for each value out of 'library definition' collect exactly and only one instance of **representation_item** in its attribute **items** whose **name** attribute equals that value.

5.2.4.350 spacing_position_compound_representation_has_name

The **spacing_position_compound_representation_has_name** rule specifies the 'item_element' attribute of a **compound_representation_item** with the class id 'spacing position' to have in the list of **representation_item** for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list.

EXPRESS specification:

```
*)
RULE spacing_position_compound_representation_has_name
FOR (compound_representation_item, group,
     applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
```



```
    t1_set    : SET OF Compound_representation_item := [];
    t2_set    : SET Of representation_item := [];
    arg_list  : LIST OF STRING := ['position number', 'position'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'spacing position');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i:=1 TO HIINDEX(t1_set) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        t2_set := t1_set[i].item_element;
        violation := (SIZEOF(QUERY(items <* t2_set |
           items.name = arg_list[j])) <> 1);
  END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
compound_representation_item: the set of all instances of **compound_representation_item** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **compound_representation_item** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equals 'spacing position' is to have its attribute **item_element** instantiated as a **list_-representation_item** which is for each value out of 'position number', or 'position' collect exactly one instance of **representation_item** whose **name** attribute equals that value.

5.2.4.351 spacing_position_with_offset_compound_representation_-has_class

The **spacing_position_with_offset_compound_representation_has_class** rule specifies the 'item_-element' attribute of a **compound_representation_item** with the class id 'spacing position with offset' to have in the list of **representation_item** exactly one **compound_representation_item** with the class id 'spacing position'.

EXPRESS specification:

```
*)
RULE spacing_position_with_offset_compound_representation_has_class
FOR (compound_representation_item, group,
     applied_classification_assignment);
  LOCAL
```

```
    c_a_set   : SET OF applied_classification_assignment := [];
    c_a_set2  : SET OF applied_classification_assignment := [];
    t1_set    : SET OF COMPOUND_REPRESENTATION_ITEM := [];
    t2_set    : SET OF COMPOUND_REPRESENTATION_ITEM := [];
    t3_set    : SET Of representation_item := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'spacing position with offset');
     REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  c_a_set2 := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME = 'spacing position');
  REPEAT i := 1 TO HIINDEX(c_a_set2);
    REPEAT j := 1 TO HIINDEX(c_a_set2[i].items);
      t2_set := t2_set + c_a_set2[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i:=1 TO HIINDEX(t1_set) WHILE (NOT violation);
    REPEAT j := 1 TO HIINDEX(t1_set[i].item_element);
      t3_set := t3_set + t1_set[i].item_element;
    END_REPEAT;
    violation := (SIZEOF(t3_set * t2_set) <> 1);
    t3_set:= [];
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
compound_representation_item: the set of all instances of **compound_representation_item** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **compound_representation_item** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equals 'spacing position with offset' is to have its attribute **item_element** instantiated as a **list_representation_item** which collect exactly one instance of **compound_representation_-item** that has an **applied_classification_assignment** whose attribute **assigned_classification** equals 'spacing position'.

5.2.4.352 spacing_position_with_offset_compound_representation_-has_name

The **spacing_position_with_offset_compound_representation_has_name** rule specifies the 'item_element' attribute of a **compound_representation_item** with the class id 'spacing position with offset' to have in the list of **representation_item** for each entry in the list exactly one **representation_item** whose **name** attribute has the value given in the list.

EXPRESS specification:

```
*)
RULE spacing_position_with_offset_compound_representation_has_name
FOR (compound_representation_item, group,
     applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF Compound_representation_item := [];
    t2_set    : SET Of representation_item := [];
    arg_list  : LIST OF STRING := ['offset'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =
                   'spacing position with offset');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i:=1 TO HIINDEX(t1_set) WHILE (NOT violation);
    REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        t2_set := t1_set[i].item_element;
        violation := (SIZEOF(QUERY(items <* t2_set |
        items.name = arg_list[j])) <> 1);
 END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
compound_representation_item: the set of all instances of **compound_representation_item** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **compound_representation_item** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equals 'spacing position with offset' is to have its attribute **item_element** instantiated as a **list_representation_item** which is for each value out of 'offset' collect exactly one instance of **representation_item** whose **name** attribute equals that value.

5.2.4.353 structural_class_rejection_has_at_least_one_role_references

The **structural_class_rejection_has_at_least_one_role_references** rule specifies that an instance of **group** with class id 'structural class rejection' is referenced by at least one instance of **applied_document_reference** via **items** whose attribute **applied_document_reference.role.name** has the value 'explanations'.

EXPRESS specification:

```
*)
RULE structural_class_rejection_has_at_least_one_role_references
FOR(group, applied_document_reference, applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF group := [];
    t2_set    : SET OF applied_document_reference := [];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.name = 'structural class rejection');
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.APPLIED_DOCUMENT_REFERENCE.ITEMS'));
    violation := SIZEOF(QUERY(t2_inst <* t2_set |
       t2_inst.role.name = 'explanations')) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_document_reference: the set of all instances of applied_document_reference entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **group** that has an **applied_classification_assignment** whose attribute **assigned_classification** is a **group** with attribute **name** equals 'structural class rejection' is referenced by one or more instances of **applied_document_reference** whose attribute **role.name** equals 'explanations' through attribute **items**.

5.2.4.354 structural_part_symmetry_relationship_has_product_definition

The **structural_part_symmetry_relationship_has_product_definition** rule specifies the relating and the related side of a relationship of type **product_definition_relationship** is to be of the same class if **product_definition_relationship** is of class 'structural part symmetry relationship'.

EXPRESS specification:

```
*)
RULE
structural_part_symmetry_relationship_has_product_definition
FOR (group, applied_classification_assignment);
  LOCAL
    cl_ass      : SET OF applied_classification_assignment := [];
    clfied_inst : SET OF product_definition_RELATIONSHIP := [];
  END_LOCAL;
  cl_ass := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.name =
                  'structural part symmetry relationship');
  REPEAT i := 1 TO HIINDEX(cl_ass);
    clfied_inst := clfied_inst + cl_ass[i].items;
  END_REPEAT;
  WHERE
    WR1:  SIZEOF(QUERY(pdr <* clfied_inst |
          (NOT(SIZEOF(WHICH_CLASS(pdr.relating_product_definition)) >0))
           )) -
      SIZEOF(QUERY(pdr <* clfied_inst |
          (NOT(SIZEOF(WHICH_CLASS(pdr.related_product_definition)) >0))
           ))
           =0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: For every entity instance of **product_definition_relationship** that is referenced by an **applied_classification_assignment** whose **assigned_classification** has an entity with **name** attribute equal to 'structural part symmetry relationship', the instances referenced by the attributes **relating_product_definition** and **related_product_definition** are referenced by applied_classification_assignments whose assigned_classifications have identical values in their **name** attributes.

5.2.4.355 structural_system_symmetry_relationship_has_product_definition

The **structural_system_symmetry_relationship_has_product_definition** rule specifies the relating and the related side of a relationship of type **product_definition_relationship** is to be of the same class if **product_definition_relationship** is of class 'structural system symmetry relationship'.

EXPRESS specification:

```
*)
RULE
structural_system_symmetry_relationship_has_product_definition
FOR (group, applied_classification_assignment);
  LOCAL
    cl_ass      : SET OF applied_classification_assignment := [];
```

```
    clfied_inst : SET OF product_definition_RELATIONSHIP := [];
  END_LOCAL;
  cl_ass := QUERY(i <* applied_classification_assignment |
                  i.assigned_class.name =
                  'structural system symmetry relationship');
  REPEAT i := 1 TO HIINDEX(cl_ass);
    clfied_inst := clfied_inst + cl_ass[i].items;
  END_REPEAT;
  WHERE
    WR1:  SIZEOF(QUERY(pdr <* clfied_inst |
         (NOT(SIZEOF(WHICH_CLASS(pdr.relating_product_definition)) >0))
          )) -
      SIZEOF(QUERY(pdr <* clfied_inst |
         (NOT(SIZEOF(WHICH_CLASS(pdr.related_product_definition)) >0))
          ))
          =0;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: For every entity instance of **product_definition_relationship** that is referenced by an **applied_classification_assignment** which has an **assigned_classification** has an entity with **name** attribute equal to 'structural system symmetry relationship', the instances referenced by the attributes **relating_product_definition** and **related_product_definition** are referenced by applied_classification_assignments which has an assigned_classifications have identical values in their **name** attributes.

5.2.4.356 unique_approvals_in_approval_history

The **unique_approvals_in_approval_history** rule specifies that all instances of **approval** in a **group** with class **approval_history** must be unique.

EXPRESS specification:

```
*)
RULE
unique_approvals_in_approval_history
FOR (group, applied_group_assignment);
  LOCAL
    t1_set  : SET OF group := [];
    t2_set  : SET OF applied_group_assignment := [];
    t3_set  : SET OF GENERIC :=[];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* group | VALUE_IN(WHICH_CLASS(i),
    'APPROVAL HISTORY'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
   t2_set := QUERY(a <* applied_group_assignment |
     a.ASSIGNED_GROUP = t1_set[i]);
   t3_set := QUERY(b <* t2_set[1].items |
```



```
      'SHIP_STRUCTURES_SCHEMA.APPROVAL' IN TYPEOF(b));
    violate := NOT (VALUE_UNIQUE(t3_set));
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.

Formal propositions:

WR1: All instances of **approval** that are items of an instance of **group** that has an **applied_group_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'approval history' must be unique.

WR2: Every **applied_group_assignment** is to have exactly one item in the **items** attribute.

5.2.4.357 user_def_function_description_required

The **user_def_function_description_required** rule specifies a **representation** that has a **descriptive_representation_item** with **name** 'function' and description 'user defined' is to have another **descriptive_representation_item** with **name** 'user def function'.

EXPRESS specification:

```
*)
RULE user_def_function_description_required
FOR (representation, representation_item);
  LOCAL
    violation : LOGICAL := FALSE;
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(REPRESENTATION) WHILE NOT violation;
    violation := (SIZEOF(QUERY(r <* Representation[i].ITEMS |
     ('SHIP_STRUCTURES_SCHEMA.DESCRIPTIVE_REPRESENTATION_ITEM'
     IN TYPEOF(r)) AND
     (r.NAME = 'function') AND
     (r\DESCRIPTIVE_REPRESENTATION_ITEM.DESCRIPTION =
     'user defined'))) > 0)
     AND
     (SIZEOF(QUERY(r <* representation[i].ITEMS |
     (r.NAME = 'user def function'))) = 0);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **representation** that has a **descriptive_representation_item** with attribute **name** equal to 'function' and attribute **description** equal to 'user defined' in its set of **items** has another **descriptive_representation_item** with attribute **name** equal to 'user def function' in its set of **items**.

5.2.4.358 user_defined_requires_user_defined_class

The **user_defined_requires_user_defined_class** rule specifies an instance of type **product_-definition** that has an **applied_classification_assignment** with **role** that defines an entity with an attribute **name** equal to 'class membership' and group.name equal to 'user defined' to have another **applied_classification_assignment** with **role** that defines an entity with an attribute **name** equal to 'class membership' but group.name is equal to 'user defined class'.

EXPRESS specification:

```
*)
RULE user_defined_requires_user_defined_class
FOR (product_definition, group, applied_classification_assignment);
  LOCAL
    cl_ass      : SET OF applied_classification_assignment := [];
    clfied_inst : SET OF GENERIC := [];
    violation   : LOGICAL := FALSE;
  END_LOCAL;
  cl_ass := QUERY(i <* applied_classification_assignment |
                  (i.ROLE.NAME = 'class membership') AND
                  (i.assigned_class.NAME = 'user defined'));
  REPEAT i := 1 TO HIINDEX(cl_ass);
    clfied_inst := clfied_inst + cl_ass[i].items;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(clfied_inst) WHILE NOT violation;
    violation := NOT('user defined class' IN WHICH_CLASS(clfied_inst[i]));
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

group: the set of all instances of **group** entities.
product_definition: the set of all instances of **product_definition** entities.
applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **product_definition** that has an **applied_classification_assignment** which has an attribute **role** identifies an object with attribute **name** equals 'class membership' and which has an attribute **assigned_class** is a **group** with attribute **name** equals 'user defined' must also have an **applied_classification_assignment** which has an attribute **role** is a **classification_role** with attribute **name** equals 'class membership' and which has an attribute **assigned_class** is a **group** with attribute **name** equals 'user defined class'.

5.2.4.359 user_defined_tightness_description_required

The **user_defined_tightness_description_required** rule specifies a **representation** that has a **descriptive_representation_item** with attribute **name** equal to 'tightness' and description 'user defined' to have another **descriptive_representation_item** with attribute **name** equal to 'user defined tightness'.

EXPRESS specification:

```
*)
RULE
user_defined_tightness_description_required
FOR (representation, representation_item);
  LOCAL
    violation : LOGICAL := FALSE;
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(REPRESENTATION) WHILE NOT violation;
    violation := (SIZEOF(QUERY(r <* representation[i].ITEMS |
      ('SHIP_STRUCTURES_SCHEMA.DESCRIPTIVE_REPRESENTATION_ITEM'
      IN TYPEOF(r)) AND
      (r.NAME = 'tightness') AND
      (r\DESCRIPTIVE_REPRESENTATION_ITEM.DESCRIPTION =
      'user defined'))) > 0)
      AND
     (SIZEOF(QUERY(r <* representation[i].ITEMS |
     (r.NAME = 'user defined tightness'))) = 0);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

representation_item: the set of all instances of **representation_item** entities.
representation: the set of all instances of **representation** entities.

Formal propositions:

WR1: Every **representation** that has a **descriptive_representation_item** with atttribute **name** equal to ‘tightness’ and **description** ‘user defined’ in its set of **items** is to have another **descriptive_-representation_item** with **name** attribute equal to ‘user defined tightness’ in its set of **items**.

5.2.4.360 version_creation_has_mandatory_attribute_description

The **version_creation_has_mandatory_attribute_description** rule specifies that for an instance of **action** with class id 'version creation' the optional attribute **description** is instantiated.

EXPRESS specification:

```
*)
RULE
version_creation_has_mandatory_attribute_description
FOR (action);
  LOCAL
    t1_set  : SET OF action := [];
```



```
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* action | VALUE_IN(WHICH_CLASS(i),
  'version creation'));
  violate := (SIZEOF(QUERY(k <* t1_set |
  NOT EXISTS (k.description))) > 0);
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

action: the set of all instances of **action** entities.

Formal propositions:

WR1: The optional attribute **description** exist for every instance of **action** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'version creation'.

5.2.4.361 version_deletion_has_mandatory_attribute_description

The **version_deletion_has_mandatory_attribute_description** rule specifies that for an instance of **action** with class id 'version deletion' the optional attribute **description** is instantiated.

EXPRESS specification:

```
*)
RULE
version_deletion_has_mandatory_attribute_description
FOR (action);
  LOCAL
    t1_set  : SET OF action := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* action | VALUE_IN(WHICH_CLASS(i),
  'version deletion'));
  violate := (SIZEOF(QUERY(k <* t1_set |
  NOT EXISTS (k.description))) > 0);
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

action: the set of all instances of **action** entities.

Formal propositions:

WR1: The optional attribute **description** exist for every instance of **action** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'version deletion'.

5.2.4.362 version_history_has_exactly_one_assigned_group

The **version_history_has_exactly_one_assigned_group** rule specifies that each instance of type **group** with class 'version history' is referenced by exactly one assignment of type **applied_group_assignment** via attribute **assigned_group**.

EXPRESS specification:

```
*)
RULE version_history_has_exactly_one_assigned_group
FOR(applied_group_assignment, group);
  LOCAL
    t1_set  : SET OF group := [];
    set_1, set_2: SET OF applied_group_assignment := [];
    set_3   : SET OF GENERIC := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* group | VALUE_IN(WHICH_CLASS(a), 'version
  history'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
   set_1 := QUERY(b <* applied_group_assignment |
         (b.assigned_group = t1_set[i]) AND (b.role.name = 'current
  version'));
   set_2 := QUERY(c <* applied_group_assignment |
               (c.assigned_group = t1_set[i]) AND (c.role.name =
  'members'));
   violate := ((SIZEOF(set_1) <> 1) OR (SIZEOF(set_2) <> 1));
   IF not violate THEN
      set_3 := set_1[1].items * set_2[1].items;
      violate := (SIZEOF(set_3) <> 1) OR
                    NOT (VALUE_IN(WHICH_CLASS(set_3[1]), 'versionable
   object'));
   END_IF;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.

Formal propositions:

WR1: Every instance of **group** that has an **applied_classification_assignment** whose **assigned_-classification** identifies an entity with attribute **name** equals 'version history' is referenced by exactly one instance of type **applied_group_assignment** through attribute **assigned_group**.

5.2.4.363 version_history_referenced_by_exactly_one_current_version

The **version_history_referenced_by_exactly_one_current_version** rule specifies that each instance of type **group** with class 'version history' is referenced by exactly one assignment of type **applied_group_assignment** with **role** 'current version' via attribute **assigned_group**.

EXPRESS specification:

```
*)
RULE version_history_referenced_by_exactly_one_current_version
FOR(applied_group_assignment, group);
  LOCAL
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  (* get all instances of group with class 'version history' *)
  t1_set := QUERY(a <* group | VALUE_IN(WHICH_CLASS(a), 'version
  history'));
  (* for all instances found above *)
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
  (* get the instances of applied_group_assignment that have a role
  'current version' *)
   a_set := QUERY(b <* applied_group_assignment |
              (b.assigned_group = t1_set[i]) AND (b.role.name = 'current
  version'));
  (* there is no such instances *)
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.

Formal propositions:

WR1: Every instance of **group** that has an **applied_classification_assignment** whose **assigned_-classification** identifies an entity with attribute **name** equals 'version history' is referenced by exactly one instance of type **applied_group_assignment** whose role equals 'current version' through attribute **assigned_group**.

5.2.4.364 version_history_referenced_by_multiple_roles

The **version_history_referenced_by_multiple_roles** rule specifies that each instance of type **group** with class 'version history' is referenced assignment of type **applied_group_assignment** with **roles** 'versions', 'current version' or 'relationships' via attribute **assigned_group**.

EXPRESS specification:

```
*)
RULE version_history_referenced_by_multiple_roles
FOR(applied_group_assignment, group);
  LOCAL
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* group | VALUE_IN(WHICH_CLASS(a), 'version
  history'));
```



```
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
   a_set := QUERY(b <* applied_group_assignment |
    (b.assigned_group = t1_set[i]) AND NOT (b.role.name IN ['versions',
  'current version', 'relationships']));
    violate := SIZEOF(a_set) > 0;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.

Formal propositions:

WR1: Every instance of **group** that has an **applied_classification_assignment** whose **assigned_classification** identifies an entity with attribute **name** equals 'version history' is only referenced by instances of type **applied_group_assignment** whose **role** equals 'versions', 'current version', or 'relationships' through attribute **assigned_group**.

5.2.4.365 version_modification_has_mandatory_attribute_description

The **version_modification_has_mandatory_attribute_description** rule specifies that for an instance of **action** with class id 'version modification' the optional attribute **description** is instantiated.

EXPRESS specification:

```
*)
RULE
version_modification_has_mandatory_attribute_description
FOR (action);
  LOCAL
    t1_set  : SET OF action := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* action | VALUE_IN(WHICH_CLASS(i),
  'version modification'));
  violate := (SIZEOF(QUERY(k <* t1_set |
  NOT EXISTS (k.description))) > 0);
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

action: the set of all instances of **action** entities.

Formal propositions:

WR1: The optional attribute **description** exist for every instance of **action** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'version modification'.

5.2.4.366 version_relationship_associates_with_versionable_object

The **version_relationship_associates_with_versionable_object** rule specifies an instance **identification_assignment_relationship** with class 'version relationship' only relates instances of type **applied_identification_assignment** with class 'versionable object'.

EXPRESS specification:

```
*)
RULE
version_relationship_associates_with_versionable_object
FOR (applied_identification_assignment);
  LOCAL
    violate1, violate2, violate : LOGICAL := FALSE;
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_identification_assignment) WHILE NOT
  violate;
  violate1 := (SIZEOF(USEDIN(applied_identification_assignment[i],
   'SHIP_STRUCTURES_SCHEMA.IDENTIFICATION_ASSIGNMENT_RELATIONSHIP.'
    + 'RELATING_IDENTIFICATION_ASSIGNMENT')) > 0) AND
   (NOT (VALUE_IN(WHICH_CLASS(applied_identification_assignment[i]),
   'versionable object')));
  violate2 := (SIZEOF(USEDIN(applied_identification_assignment[i],
   'SHIP_STRUCTURES_SCHEMA.IDENTIFICATION_ASSIGNMENT_RELATIONSHIP.'
    + 'RELATED_IDENTIFICATION_ASSIGNMENT')) > 0) AND
   (NOT (VALUE_IN(WHICH_CLASS(applied_identification_assignment[i]),
   'versionable object')));

  violate := violate1 OR violate2;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_identification_assignment: the set of all instances of **applied_identification_assignment** entities.

Formal propositions:

WR1: Every instance of **identification_assignment_relationship** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'version relationship' is only to reference or be referenced by instances of **applied_-identification_assignment** that have an **applied_classification_assignment** whose **assigned_-classification** is a **group** with attribute **name** equals 'versionable object'.

5.2.4.367 version_relationship_has_mandatory_attribute_description

The **version_relationship_has_mandatory_attribute_description** rule specifies that for an instance of **identification_assignment_relationship** with class id 'version relationship' the optional attribute **description** is instantiated.

EXPRESS specification:

```
*)
RULE
version_relationship_has_mandatory_attribute_description
FOR (identification_assignment_relationship);
  LOCAL
    t1_set  : SET OF identification_assignment_relationship := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(i <* identification_assignment_relationship |
  VALUE_IN(WHICH_CLASS(i), 'version relationship'));
  violate := (SIZEOF(QUERY(k <* t1_set | NOT EXISTS (k.description))) >
  0);
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

identification_assignment_relationship: the set of all instances of **identification_assignment_-relationship** entities.

Formal propositions:

WR1: The optional attribute **description** exist for every instance of **identification_assignment_-relationship** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'version relationship'.

5.2.4.368 version_relationship_has_unique_versions

The **version_relationship_has_unique_versions** rule specifies that all instance of **identification_-assignment_relationship** with class 'version relationship', versionable objects successor and predecessor must be different, and must expose different **version_id**.

EXPRESS specification:

```
*)
RULE
version_relationship_has_unique_versions
FOR (identification_assignment_relationship);
  LOCAL
    t1_set:  SET OF identification_assignment_relationship := [];
    violate: LOGICAL := FALSE;
  END_LOCAL;
    t1_set := QUERY(a <* identification_assignment_relationship |
    VALUE_IN(WHICH_CLASS(a), 'version relationship'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    violate :=
```

```
          ( t1_set[i].relating_identification_assignment.assigned_id =
            t1_set[i].related_identification_assignment.assigned_id );
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

identification_assignment_relationship: the set of all instances of **identification_assignment_-relationship** entities.

Formal propositions:

WR1: The **assigned_id** of the **related_identification_assignment** and the **relating_identification_-assignment** of every instance of **identification_assignment_relationship** that has an **applied_classification_assignment** whose **assigned_classification** is a **group** with attribute **name** equals 'version relationship' is distinct.

5.2.4.369 versionable_object_has_one_version_id

The **versionable_object_has_one_version_id** rule specifies that a **versionable_object** is referenced by zero or one **applied_identification_assignment** whose **role** is 'version id' has exactly one reference of this type.

EXPRESS specification:

```
*)
RULE versionable_object_has_one_version_id
FOR(applied_identification_assignment);
  LOCAL
    version_ids         : SET OF applied_identification_assignment := [];
    versionable_objects : BAG OF GENERIC := [];
    duplicate           : LOGICAL := FALSE;
  END_LOCAL;
  version_ids := QUERY(i <* applied_identification_assignment |
                       i.role.name = 'version id');
  REPEAT i := 1 TO HIINDEX(version_ids);
    versionable_objects := versionable_objects + version_ids[i].items;
  END_REPEAT;
   REPEAT i := 1 TO HIINDEX(versionable_objects) WHILE NOT duplicate;
    REPEAT j := i + 1 TO HIINDEX(versionable_objects) WHILE NOT duplicate;
      duplicate := versionable_objects[i] :=: versionable_objects[j];
    END_REPEAT;
  END_REPEAT;
  WHERE
    WR1: NOT duplicate;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every **versionable_object** has zero or one **applied_identification_assignment** whose attribute **role** is 'version id'.

5.2.4.370 versioned_action_request_with_identification_assignment

The **versioned_action_request_with_identification_assignment** rule specifies a list of entities that require an identification. The identification is defined by the **applied_identification_assignment** attribute.

EXPRESS specification:

```
*)
RULE versioned_action_request_with_identification_assignment
FOR (applied_classification_assignment);
  LOCAL
    c_a_set   : SET OF applied_classification_assignment := [];
    t1_set    : SET OF versioned_action_request := [];
    t2_set    : SET OF versioned_action_request := [];
    arg_list  : LIST OF STRING := [ 'definable object'];
    violation : LOGICAL := FALSE;
  END_LOCAL;
  (* get all classification_assignment instances with id 'moulded form
  characteristics definition' *)
  REPEAT j:=1 TO HIINDEX(arg_list) WHILE (NOT violation);
        c_a_set := QUERY(i <* applied_classification_assignment |
                   i.assigned_class.NAME =  arg_LIST[j]);
  END_REPEAT;
  (* get all instances of property_definition that have class id *)
  REPEAT i := 1 TO HIINDEX(c_a_set);
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items);
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
    'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
      violation := NOT (SIZEOF(T2_SET) = 1);
  END_REPEAT;
  WHERE
    WR1: NOT violation;
END_RULE;
(*
```

Argument definitions:

applied_classification_assignment: the set of all instances of **applied_classification_assignment** entities.

Formal propositions:

WR1: Every instance of **property_definition** that is referenced by an **applied_classification_assignment** whose assigned_classification has a **name** attribute of value 'definable object' require an **applied_identification_assignment** to define the instance identifier.

5.2.4.371 versions_is_referenced_by_at_least_one_version_history

The **versions_is_referenced_by_at_least_one_version_history** rule specifies that each instance of type **group** with class 'versions' is referenced by at least one assignment of type **applied_group_assignment** with **role** 'version history' via attribute **assigned_group**.

EXPRESS specification:

```
*)
RULE versions_is_referenced_by_at_least_one_version_history
FOR(applied_group_assignment, group);
  LOCAL
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
    violate : LOGICAL := FALSE;
  END_LOCAL;
  t1_set := QUERY(a <* group | VALUE_IN(WHICH_CLASS(a), 'versions'));
  REPEAT i := 1 TO HIINDEX(t1_set) WHILE NOT violate;
    a_set := QUERY(b <* applied_group_assignment |
          (b.assigned_group = t1_set[i]) AND (b.role.name = 'version
    history'));
    violate := SIZEOF(a_set) < 1;
  END_REPEAT;
  WHERE
    WR1: NOT violate;
END_RULE;
(*
```

Argument definitions:

applied_group_assignment: the set of all instances of **applied_group_assignment** entities.
group: the set of all instances of **group** entities.

Formal propositions:

WR1: Every instance of **group** that has an **applied_classification_assignment** whose **assigned_classification** identifies an entity with attribute **name** equals 'versions' is referenced by one or more instances of type **applied_group_assignment** whose **role** equals 'version history' through attribute **assigned_group**.

5.2.5 Ship structures function definitions

5.2.5.1 which_class

The function **which_class** determines the **applied_classification_assignment**s pointing to an instance of an arbitrary type and returns the class ids in a string list.

EXPRESS specification:

```
*)
FUNCTION WHICH_CLASS(T: GENERIC): LIST OF STRING;
  LOCAL
     elements: BAG OF applied_classification_assignment;
     class_list: LIST OF STRING :=[];
```



```
  END_LOCAL;
  elements :=
      USEDIN(T,
      'SHIP_STRUCTURES_SCHEMA.applied_classification_assignment.ITEMS');
  REPEAT i:=1 TO HIINDEX(elements);
 IF (elements[i]\classification_assignment.role.name = 'class membership')
THEN
  class_list := class_list +
  elements[i]\classification_assignment.assigned_class\group.name;
 END_IF;
  END_REPEAT;
  RETURN(class_list);
END_FUNCTION;

*)
END_SCHEMA;
(*
```

6 Conformance requirements

Conformance to this part of ISO 10303 includes satisfying the requirements stated in this part, the requirements of the implementation methods supported, and the relevant requirements of the normative references.

An implementation shall support at least one of the following implementation methods:

— ISO 10303-21;

— ISO 10303-22;

— ISO 10303-28.

Requirements with respect to implementation methods- specific requirements are specified in annex C.

The Protocol Information Conformance Statement (PICS) proforma lists the options or combinations of options that may be included in the implementation. The PICS proforma is provided in annex D.

This part of ISO 10303 provides a number of options that may be supported by an implementation. These options have been grouped into the following conformance classes:

— Class 1 - 2, ship structures definition and approval data that is created at the preliminary design stage of a ship, has structural definitions and shape representations of this stage, is exchanged between the shipyard and the subcontractor; and, the early class approval data for the preliminary design of the ship, including the definition of hull cross sections, has class approvals with regard to the detailed design definitions, is exchanged between the subcontractor and shipyard, and between the shipyard and the classification society;

— Class 3 - 4 ship structures definition and approval data that is elaborated at the detailed design stage of a ship, under consideration of the production design of ship structures, is exchanged between the shipyard and the subcontractor;

— Class 5 - 8, ship structures definition and approval data that is completed at the product manufacturing stage of a ship, has manufacturing and welding definitions, is exchanged between the design department and the manufacturing department of the shipyard;

— Class 9, class approval data for the structural parts (plates and profiles) of the ship, has class approvals with regard to the manufacturing definitions, is exchanged between the shipyard and the classification society.

Conformance to a particular class requires that all AIM elements defined as part of that class be supported. Tables 1 through 5 define the conformance classes to which each AIM element belongs.

NOTE AP 218 Test Case Definition [3] defines the test cases to be used in the assessment of conformance.

Table 1 — Conformance classes - Early design & class approval

Unit of Functionality	Conformance class	
	Early design & class approval	
	class 1	class 2
class_approvals	X	X
configuration_management	X	X
definitions	X	X
design_loads	X	X
external_references	X	X
hull_class_applicability		X
hull_cross_sections	X	X
items	X	X
library_reference	X	X
location_concept	X	X
product_structures	X	X
shape (edge_based_wireframe)	X	
shape (non_manifold_surface)		X
shape (geometrically_bounded_wireframe)		
shape (advanced_B-rep)		
ship_general_characteristics	X	X
ship_manufacturing_definitions		
ship_materials	X	X
ship_measures	X	X
structural_features		
structural_parts		
structural_systems	X	X
welds		

Table 2 — Conformance classes - Detail design

Unit of Functionality	Conformance class	
	Detail design	
	class 3	class 4
class_approvals		
configuration_management	X	X
definitions	X	X
design_loads	X	X
external_references	X	X
hull_class_applicability		X
hull_cross_sections	X	X
items	X	X
library_reference	X	X
location_concept	X	X
product_structures	X	X
shape (edge_based_wireframe)		
shape (non_manifold_surface)		X
shape (geometrically_bounded_wireframe)		
shape (advanced_B-rep)	X	X
ship_general_characteristics	X	X
ship_manufacturing_definitions		
ship_materials	X	X
ship_measures	X	X
structural_features	X	X
structural_parts	X	X
structural_systems	X	X
welds	X	X

Table 3 — Conformance classes - Manufacturing

Unit of Functionality	Conformance class			
	Manufacturing			
	class 5	class 6	class 7	class 8
class_approvals				
configuration_management	X	X	X	X
definitions	X	X	X	X
design_loads				
external_references	X	X	X	X
hull_class_applicability	X	X		X
hull_cross_sections				
items	X	X	X	X
library_reference	X	X	X	X
location_concept	X	X	X	X
product_structures	X	X	X	X
shape (edge_based_wireframe)	X			
shape (non_manifold_surface)		X		
shape (geometrically_bounded_wireframe)			X	
shape (advanced_B-rep)				X
ship_general_characteristics	X	X	X	X
ship_manufacturing_definitions	X	X	X	X
ship_materials	X	X	X	X
ship_measures	X	X	X	X
structural_features	X	X	X	X
structural_parts	X	X	X	X
structural_systems	X	X	X	X
welds	X	X	X	X

Table 4 — Conformance class - Class approval

Unit of Functionality	Conformance class
	Class approval
	class 9
class_approvals	X
configuration_management	X
definitions	X
design_loads	X
external_references	X
hull_class_applicability	X
hull_cross_sections	X
items	X
library_reference	X
location_concept	X
product_structures	X
shape (edge_based_wireframe)	
shape (non_manifold_surface)	X
shape (geometrically_bounded_wireframe)	
shape (advanced_B-rep)	
ship_general_characteristics	X
ship_manufacturing_definitions	X
ship_materials	X
ship_measures	X
structural_features	X
structural_parts	X
structural_systems	X
welds	X

Table 5 — Conformance class elements

AIM Element	Class								
	1	2	3	4	5	6	7	8	9
ACTION	X	X	X	X	X	X	X	X	X
ACTION_METHOD	X	X	X	X	X	X	X	X	X
ACTION_REQUEST_ASSIGNMENT	X	X	X	X	X	X	X	X	X
ACTION_REQUEST_SOLUTION	X	X	X	X	X	X	X	X	X
ADDRESS	X	X	X	X	X	X	X	X	X
ADVANCED_BREP_SHAPE_REPRESENTATION			X	X				X	
ADVANCED_FACE		X		X		X			X
APPLICATION_CONTEXT	X	X	X	X	X	X	X	X	X
APPLICATION_CONTEXT_ELEMENT	X	X	X	X	X	X	X	X	X
APPLICATION_PROTOCOL_DEFINITION	X	X	X	X	X	X	X	X	X
APPLIED_ACTION_REQUEST_ASSIGNMENT	X	X	X	X	X	X	X	X	X
APPLIED_APPROVAL_ASSIGNMENT	X	X							X
applied_classification_assignment	X	X	X	X	X	X	X	X	X
APPLIED_DATE_AND_TIME_ASSIGNMENT	X	X	X	X	X	X	X	X	X
APPLIED_DOCUMENT_REFERENCE	X	X	X	X	X	X	X	X	X
APPLIED_EXTERNAL_IDENTIFICATION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
APPLIED_GROUP_ASSIGNMENT	X	X	X	X	X	X	X	X	X
APPLIED_IDENTIFICATION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
APPLIED_ORGANIZATION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
APPLIED_PERSON_AND_ORGANIZATION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
APPLIED_PERSON_ASSIGNMENT	X	X	X	X	X	X	X	X	X
APPROVAL	X	X							X
APPROVAL_ASSIGNMENT	X	X							X
APPROVAL_DATE_TIME	X	X							X
APPROVAL_PERSON_ORGANIZATION	X	X							X
APPROVAL_ROLE	X	X							X
APPROVAL_STATUS	X	X							X

Table 5 — Conformance class elements (continued)

AIM Element	Class								
	1	2	3	4	5	6	7	8	9
AXIS1_PLACEMENT	X	X	X	X	X	X	X	X	X
AXIS2_PLACEMENT_2D	X	X	X	X	X	X	X	X	X
AXIS2_PLACEMENT_3D	X	X	X	X	X	X	X	X	X
B_SPLINE_CURVE		X		X		X			X
B_SPLINE_CURVE_WITH_KNOTS	X		X	X				X	
B_SPLINE_SURFACE		X		X		X			X
B_SPLINE_SURFACE_WITH_KNOTS			X	X				X	
BEZIER_CURVE		X	X	X	X		X	X	
BEZIER_SURFACE			X	X				X	
BOUNDED_CURVE		X		X		X			X
BOUNDED_SURFACE_CURVE			X	X				X	
BOUNDED_SURFACE		X		X		X			X
BOUNDED_PCURVE			X	X				X	
BREP_WITH_VOIDS			X	X				X	
CALENDAR_DATE	X	X	X	X	X	X	X	X	X
CARTESIAN_POINT	X	X	X	X	X	X	X	X	X
CARTESIAN_TRANSFORMATION_OPERATOR	X	X	X	X	X	X	X	X	X
CARTESIAN_TRANSFORMATION_OPERATOR_3D	X	X	X	X	X	X	X	X	X
CHARACTERIZED_OBJECT	X	X	X	X	X	X	X	X	X
CIRCLE	X	X	X	X	X	X	X	X	X
CLASS	X	X	X	X	X	X	X	X	X
CLASSIFICATION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
CLASSIFICATION_ROLE	X	X	X	X	X	X	X	X	X
CLOSED_SHELL		X	X	X		X		X	X
COMPOSITE_CURVE			X	X			X	X	
COMPOSITE_CURVE_ON_SURFACE			X	X				X	
COMPOSITE_CURVE_SEGMENT			X	X				X	
COMPOUND_REPRESENTATION_ITEM	X	X	X	X	X	X	X	X	X
CONIC		X	X	X	X		X	X	
CONICAL_SURFACE			X	X				X	

Table 5 — Conformance class elements (continued)

AIM Element	Class								
	1	2	3	4	5	6	7	8	9
CONNECTED_EDGE_SET			X	X				X	
CONNECTED_FACE_SET		X	X	X		X		X	X
CONTEXT_DEPENDENT_UNIT	X	X	X	X	X	X	X	X	X
CONVERSION_BASED_UNIT	X	X	X	X	X	X	X	X	X
COORDINATED_UNIVERSAL_TIME_OFFSET	X	X	X	X	X	X	X	X	X
CURVE		X		X	X	X			X
CURVE_REPLICA		X	X	X	X	X	X	X	X
CYLINDRICAL_SURFACE			X	X				X	
DATE	X	X	X	X	X	X	X	X	X
DATE_AND_TIME	X	X	X	X	X	X	X	X	X
DATE_AND_TIME_ASSIGNMENT	X	X	X	X	X	X	X	X	X
DATE_TIME_ROLE	X	X	X	X	X	X	X	X	X
DEFINITIONAL_REPRESENTATION			X	X				X	
DEGENERATE_PCURVE		X		X		X			X
DEGENERATE_TOROIDAL_SURFACE			X	X				X	
DERIVED_UNIT	X	X	X	X	X	X	X	X	X
DERIVED_UNIT_ELEMENT	X	X	X	X	X	X	X	X	X
DESCRIPTION_ATTRIBUTE	X	X	X	X	X	X	X	X	X
DESCRIPTIVE_REPRESENTATION_ITEM	X	X	X	X	X	X	X	X	X
DIMENSIONAL_EXPONENTS	X	X	X	X	X	X	X	X	X
DIRECTION			X	X				X	
DOCUMENT	X	X	X	X	X	X	X	X	X
DOCUMENT_REFERENCE	X	X	X	X	X	X	X	X	X
DOCUMENT_REPRESENTATION_TYPE	X	X	X	X	X	X	X	X	X
DOCUMENT_TYPE	X	X	X	X	X	X	X	X	X
DOCUMENT_USAGE_CONSTRAINT	X	X	X	X	X	X	X	X	X
EDGE			X	X				X	
EDGE_BASED_WIREFRAME_MODEL		X			X				
EDGE_BASED_WIREFRAME_SHAPE_REPRESENTATION		X			X				

Table 5 — Conformance class elements (continued)

AIM Element	Class								
	1	2	3	4	5	6	7	8	9
EDGE_CURVE		X	X	X	X			X	
EDGE_LOOP			X	X				X	
ELEMENTARY_SURFACE		X		X		X			X
ELLIPSE		X	X	X	X		X	X	
EXECUTED_ACTION	X	X	X	X	X	X	X	X	X
EXTERNAL_IDENTIFICATION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
EXTERNAL_SOURCE	X	X	X	X	X	X	X	X	X
EXTERNAL_SOURCE_RELATIONSHIP	X	X	X	X	X	X	X	X	X
EXTERNALLY_DEFINED_CLASS	X	X	X	X	X	X	X	X	X
EXTERNALLY_DEFINED_GENERAL_PROPERTY	X	X	X	X	X	X	X	X	X
EXTERNALLY_DEFINED_ITEM	X	X	X	X	X	X	X	X	X
EVALUATED_DEGENERATE_PCURVE			X	X				X	
FACE		X		X		X			X
FACE_BASED_SURFACE_MODEL		X		X		X			X
FACE_BOUND			X	X				X	
FACE_OUTER_BOUND			X	X				X	
FACE_SURFACE			X	X				X	
FACETED_BREP			X	X				X	
FOUNDED_ITEM	X	X	X	X	X	X	X	X	X
FUNCTIONALLY_DEFINED_TRANSFORMATION	X	X	X	X	X	X	X	X	X
GENERAL_PROPERTY	X	X	X	X	X	X	X	X	X
GEOMETRIC_CURVE_SET				X			X		
GEOMETRIC_REPRESENTATION_CONTEXT		X	X	X	X		X	X	
GEOMETRIC_REPRESENTATION_ITEM	X	X	X	X	X	X	X	X	X
GEOMETRIC_SET				X			X		
GEOMETRICALLY_BOUNDED_WIREFRAME_SHAPE_ REPRESENTATION				X			X		
GLOBAL_UNCERTAINTY_ASSIGNED_CONTEXT	X	X	X	X	X	X	X	X	X
GLOBAL_UNIT_ASSIGNED_CONTEXT	X	X	X	X	X	X	X	X	X
GROUP	X	X	X	X	X	X	X	X	X

Table 5 — Conformance class elements (continued)

AIM Element	Class								
	1	2	3	4	5	6	7	8	9
GROUP_ASSIGNMENT	X	X	X	X	X	X	X	X	X
GROUP_RELATIONSHIP	X	X	X	X	X	X	X	X	X
HYPERBOLA		X	X	X	X		X	X	
ID_ATTRIBUTE	X	X	X	X	X	X	X	X	X
IDENTIFICATION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
IDENTIFICATION_ASSIGNMENT_RELATIONSHIP	X	X	X	X	X	X	X	X	X
IDENTIFICATION_ROLE	X	X	X	X	X	X	X	X	X
INTERSECTION_CURVE			X	X				X	
ITEM_DEFINED_TRANSFORMATION	X	X	X	X	X	X	X	X	X
KNOWN_SOURCE	X	X	X	X	X	X	X	X	X
LENGTH_UNIT	X	X	X	X	X	X	X	X	X
LIBRARY_APPLIED_VERSION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
LIBRARY_CLASS_VERSION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
LINE		X	X	X	X		X	X	
LOCAL_TIME	X	X	X	X	X	X	X	X	X
LOOP		X	X	X	X		X	X	
MANIFOLD_SOLID_BREP			X	X				X	
MAPPED_ITEM	X	X	X	X	X	X	X	X	X
MEASURE_WITH_UNIT	X	X	X	X	X	X	X	X	X
NAME_ATTRIBUTE	X	X	X	X	X	X	X	X	X
NAMED_UNIT	X	X	X	X	X	X	X	X	X
NON_MANIFOLD_SURFACE_SHAPE_REPRESENTATION		X		X		X			
OBJECT_ROLE	X	X	X	X	X	X	X	X	X
OFFSET_CURVE_3D		X	X	X	X	X	X	X	X
OFFSET_SURFACE		X		X		X			X
OPEN_SHELL		X		X		X			X
ORDINAL_DATE	X	X	X	X	X	X	X	X	X
ORGANIZATION	X	X	X	X	X	X	X	X	X
ORGANIZATION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
ORGANIZATION_ROLE	X	X	X	X	X	X	X	X	X

Table 5 — Conformance class elements (continued)

AIM Element	Class								
	1	2	3	4	5	6	7	8	9
ORGANIZATIONAL_ADDRESS	X	X	X	X	X	X	X	X	X
ORGANIZATIONAL_PROJECT	X	X	X	X	X	X	X	X	X
ORIENTED_CLOSED_SHELL			X	X				X	
ORIENTED_EDGE			X	X				X	
ORIENTED_FACE		X		X		X			X
ORIENTED_OPEN_SHELL		X		X		X			X
ORIENTED_PATH			X	X				X	
ORIENTED_SURFACE		X		X		X			X
PARABOLA		X	X	X	X		X	X	
PARAMETRIC_REPRESENTATION_CONTEXT			X	X				X	
PARAMETERISED_COMPOSITE_CURVE_SEGMENT			X	X				X	
PATH			X	X				X	
PCURVE			X	X				X	
PERSON	X	X	X	X	X	X	X	X	X
PERSON_AND_ORGANIZATION	X	X	X	X	X	X	X	X	X
PERSON_AND_ORGANIZATION_ASSIGNMENT	X	X	X	X	X	X	X	X	X
PERSON_AND_ORGANIZATION_ROLE	X	X	X	X	X	X	X	X	X
PERSON_ASSIGNMENT	X	X	X	X	X	X	X	X	X
PERSON_ROLE	X	X	X	X	X	X	X	X	X
PERSONAL_ADDRESS	X	X	X	X	X	X	X	X	X
PLACEMENT	X	X	X	X	X	X	X	X	X
PLANE		X	X	X		X		X	X
PLANE_ANGLE_UNIT	X	X	X	X	X	X	X	X	X
POINT	X	X	X	X	X	X	X	X	X
POINT_ON_CURVE		X	X	X		X	X	X	
POINT_ON_SURFACE		X		X		X			
POINT_REPLICA		X	X	X	X		X	X	
POLY_LOOP		X	X	X	X		X	X	
POLYLINE		X	X	X	X		X	X	
PRE_DEFINED_ITEM	X	X	X	X	X	X	X	X	X

Table 5 — Conformance class elements (continued)

AIM Element	Class								
	1	2	3	4	5	6	7	8	9
PRODUCT	X	X	X	X	X	X	X	X	X
PRODUCT_CATEGORY	X	X	X	X	X	X	X	X	X
PRODUCT_CATEGORY_RELATIONSHIP	X	X	X	X	X	X	X	X	X
PRODUCT_CONTEXT	X	X	X	X	X	X	X	X	X
PRODUCT_DEFINITION	X	X	X	X	X	X	X	X	X
PRODUCT_DEFINITION_CONTEXT	X	X	X	X	X	X	X	X	X
PRODUCT_DEFINITION_FORMATION	X	X	X	X	X	X	X	X	X
PRODUCT_DEFINITION_RELATIONSHIP	X	X	X	X	X	X	X	X	X
PRODUCT_DEFINITION_SHAPE	X	X	X	X	X	X	X	X	X
PRODUCT_DEFINITION_WITH_ASSOCIATED_DOCUMENTS	X	X	X	X	X	X	X	X	X
PRODUCT_RELATED_PRODUCT_CATEGORY	X	X	X	X	X	X	X	X	X
PROPERTY_DEFINITION	X	X	X	X	X	X	X	X	X
PROPERTY_DEFINITION_RELATIONSHIP	X	X	X	X	X	X	X	X	X
PROPERTY_DEFINITION_REPRESENTATION	X	X	X	X	X	X	X	X	X
QUASI_UNIFORM_CURVE		X	X	X	X		X	X	
QUASI_UNIFORM_SURFACE			X	X				X	
RATIONAL_B_SPLINE_CURVE		X	X	X	X		X	X	
RATIONAL_B_SPLINE_SURFACE			X	X				X	
REPRESENTATION	X	X	X	X	X	X	X	X	X
REPRESENTATION_CONTEXT	X	X	X	X	X	X	X	X	X
REPRESENTATION_ITEM	X	X	X	X	X	X	X	X	X
REPRESENTATION_MAP	X	X	X	X	X	X	X	X	X
REPRESENTATION_RELATIONSHIP	X	X	X	X	X	X	X	X	X
ROLE_ASSOCIATION	X	X	X	X	X	X	X	X	X
SEAM_CURVE			X	X				X	
SHAPE_ASPECT	X	X	X	X	X	X	X	X	X
SHAPE_ASPECT_RELATIONSHIP	X	X	X	X	X	X	X	X	X
SHAPE_DEFINITION_REPRESENTATION	X	X	X	X	X	X	X	X	X
SHAPE_REPRESENTATION	X	X	X	X	X	X	X	X	X
SI_UNIT	X	X	X	X	X	X	X	X	X
SOLID_MODEL	X	X	X	X	X	X	X	X	X

Table 5 — Conformance class elements (concluded)

AIM Element	Class								
	1	2	3	4	5	6	7	8	9
SPHERICAL_SURFACE			X	X				X	
SURFACE		X		X		X			X
SURFACE_CURVE			X	X				X	
SURFACE_OF_LINEAR_EXTRUSION			X	X				X	
SURFACE_OF_REVOLUTION			X	X				X	
SURFACE_REPLICA		X		X		X			X
SWEPT_SURFACE			X	X				X	
THERMODYNAMIC_TEMPERATURE_UNIT	X	X	X	X	X	X	X	X	X
TIME_UNIT	X	X	X	X	X	X	X	X	X
TOPOLOGICAL_REPRESENTATION_ITEM			X	X				X	
TOROIDAL_SURFACE			X	X				X	
TRIMMED_CURVE			X	X			X	X	
UNCERTAINTY_MEASURE_WITH_UNIT	X	X	X	X	X	X	X	X	X
UNIFORM_CURVE		X	X	X	X		X	X	
UNIFORM_SURFACE			X	X				X	
VALUE_REPRESENTATION_ITEM	X	X	X	X	X	X	X	X	X
VECTOR			X	X				X	
VERSIONED_ACTION_REQUEST	X	X	X	X	X	X	X	X	X
VERTEX		X	X	X	X			X	
VERTEX_LOOP			X	X				X	
VERTEX_POINT		X	X	X	X			X	
WEEK_OF_YEAR_AND_DAY_DATE	X	X	X	X	X	X	X	X	X

Annex A
(normative)

AIM EXPRESS expanded listing

```
SCHEMA ship_structures_schema;

 CONSTANT
 dummy_gri : geometric_representation_item := representation_item('') ||
             geometric_representation_item();
 dummy_tri : topological_representation_item := representation_item('')
             || topological_representation_item();
 END_CONSTANT;

 TYPE action_request_item = SELECT
  (action,
   executed_action,
   group,
   product,
   product_definition,
   product_definition_relationship,
   product_definition_shape,
   property_definition);
 END_TYPE; -- action_request_item

 TYPE ahead_or_behind = ENUMERATION OF
  (ahead,
   exact,
   behind);
 END_TYPE; -- ahead_or_behind

 TYPE amount_of_substance_measure = REAL;
 END_TYPE; -- amount_of_substance_measure

 TYPE approval_item = SELECT
  (product_definition_shape,
   product_definition,
   property_definition);
 END_TYPE; -- approval_item

 TYPE area_measure = REAL;
 END_TYPE; -- area_measure

 TYPE attribute_type = SELECT
  (label,
   text);
 END_TYPE; -- attribute_type
```

```
TYPE axis2_placement = SELECT
  (axis2_placement_2d,
   axis2_placement_3d);
END_TYPE; -- axis2_placement

TYPE b_spline_curve_form = ENUMERATION OF
  (polyline_form,
   circular_arc,
   elliptic_arc,
   parabolic_arc,
   hyperbolic_arc,
   unspecified);
END_TYPE; -- b_spline_curve_form

TYPE b_spline_surface_form = ENUMERATION OF
  (plane_surf,
   cylindrical_surf,
   conical_surf,
   spherical_surf,
   toroidal_surf,
   surf_of_revolution,
   ruled_surf,
   generalised_cone,
   quadric_surf,
   surf_of_linear_extrusion,
   unspecified);
END_TYPE; -- b_spline_surface_form

TYPE boolean_operand = SELECT
  (solid_model);
END_TYPE; -- boolean_operand

TYPE celsius_temperature_measure = REAL;
END_TYPE; -- celsius_temperature_measure

TYPE characterized_action_definition = SELECT
  (action,
   action_method);
END_TYPE; -- characterized_action_definition

TYPE characterized_definition = SELECT
  (characterized_object,
   characterized_product_definition,
   shape_definition);
END_TYPE; -- characterized_definition

TYPE characterized_product_definition = SELECT
  (product_definition,
   product_definition_relationship);
END_TYPE; -- characterized_product_definition
```



```
TYPE classification_item = SELECT
  (action,
   action_request_solution,
   compound_representation_item,
   group,
   product,
   product_definition,
   product_definition_with_associated_documents,
   product_definition_relationship,
   product_definition_shape,
   product_related_product_category,
   property_definition,
   property_definition_relationship,
   representation,
   shape_aspect,
   shape_aspect_relationship,
   versioned_action_request);
END_TYPE; -- classification_item

TYPE compound_item_definition = SELECT
  (list_representation_item,
   set_representation_item);
END_TYPE; -- compound_item_definition

TYPE configuration_design_item = SELECT
  (product_definition,
   product_definition_formation);
END_TYPE; -- configuration_design_item

TYPE context_dependent_measure = REAL;
END_TYPE; -- context_dependent_measure

TYPE count_measure = NUMBER;
END_TYPE; -- count_measure

TYPE curve_on_surface = SELECT
  (pcurve,
   surface_curve,
   composite_curve_on_surface);
END_TYPE; -- curve_on_surface

TYPE date_and_time_item = SELECT
  (action,
   action_request_solution,
   product_definition,
   versioned_action_request);
END_TYPE; -- date_and_time_item

TYPE date_time_or_event_occurrence = SELECT
  (date_time_select);
```

```
END_TYPE; -- date_time_or_event_occurrence

TYPE date_time_select = SELECT
  (date,
   local_time,
   date_and_time);
END_TYPE; -- date_time_select

TYPE day_in_month_number = INTEGER;
WHERE
  WR1: ((1 <= SELF) AND (SELF <= 31));
END_TYPE; -- day_in_month_number

TYPE day_in_week_number = INTEGER;
WHERE
  WR1: ((1 <= SELF) AND (SELF <= 7));
END_TYPE; -- day_in_week_number

TYPE day_in_year_number = INTEGER;
WHERE
  WR1: ((1 <= SELF) AND (SELF <= 366));
END_TYPE; -- day_in_year_number

TYPE derived_property_select = SELECT
  (property_definition);
END_TYPE; -- derived_property_select

TYPE description_attribute_select = SELECT
  (action_request_solution,
   application_context,
   approval_role,
   date_time_role,
   external_source,
   organization_role,
   person_and_organization_role,
   person_and_organization,
   person_role,
   property_definition_representation,
   representation);
END_TYPE; -- description_attribute_select

TYPE dimension_count = INTEGER;
WHERE
  WR1: (SELF > 0);
END_TYPE; -- dimension_count

TYPE document_reference_item = SELECT
  (group,
   property_definition);
END_TYPE; -- document_reference_item
```



```
TYPE electric_current_measure = REAL;
END_TYPE; -- electric_current_measure

TYPE external_identification_item = SELECT
  (document,
   document_reference,
   product,
   product_definition,
   product_definition_shape,
   property_definition,
   shape_aspect);
END_TYPE; -- external_identification_item

TYPE founded_item_select = SELECT
  (founded_item,
   representation_item);
END_TYPE; -- founded_item_select

TYPE geometric_set_select = SELECT
  (point,
   curve,
   surface);
END_TYPE; -- geometric_set_select

TYPE group_item = SELECT
  (approval,
   product_definition,
   product_definition_shape,
   product_definition_with_associated_documents,
   product_definition_relationship,
   representation,
   shape_aspect);
END_TYPE; -- group_item

TYPE hour_in_day = INTEGER;
WHERE
  WR1: ((0 <= SELF) AND (SELF < 24));
END_TYPE; -- hour_in_day

TYPE id_attribute_select = SELECT
  (action,
   address,
   product_category,
   property_definition,
   shape_aspect,
   shape_aspect_relationship,
   application_context,
   group,
   organizational_project,
   representation);
```

```
END_TYPE; -- id_attribute_select

TYPE identification_item = SELECT
  (action_request_solution,
   executed_action,
   externally_defined_general_property,
   externally_defined_class,
   group,
   product,
   product_definition,
   product_definition_shape,
   product_definition_relationship,
   product_related_product_category,
   property_definition_relationship,
   property_definition,
   representation,
   shape_aspect,
   shape_aspect_relationship,
   versioned_action_request);
END_TYPE; -- identification_item

TYPE identifier = STRING;
END_TYPE; -- identifier

TYPE knot_type = ENUMERATION OF
  (uniform_knots,
   quasi_uniform_knots,
   piecewise_bezier_knots,
   unspecified);
END_TYPE; -- knot_type

TYPE label = STRING;
END_TYPE; -- label

TYPE length_measure = REAL;
END_TYPE; -- length_measure

TYPE list_of_reversible_topology_item = LIST [0:?] OF
        reversible_topology_item;
END_TYPE; -- list_of_reversible_topology_item

TYPE list_representation_item = LIST [1:?] OF representation_item;
END_TYPE; -- list_representation_item

TYPE luminous_intensity_measure = REAL;
END_TYPE; -- luminous_intensity_measure

TYPE mass_measure = REAL;
END_TYPE; -- mass_measure
```



```
TYPE measure_value = SELECT
  (length_measure,
   mass_measure,
   time_measure,
   electric_current_measure,
   thermodynamic_temperature_measure,
   celsius_temperature_measure,
   amount_of_substance_measure,
   luminous_intensity_measure,
   plane_angle_measure,
   solid_angle_measure,
   area_measure,
   volume_measure,
   ratio_measure,
   parameter_value,
   context_dependent_measure,
   positive_length_measure,
   positive_plane_angle_measure,
   count_measure);
END_TYPE; -- measure_value

TYPE minute_in_hour = INTEGER;
WHERE
  WR1: ((0 <= SELF) AND (SELF <= 59));
END_TYPE; -- minute_in_hour

TYPE month_in_year_number = INTEGER;
WHERE
  WR1: ((1 <= SELF) AND (SELF <= 12));
END_TYPE; -- month_in_year_number

TYPE name_attribute_select = SELECT
  (action_request_solution,
   address,
   derived_unit,
   person_and_organization,
   product_definition,
   property_definition_representation);
END_TYPE; -- name_attribute_select

TYPE organization_item = SELECT
  (externally_defined_class,
   document,
   known_source,
   product_definition,
   property_definition);
END_TYPE; -- organization_item

TYPE parameter_value = REAL;
END_TYPE; -- parameter_value
```

```
TYPE pcurve_or_surface = SELECT
  (pcurve,
   surface);
END_TYPE; -- pcurve_or_surface

TYPE person_and_organization_item = SELECT
  (action_request_solution,
   executed_action,
   document,
   versioned_action_request);
END_TYPE; -- person_and_organization_item

TYPE person_item = SELECT
  (action_request_solution,
   document,
   executed_action,
   versioned_action_request);
END_TYPE; -- person_item

TYPE person_organization_select = SELECT
  (person,
   organization,
   person_and_organization);
END_TYPE; -- person_organization_select

TYPE plane_angle_measure = REAL;
END_TYPE; -- plane_angle_measure

TYPE positive_length_measure = length_measure;
WHERE
  WR1: (SELF > 0);
END_TYPE; -- positive_length_measure

TYPE positive_plane_angle_measure = plane_angle_measure;
WHERE
  WR1: (SELF > 0);
END_TYPE; -- positive_plane_angle_measure

TYPE preferred_surface_curve_representation = ENUMERATION OF
  (curve_3d,
   pcurve_s1,
   pcurve_s2);
END_TYPE; -- preferred_surface_curve_representation

TYPE product_or_formation_or_definition = SELECT
  (product,
   product_definition_formation,
   product_definition);
END_TYPE; -- product_or_formation_or_definition
```



```
TYPE property_or_shape_select = SELECT
  (property_definition,
   shape_definition);
END_TYPE; -- property_or_shape_select

TYPE ratio_measure = REAL;
END_TYPE; -- ratio_measure

TYPE represented_definition = SELECT
  (general_property,
   property_definition,
   property_definition_relationship,
   shape_aspect,
   shape_aspect_relationship);
END_TYPE; -- represented_definition

TYPE reversible_topology = SELECT
  (reversible_topology_item,
   list_of_reversible_topology_item,
   set_of_reversible_topology_item);
END_TYPE; -- reversible_topology

TYPE reversible_topology_item = SELECT
  (edge,
   path,
   face,
   face_bound,
   closed_shell,
   open_shell);
END_TYPE; -- reversible_topology_item

TYPE role_select = SELECT
  (action_request_assignment,
   approval_assignment,
   approval_date_time,
   document_reference,
   group_assignment);
END_TYPE; -- role_select

TYPE second_in_minute = REAL;
WHERE
  WR1: ((0 <= SELF) AND (SELF <= 60));
END_TYPE; -- second_in_minute

TYPE set_of_reversible_topology_item = SET [0:?] OF
          reversible_topology_item;
END_TYPE; -- set_of_reversible_topology_item

TYPE set_representation_item = SET [1:?] OF representation_item;
END_TYPE; -- set_representation_item
```

```
TYPE shape_definition = SELECT
  (product_definition_shape,
   shape_aspect,
   shape_aspect_relationship);
END_TYPE; -- shape_definition

TYPE shell = SELECT
  (open_shell,
   closed_shell);
END_TYPE; -- shell

TYPE si_prefix = ENUMERATION OF
  (exa,
   peta,
   tera,
   giga,
   mega,
   kilo,
   hecto,
   deca,
   deci,
   centi,
   milli,
   micro,
   nano,
   pico,
   femto,
   atto);
END_TYPE; -- si_prefix

TYPE si_unit_name = ENUMERATION OF
  (metre,
   gram,
   second,
   ampere,
   kelvin,
   mole,
   candela,
   radian,
   steradian,
   hertz,
   newton,
   pascal,
   joule,
   watt,
   coulomb,
   volt,
   farad,
   ohm,
   siemens,
```



```
    weber,
    tesla,
    henry,
    degree_celsius,
    lumen,
    lux,
    becquerel,
    gray,
    sievert);
END_TYPE; -- si_unit_name

TYPE solid_angle_measure = REAL;
END_TYPE; -- solid_angle_measure

TYPE source_item = SELECT
  (identifier);
END_TYPE; -- source_item

TYPE supported_item = SELECT
  (action,
   action_method);
END_TYPE; -- supported_item

TYPE surface_boundary = SELECT
  (degenerate_pcurve);
END_TYPE; -- surface_boundary

TYPE surface_model = SELECT
  (face_based_surface_model);
END_TYPE; -- surface_model

TYPE text = STRING;
END_TYPE; -- text

TYPE thermodynamic_temperature_measure = REAL;
END_TYPE; -- thermodynamic_temperature_measure

TYPE time_measure = REAL;
END_TYPE; -- time_measure

TYPE transformation = SELECT
  (item_defined_transformation,
   functionally_defined_transformation);
END_TYPE; -- transformation

TYPE transition_code = ENUMERATION OF
  (discontinuous,
   continuous,
   cont_same_gradient,
   cont_same_gradient_same_curvature);
```



```
END_TYPE; -- transition_code

TYPE trimming_preference = ENUMERATION OF
  (cartesian,
   parameter,
   unspecified);
END_TYPE; -- trimming_preference

TYPE trimming_select = SELECT
  (cartesian_point,
   parameter_value);
END_TYPE; -- trimming_select

TYPE unit = SELECT
  (named_unit,
   derived_unit);
END_TYPE; -- unit

TYPE vector_or_direction = SELECT
  (vector,
   direction);
END_TYPE; -- vector_or_direction

TYPE volume_measure = REAL;
END_TYPE; -- volume_measure

TYPE week_in_year_number = INTEGER;
WHERE
  WR1: ((1 <= SELF) AND (SELF <= 53));
END_TYPE; -- week_in_year_number

TYPE wireframe_model = SELECT
  (edge_based_wireframe_model);
END_TYPE; -- wireframe_model

TYPE year_number = INTEGER;
END_TYPE; -- year_number

ENTITY action;
    name         : label;
    description   : OPTIONAL text;
    chosen_method : action_method;
  DERIVE
    id : identifier := get_id_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1);
END_ENTITY; -- action

ENTITY action_method;
```



```
    name       : label;
    description : OPTIONAL text;
    consequence : text;
    purpose    : text;
END_ENTITY; -- action_method

ENTITY action_request_assignment
  ABSTRACT SUPERTYPE;
    assigned_action_request : versioned_action_request;
  DERIVE
    role : object_role := get_role(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'ROLE_ASSOCIATION.ITEM_WITH_ROLE')) <= 1);
END_ENTITY; -- action_request_assignment

ENTITY action_request_solution;
    method  : action_method;
    request : versioned_action_request;
  DERIVE
    description : text := get_description_value(SELF);
    name       : label := get_name_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
    WR2: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'NAME_ATTRIBUTE.NAMED_ITEM')) <= 1);
END_ENTITY; -- action_request_solution

ENTITY address;
    internal_location      : OPTIONAL label;
    street_number          : OPTIONAL label;
    street                 : OPTIONAL label;
    postal_box             : OPTIONAL label;
    town                   : OPTIONAL label;
    region                 : OPTIONAL label;
    postal_code            : OPTIONAL label;
    country                : OPTIONAL label;
    facsimile_number       : OPTIONAL label;
    telephone_number       : OPTIONAL label;
    electronic_mail_address : OPTIONAL label;
    telex_number           : OPTIONAL label;
  DERIVE
    name : label := get_name_value(SELF);
    url  : identifier := get_id_value(SELF);
  WHERE
    WR1: (EXISTS(internal_location) OR EXISTS(street_number) OR EXISTS(
          street) OR EXISTS(postal_box) OR EXISTS(town) OR EXISTS(
          region) OR EXISTS(postal_code) OR EXISTS(country) OR EXISTS(
          facsimile_number) OR EXISTS(telephone_number) OR EXISTS(
```

```
        electronic_mail_address) OR EXISTS(telex_number));
 END_ENTITY; -- address

 ENTITY advanced_brep_shape_representation
  SUBTYPE OF (shape_representation);
  WHERE
   WR1: (SIZEOF(QUERY ( it <* SELF.items | (NOT (SIZEOF([
        'SHIP_STRUCTURES_SCHEMA.MANIFOLD_SOLID_BREP',
        'SHIP_STRUCTURES_SCHEMA.FACETED_BREP',
        'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM',
        'SHIP_STRUCTURES_SCHEMA.AXIS2_PLACEMENT_3D'] * TYPEOF(it)) =
         1)) )) = 0);
   WR2: (SIZEOF(QUERY ( it <* SELF.items | (SIZEOF([
        'SHIP_STRUCTURES_SCHEMA.MANIFOLD_SOLID_BREP',
        'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM'] * TYPEOF(it)) = 1) ))
        > 0);
   WR3: (SIZEOF(QUERY ( msb <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.MANIFOLD_SOLID_BREP' IN TYPEOF(it)) )
         | (NOT (SIZEOF(QUERY ( csh <* msb_shells(msb) | (NOT (
        SIZEOF(QUERY ( fcs <* csh\connected_face_set.cfs_faces | (
        NOT ('SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(fcs))) ))
        = 0)) )) = 0)) )) = 0);
   WR4: (SIZEOF(QUERY ( msb <* QUERY ( it <* items | (
        'SHIP_STRUCTURES_SCHEMA.MANIFOLD_SOLID_BREP' IN TYPEOF(it)) )
         | ('SHIP_STRUCTURES_SCHEMA.ORIENTED_CLOSED_SHELL' IN
        TYPEOF(msb\manifold_solid_brep.outer)) )) = 0);
   WR5: (SIZEOF(QUERY ( brv <* QUERY ( it <* items | (
        'SHIP_STRUCTURES_SCHEMA.BREP_WITH_VOIDS' IN TYPEOF(it)) ) |
        (NOT (SIZEOF(QUERY ( csh <* brv\brep_with_voids.voids | csh\
        oriented_closed_shell.orientation )) = 0)) )) = 0);
   WR6: (SIZEOF(QUERY ( mi <* QUERY ( it <* items | (
        'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM' IN TYPEOF(it)) ) | (
        NOT (
        'SHIP_STRUCTURES_SCHEMA.ADVANCED_BREP_SHAPE_REPRESENTATION'
        IN TYPEOF(mi\mapped_item.mapping_source.
        mapped_representation))) )) = 0);
 END_ENTITY; -- advanced_brep_shape_representation

 ENTITY advanced_face
  SUBTYPE OF (face_surface);
  WHERE
   WR1 : (SIZEOF(['SHIP_STRUCTURES_SCHEMA.ELEMENTARY_SURFACE',
         'SHIP_STRUCTURES_SCHEMA.B_SPLINE_SURFACE',
         'SHIP_STRUCTURES_SCHEMA.SWEPT_SURFACE'] * TYPEOF(
         face_geometry)) = 1);
   WR2 : (SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* bounds | (
         'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
          | (NOT (SIZEOF(QUERY ( oe <* elp_fbnds.bound\path.
         edge_list | (NOT ('SHIP_STRUCTURES_SCHEMA.EDGE_CURVE' IN
         TYPEOF(oe\oriented_edge.edge_element))) )) = 0)) )) = 0);
```



```
WR3 : (SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* bounds | (
      'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
       | (NOT (SIZEOF(QUERY ( oe <* elp_fbnds.bound\path.
      edge_list | (NOT (SIZEOF(['SHIP_STRUCTURES_SCHEMA.LINE',
      'SHIP_STRUCTURES_SCHEMA.CONIC',
      'SHIP_STRUCTURES_SCHEMA.POLYLINE',
      'SHIP_STRUCTURES_SCHEMA.SURFACE_CURVE',
      'SHIP_STRUCTURES_SCHEMA.B_SPLINE_CURVE'] * TYPEOF(oe.
      edge_element\edge_curve.edge_geometry)) = 1)) )) = 0)) )) =
       0);
WR4 : (SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* bounds | (
      'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
       | (NOT (SIZEOF(QUERY ( oe <* elp_fbnds.bound\path.
      edge_list | (NOT (('SHIP_STRUCTURES_SCHEMA.VERTEX_POINT' IN
       TYPEOF(oe\edge.edge_start)) AND (
      'SHIP_STRUCTURES_SCHEMA.CARTESIAN_POINT' IN TYPEOF(oe\edge.
      edge_start\vertex_point.vertex_geometry)) AND (
      'SHIP_STRUCTURES_SCHEMA.VERTEX_POINT' IN TYPEOF(oe\edge.
      edge_end)) AND ('SHIP_STRUCTURES_SCHEMA.CARTESIAN_POINT' IN
       TYPEOF(oe\edge.edge_end\vertex_point.vertex_geometry)))) ))
      = 0)) )) = 0);
WR5 : (SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* bounds | (
      'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
       | ('SHIP_STRUCTURES_SCHEMA.ORIENTED_PATH' IN TYPEOF(
      elp_fbnds.bound)) )) = 0);
WR6 : ((NOT ('SHIP_STRUCTURES_SCHEMA.SWEPT_SURFACE' IN TYPEOF(
      face_geometry))) OR (SIZEOF(['SHIP_STRUCTURES_SCHEMA.LINE',
      'SHIP_STRUCTURES_SCHEMA.CONIC',
      'SHIP_STRUCTURES_SCHEMA.POLYLINE',
      'SHIP_STRUCTURES_SCHEMA.B_SPLINE_CURVE'] * TYPEOF(
      face_geometry\swept_surface.swept_curve)) = 1));
wr7 : (SIZEOF(QUERY ( vlp_fbnds <* QUERY ( bnds <* bounds | (
      'SHIP_STRUCTURES_SCHEMA.VERTEX_LOOP' IN TYPEOF(bnds.bound)) )
       | (NOT (('SHIP_STRUCTURES_SCHEMA.VERTEX_POINT' IN TYPEOF(
      vlp_fbnds\face_bound.bound\vertex_loop.loop_vertex)) AND (
      'SHIP_STRUCTURES_SCHEMA.CARTESIAN_POINT' IN TYPEOF(
      vlp_fbnds\face_bound.bound\vertex_loop.loop_vertex\
      vertex_point.vertex_geometry)))) )) = 0);
wr8 : (SIZEOF(QUERY ( bnd <* bounds | (NOT (SIZEOF([
      'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP',
      'SHIP_STRUCTURES_SCHEMA.VERTEX_LOOP'] * TYPEOF(bnd.bound))
      = 1)) )) = 0);
wr9 : (SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* bounds | (
      'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
       | (NOT (SIZEOF(QUERY ( oe <* elp_fbnds.bound\path.
      edge_list | (('SHIP_STRUCTURES_SCHEMA.SURFACE_CURVE' IN
      TYPEOF(oe\oriented_edge.edge_element\edge_curve.
      edge_geometry)) AND (NOT (SIZEOF(QUERY ( sc_ag <* oe.
      edge_element\edge_curve.edge_geometry\surface_curve.
      associated_geometry | (NOT ('SHIP_STRUCTURES_SCHEMA.PCURVE'
```

```
        IN TYPEOF(sc_ag))) )) = 0))) )) = 0)) )) = 0);
  WR10: (((NOT ('SHIP_STRUCTURES_SCHEMA.SWEPT_SURFACE' IN TYPEOF(
        face_geometry))) OR (NOT ('SHIP_STRUCTURES_SCHEMA.POLYLINE'
        IN TYPEOF(face_geometry\swept_surface.swept_curve))) OR (
        SIZEOF(face_geometry\swept_surface.swept_curve\polyline.
        points) >= 3)) AND (SIZEOF(QUERY ( elp_fbnds <*
        QUERY ( bnds <* bounds | (
        'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
        | (NOT (SIZEOF(QUERY ( oe <* elp_fbnds.bound\path.
        edge_list | (('SHIP_STRUCTURES_SCHEMA.POLYLINE' IN TYPEOF(
        oe\oriented_edge.edge_element\edge_curve.edge_geometry))
        AND (NOT (SIZEOF(oe\oriented_edge.edge_element\edge_curve.
        edge_geometry\polyline.points) >= 3))) )) = 0)) )) = 0));
END_ENTITY; -- advanced_face

ENTITY application_context;
  application : label;
 DERIVE
  description : text := get_description_value(SELF);
  id          : identifier := get_id_value(SELF);
 INVERSE
  context_elements : SET [1:?] OF application_context_element FOR
                     frame_of_reference;
 WHERE
  WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
       'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
  WR2: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
       'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1);
END_ENTITY; -- application_context

ENTITY application_context_element
 SUPERTYPE OF (ONEOF (product_context,product_definition_context));
  name              : label;
  frame_of_reference : application_context;
END_ENTITY; -- application_context_element

ENTITY application_protocol_definition;
  status                              : label;
  application_interpreted_model_schema_name : label;
  application_protocol_year           : year_number;
  application                         : application_context;
END_ENTITY; -- application_protocol_definition

ENTITY applied_action_request_assignment
 SUBTYPE OF (action_request_assignment);
  items : SET [1:?] OF action_request_item;
END_ENTITY; -- applied_action_request_assignment

ENTITY applied_approval_assignment
 SUBTYPE OF (approval_assignment);
```



```
    items : SET [1:?] OF approval_item;
   WHERE
    WR1: ((NOT (SELF\approval_assignment.role.name =
          'proposed alternative')) OR (SIZEOF(QUERY ( app <* USEDIN(

SELF\approval_assignment.assigned_approval,'SHIP_STRUCTURES_SCHEMA.APPROVAL_ASSI
GNMENT.ASSIGNED_APPROVAL')
           | (('SHIP_STRUCTURES_SCHEMA.APPLIED_APPROVAL_ASSIGNMENT' IN
           TYPEOF(app)) AND (app\approval_assignment.role.name =
          'subject')) )) = 1));
 END_ENTITY; -- applied_approval_assignment

 ENTITY applied_classification_assignment
  SUBTYPE OF (classification_assignment);
    items : SET [1:?] OF classification_item;
 END_ENTITY; -- applied_classification_assignment

 ENTITY applied_date_and_time_assignment
  SUBTYPE OF (date_and_time_assignment);
    items : SET [1:?] OF date_and_time_item;
 END_ENTITY; -- applied_date_and_time_assignment

 ENTITY applied_document_reference
  SUBTYPE OF (document_reference);
    items : SET [1:?] OF document_reference_item;
 END_ENTITY; -- applied_document_reference

 ENTITY applied_external_identification_assignment
  SUBTYPE OF (external_identification_assignment);
    items : SET [1:?] OF external_identification_item;
 END_ENTITY; -- applied_external_identification_assignment

 ENTITY applied_group_assignment
  SUBTYPE OF (group_assignment);
    items : SET [1:?] OF group_item;
 END_ENTITY; -- applied_group_assignment

 ENTITY applied_identification_assignment
  SUBTYPE OF (identification_assignment);
    items : SET [1:?] OF identification_item;
 END_ENTITY; -- applied_identification_assignment

 ENTITY applied_organization_assignment
  SUBTYPE OF (organization_assignment);
    items : SET [1:?] OF organization_item;
 END_ENTITY; -- applied_organization_assignment

 ENTITY applied_person_and_organization_assignment
  SUBTYPE OF (person_and_organization_assignment);
    items : SET [1:?] OF person_and_organization_item;
```

```
END_ENTITY; -- applied_person_and_organization_assignment

ENTITY applied_person_assignment
  SUBTYPE OF (person_assignment);
    items : SET [1:?] OF person_item;
END_ENTITY; -- applied_person_assignment

ENTITY approval;
    status : approval_status;
    level  : label;
END_ENTITY; -- approval

ENTITY approval_assignment
  ABSTRACT SUPERTYPE;
    assigned_approval : approval;
  DERIVE
    role : object_role := get_role(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'ROLE_ASSOCIATION.ITEM_WITH_ROLE')) <= 1);
END_ENTITY; -- approval_assignment

ENTITY approval_date_time;
    date_time      : date_time_select;
    dated_approval : approval;
  DERIVE
    role : object_role := get_role(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'ROLE_ASSOCIATION.ITEM_WITH_ROLE')) <= 1);
END_ENTITY; -- approval_date_time

ENTITY approval_person_organization;
    person_organization : person_organization_select;
    authorized_approval : approval;
    role                : approval_role;
END_ENTITY; -- approval_person_organization

ENTITY approval_role;
    role : label;
  DERIVE
    description : text := get_description_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
END_ENTITY; -- approval_role

ENTITY approval_status;
    name : label;
END_ENTITY; -- approval_status
```



```
ENTITY axis1_placement
  SUBTYPE OF (placement);
    axis : OPTIONAL direction;
  DERIVE
    z : direction := NVL(normalise(axis),dummy_gri || direction([0,0,1]));
  WHERE
    WR1: (SELF\geometric_representation_item.dim = 3);
END_ENTITY; -- axis1_placement

ENTITY axis2_placement_2d
  SUBTYPE OF (placement);
    ref_direction : OPTIONAL direction;
  DERIVE
    p : LIST [2:2] OF direction := build_2axes(ref_direction);
  WHERE
    WR1: (SELF\geometric_representation_item.dim = 2);
END_ENTITY; -- axis2_placement_2d

ENTITY axis2_placement_3d
  SUBTYPE OF (placement);
    axis         : OPTIONAL direction;
    ref_direction : OPTIONAL direction;
  DERIVE
    p : LIST [3:3] OF direction := build_axes(axis,ref_direction);
  WHERE
    WR1: (SELF\placement.location.dim = 3);
    WR2: ((NOT EXISTS(axis)) OR (axis.dim = 3));
    WR3: ((NOT EXISTS(ref_direction)) OR (ref_direction.dim = 3));
    WR4: ((NOT EXISTS(axis)) OR (NOT EXISTS(ref_direction)) OR (
           cross_product(axis,ref_direction).magnitude > 0));
END_ENTITY; -- axis2_placement_3d

ENTITY b_spline_curve
  SUPERTYPE OF (ONEOF (uniform_curve,b_spline_curve_with_knots,
     quasi_uniform_curve,bezier_curve) ANDOR rational_b_spline_curve)
  SUBTYPE OF (bounded_curve);
    degree             : INTEGER;
    control_points_list : LIST [2:?] OF cartesian_point;
    curve_form         : b_spline_curve_form;
    closed_curve       : LOGICAL;
    self_intersect     : LOGICAL;
  DERIVE
    upper_index_on_control_points : INTEGER := SIZEOF(
                                control_points_list) - 1;
    control_points              : ARRAY [0:
                                upper_index_on_control_points] OF
                                 cartesian_point := list_to_array(
                                control_points_list,0,
                                upper_index_on_control_points);
  WHERE
```



```
    WR1: (('SHIP_STRUCTURES_SCHEMA.UNIFORM_CURVE' IN TYPEOF(SELF)) OR (
         'SHIP_STRUCTURES_SCHEMA.QUASI_UNIFORM_CURVE' IN TYPEOF(SELF))
         OR ('SHIP_STRUCTURES_SCHEMA.BEZIER_CURVE' IN TYPEOF(SELF))
         OR ('SHIP_STRUCTURES_SCHEMA.B_SPLINE_CURVE_WITH_KNOTS' IN
         TYPEOF(SELF)));
END_ENTITY; -- b_spline_curve

ENTITY b_spline_curve_with_knots
  SUBTYPE OF (b_spline_curve);
    knot_multiplicities : LIST [2:?] OF INTEGER;
    knots               : LIST [2:?] OF parameter_value;
    knot_spec           : knot_type;
  DERIVE
    upper_index_on_knots : INTEGER := SIZEOF(knots);
  WHERE
    WR1: constraints_param_b_spline(degree,upper_index_on_knots,
         upper_index_on_control_points,knot_multiplicities,knots);
    WR2: (SIZEOF(knot_multiplicities) = upper_index_on_knots);
END_ENTITY; -- b_spline_curve_with_knots

ENTITY b_spline_surface
  SUPERTYPE OF (ONEOF (b_spline_surface_with_knots,uniform_surface,
     quasi_uniform_surface,bezier_surface) ANDOR
     rational_b_spline_surface)
  SUBTYPE OF (bounded_surface);
    u_degree           : INTEGER;
    v_degree           : INTEGER;
    control_points_list : LIST [2:?] OF LIST [2:?] OF cartesian_point;
    surface_form       : b_spline_surface_form;
    u_closed           : LOGICAL;
    v_closed           : LOGICAL;
    self_intersect     : LOGICAL;
  DERIVE
    u_upper       : INTEGER := SIZEOF(control_points_list) - 1;
    v_upper       : INTEGER := SIZEOF(control_points_list[1]) - 1;
    control_points : ARRAY [0:u_upper] OF ARRAY [0:v_upper] OF
                     cartesian_point := make_array_of_array(
                    control_points_list,0,u_upper,0,v_upper);
  WHERE
    WR1: (('SHIP_STRUCTURES_SCHEMA.UNIFORM_SURFACE' IN TYPEOF(SELF)) OR
         ('SHIP_STRUCTURES_SCHEMA.QUASI_UNIFORM_SURFACE' IN TYPEOF(
         SELF)) OR ('SHIP_STRUCTURES_SCHEMA.BEZIER_SURFACE' IN
         TYPEOF(SELF)) OR (
         'SHIP_STRUCTURES_SCHEMA.B_SPLINE_SURFACE_WITH_KNOTS' IN
         TYPEOF(SELF)));
END_ENTITY; -- b_spline_surface

ENTITY b_spline_surface_with_knots
  SUBTYPE OF (b_spline_surface);
    u_multiplicities : LIST [2:?] OF INTEGER;
```



```
    v_multiplicities : LIST [2:?] OF INTEGER;
    u_knots          : LIST [2:?] OF parameter_value;
    v_knots          : LIST [2:?] OF parameter_value;
    knot_spec        : knot_type;
  DERIVE
    knot_u_upper : INTEGER := SIZEOF(u_knots);
    knot_v_upper : INTEGER := SIZEOF(v_knots);
  WHERE
    WR1: constraints_param_b_spline(SELF\b_spline_surface.u_degree,
         knot_u_upper,SELF\b_spline_surface.u_upper,u_multiplicities,
         u_knots);
    WR2: constraints_param_b_spline(SELF\b_spline_surface.v_degree,
         knot_v_upper,SELF\b_spline_surface.v_upper,v_multiplicities,
         v_knots);
    WR3: (SIZEOF(u_multiplicities) = knot_u_upper);
    WR4: (SIZEOF(v_multiplicities) = knot_v_upper);
END_ENTITY; -- b_spline_surface_with_knots

ENTITY bezier_curve
  SUBTYPE OF (b_spline_curve);
END_ENTITY; -- bezier_curve

ENTITY bezier_surface
  SUBTYPE OF (b_spline_surface);
END_ENTITY; -- bezier_surface

ENTITY bounded_curve
  SUPERTYPE OF (ONEOF (polyline,b_spline_curve,trimmed_curve,
     bounded_pcurve,bounded_surface_curve,composite_curve))
  SUBTYPE OF (curve);
END_ENTITY; -- bounded_curve

ENTITY bounded_pcurve
  SUBTYPE OF (pcurve, bounded_curve);
  WHERE
    WR1: ('SHIP_STRUCTURES_SCHEMA.BOUNDED_CURVE' IN TYPEOF(SELF\pcurve.
         reference_to_curve.items[1]));
END_ENTITY; -- bounded_pcurve

ENTITY bounded_surface
  SUPERTYPE OF (b_spline_surface)
  SUBTYPE OF (surface);
END_ENTITY; -- bounded_surface

ENTITY bounded_surface_curve
  SUBTYPE OF (surface_curve, bounded_curve);
  WHERE
    WR1: ('SHIP_STRUCTURES_SCHEMA.BOUNDED_CURVE' IN TYPEOF(SELF\
         surface_curve.curve_3d));
END_ENTITY; -- bounded_surface_curve
```



```
ENTITY brep_with_voids
  SUBTYPE OF (manifold_solid_brep);
    voids : SET [1:?] OF oriented_closed_shell;
END_ENTITY; -- brep_with_voids

ENTITY calendar_date
  SUBTYPE OF (date);
    day_component   : day_in_month_number;
    month_component : month_in_year_number;
  WHERE
    WR1: valid_calendar_date(SELF);
END_ENTITY; -- calendar_date

ENTITY cartesian_point
  SUBTYPE OF (point);
    coordinates : LIST [1:3] OF length_measure;
END_ENTITY; -- cartesian_point

ENTITY cartesian_transformation_operator
  SUPERTYPE OF (cartesian_transformation_operator_3d)
  SUBTYPE OF (geometric_representation_item,
      functionally_defined_transformation);
    axis1       : OPTIONAL direction;
    axis2       : OPTIONAL direction;
    local_origin : cartesian_point;
    scale       : OPTIONAL REAL;
  DERIVE
    scl : REAL := NVL(scale,1);
  WHERE
    WR1: (scl > 0);
END_ENTITY; -- cartesian_transformation_operator

ENTITY cartesian_transformation_operator_3d
  SUBTYPE OF (cartesian_transformation_operator);
    axis3 : OPTIONAL direction;
  DERIVE
    u : LIST [3:3] OF direction := base_axis(3,SELF\
          cartesian_transformation_operator.axis1,SELF\
          cartesian_transformation_operator.axis2,axis3);
  WHERE
    WR1: (SELF\geometric_representation_item.dim = 3);
END_ENTITY; -- cartesian_transformation_operator_3d

ENTITY characterized_object;
    name        : label;
    description : OPTIONAL text;
END_ENTITY; -- characterized_object

ENTITY circle
  SUBTYPE OF (conic);
```



```
   radius : positive_length_measure;
END_ENTITY; -- circle

ENTITY class
  SUBTYPE OF (group);
  WHERE
    WR1: (SIZEOF(QUERY ( oa <* USEDIN(SELF,
          'SHIP_STRUCTURES_SCHEMA.GROUP_ASSIGNMENT.ASSIGNED_GROUP') |
          (NOT ('SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT' IN
          TYPEOF(oa))) )) = 0);
END_ENTITY; -- class

ENTITY classification_assignment
  ABSTRACT SUPERTYPE;
    assigned_class : group;
    role           : classification_role;
END_ENTITY; -- classification_assignment

ENTITY classification_role;
    name        : label;
    description : OPTIONAL text;
END_ENTITY; -- classification_role

ENTITY closed_shell
  SUBTYPE OF (connected_face_set);
END_ENTITY; -- closed_shell

ENTITY composite_curve
  SUBTYPE OF (bounded_curve);
    segments      : LIST [1:?] OF composite_curve_segment;
    self_intersect : LOGICAL;
  DERIVE
    n_segments   : INTEGER := SIZEOF(segments);
    closed_curve : LOGICAL := segments[n_segments].transition <>
                   discontinuous;
  WHERE
    WR1: (((NOT closed_curve) AND (SIZEOF(QUERY ( temp <* segments | (
          temp.transition = discontinuous) )) = 1)) OR (closed_curve
          AND (SIZEOF(QUERY ( temp <* segments | (temp.transition =
          discontinuous) )) = 0)));
END_ENTITY; -- composite_curve

ENTITY composite_curve_on_surface
  SUBTYPE OF (composite_curve);
  DERIVE
    basis_surface : SET [0:2] OF surface := get_basis_surface(SELF);
  WHERE
    WR1: (SIZEOF(basis_surface) > 0);
    WR2: constraints_composite_curve_on_surface(SELF);
END_ENTITY; -- composite_curve_on_surface
```



```
ENTITY composite_curve_segment
  SUBTYPE OF (founded_item);
    transition   : transition_code;
    same_sense   : BOOLEAN;
    parent_curve : curve;
  INVERSE
    using_curves : BAG [1:?] OF composite_curve FOR segments;
  WHERE
    WR1: ('SHIP_STRUCTURES_SCHEMA.BOUNDED_CURVE' IN TYPEOF(parent_curve));
END_ENTITY; -- composite_curve_segment

ENTITY compound_representation_item
  SUBTYPE OF (representation_item);
    item_element : compound_item_definition;
END_ENTITY; -- compound_representation_item

ENTITY conic
  SUPERTYPE OF (ONEOF (circle,ellipse,hyperbola,parabola))
  SUBTYPE OF (curve);
    position : axis2_placement;
END_ENTITY; -- conic

ENTITY conical_surface
  SUBTYPE OF (elementary_surface);
    radius     : length_measure;
    semi_angle : plane_angle_measure;
  WHERE
    WR1: (radius >= 0);
END_ENTITY; -- conical_surface

ENTITY connected_edge_set
  SUBTYPE OF (topological_representation_item);
    ces_edges : SET [1:?] OF edge;
END_ENTITY; -- connected_edge_set

ENTITY connected_face_set
  SUPERTYPE OF (ONEOF (closed_shell,open_shell))
  SUBTYPE OF (topological_representation_item);
    cfs_faces : SET [1:?] OF face;
END_ENTITY; -- connected_face_set

ENTITY context_dependent_unit
  SUBTYPE OF (named_unit);
    name : label;
END_ENTITY; -- context_dependent_unit

ENTITY conversion_based_unit
  SUBTYPE OF (named_unit);
    name              : label;
    conversion_factor : measure_with_unit;
```



```
END_ENTITY; -- conversion_based_unit

ENTITY coordinated_universal_time_offset;
   hour_offset   : INTEGER;
   minute_offset : OPTIONAL INTEGER;
   sense         : ahead_or_behind;
  DERIVE
   actual_minute_offset : INTEGER := NVL(minute_offset,0);
  WHERE
   WR1: ((0 <= hour_offset) AND (hour_offset < 24));
   WR2: ((0 <= actual_minute_offset) AND (actual_minute_offset <= 59));
   WR3: (NOT (((hour_offset <> 0) OR (actual_minute_offset <> 0)) AND (
         sense = exact)));
END_ENTITY; -- coordinated_universal_time_offset

ENTITY curve
  SUPERTYPE OF (ONEOF (line,conic,pcurve,surface_curve,offset_curve_3d,
     curve_replica))
  SUBTYPE OF (geometric_representation_item);
END_ENTITY; -- curve

ENTITY curve_replica
  SUBTYPE OF (curve);
   parent_curve   : curve;
   transformation : cartesian_transformation_operator;
  WHERE
   WR1: (transformation.dim = parent_curve.dim);
   WR2: acyclic_curve_replica(SELF,parent_curve);
END_ENTITY; -- curve_replica

ENTITY cylindrical_surface
  SUBTYPE OF (elementary_surface);
   radius : positive_length_measure;
END_ENTITY; -- cylindrical_surface

ENTITY date
  SUPERTYPE OF (ONEOF (calendar_date,ordinal_date,
     week_of_year_and_day_date));
   year_component : year_number;
END_ENTITY; -- date

ENTITY date_and_time;
   date_component : date;
   time_component : local_time;
END_ENTITY; -- date_and_time

ENTITY date_and_time_assignment
  ABSTRACT SUPERTYPE;
   assigned_date_and_time : date_and_time;
   role                   : date_time_role;
```



```
END_ENTITY; -- date_and_time_assignment

ENTITY date_time_role;
   name : label;
  DERIVE
   description : text := get_description_value(SELF);
  WHERE
   WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
        'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
END_ENTITY; -- date_time_role

ENTITY definitional_representation
  SUBTYPE OF (representation);
  WHERE
   WR1: ('SHIP_STRUCTURES_SCHEMA.PARAMETRIC_REPRESENTATION_CONTEXT' IN
        TYPEOF(SELF\representation.context_of_items));
END_ENTITY; -- definitional_representation

ENTITY degenerate_pcurve
  SUBTYPE OF (point);
   basis_surface      : surface;
   reference_to_curve : definitional_representation;
  WHERE
   WR1: (SIZEOF(reference_to_curve\representation.items) = 1);
   WR2: ('SHIP_STRUCTURES_SCHEMA.CURVE' IN TYPEOF(reference_to_curve\
        representation.items[1]));
   WR3: (reference_to_curve\representation.items[1]\
        geometric_representation_item.dim = 2);
END_ENTITY; -- degenerate_pcurve

ENTITY degenerate_toroidal_surface
  SUBTYPE OF (toroidal_surface);
   select_outer : BOOLEAN;
  WHERE
   WR1: (major_radius < minor_radius);
END_ENTITY; -- degenerate_toroidal_surface

ENTITY derived_unit;
   elements : SET [1:?] OF derived_unit_element;
  DERIVE
   name : label := get_name_value(SELF);
  WHERE
   WR1: ((SIZEOF(elements) > 1) OR ((SIZEOF(elements) = 1) AND (
        elements[1].exponent <> 1)));
   WR2: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
        'NAME_ATTRIBUTE.NAMED_ITEM')) <= 1);
END_ENTITY; -- derived_unit

ENTITY derived_unit_element;
   unit     : named_unit;
```



```
   exponent : REAL;
END_ENTITY; -- derived_unit_element

ENTITY description_attribute;
   attribute_value : text;
   described_item  : description_attribute_select;
END_ENTITY; -- description_attribute

ENTITY descriptive_representation_item
  SUBTYPE OF (representation_item);
   description : text;
END_ENTITY; -- descriptive_representation_item

ENTITY dimensional_exponents;
   length_exponent                : REAL;
   mass_exponent                  : REAL;
   time_exponent                  : REAL;
   electric_current_exponent      : REAL;
   thermodynamic_temperature_exponent : REAL;
   amount_of_substance_exponent   : REAL;
   luminous_intensity_exponent    : REAL;
END_ENTITY; -- dimensional_exponents

ENTITY direction
  SUBTYPE OF (geometric_representation_item);
   direction_ratios : LIST [2:3] OF REAL;
  WHERE
   WR1: (SIZEOF(QUERY ( tmp <* direction_ratios | (tmp <> 0) )) > 0);
END_ENTITY; -- direction

ENTITY document;
   id          : identifier;
   name        : label;
   description : OPTIONAL text;
   kind        : document_type;
  INVERSE
   representation_types : SET [0:?] OF document_representation_type FOR
                          represented_document;
END_ENTITY; -- document

ENTITY document_reference
  ABSTRACT SUPERTYPE;
   assigned_document : document;
   source            : label;
  DERIVE
   role : object_role := get_role(SELF);
  WHERE
   WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
         'ROLE_ASSOCIATION.ITEM_WITH_ROLE')) <= 1);
END_ENTITY; -- document_reference
```

```
ENTITY document_representation_type;
   name                : label;
   represented_document : document;
END_ENTITY; -- document_representation_type

ENTITY document_type;
   product_data_type : label;
END_ENTITY; -- document_type

ENTITY document_usage_constraint;
   source              : document;
   subject_element     : label;
   subject_element_value : text;
END_ENTITY; -- document_usage_constraint

ENTITY edge
 SUPERTYPE OF (ONEOF (edge_curve,oriented_edge))
 SUBTYPE OF (topological_representation_item);
  edge_start : vertex;
  edge_end   : vertex;
END_ENTITY; -- edge

ENTITY edge_based_wireframe_model
 SUBTYPE OF (geometric_representation_item);
  ebwm_boundary : SET [1:?] OF connected_edge_set;
END_ENTITY; -- edge_based_wireframe_model

ENTITY edge_based_wireframe_shape_representation
 SUBTYPE OF (shape_representation);
 WHERE
  WR1: (SIZEOF(QUERY ( it <* SELF.items | (NOT (SIZEOF([
       'SHIP_STRUCTURES_SCHEMA.EDGE_BASED_WIREFRAME_MODEL',
       'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM',
       'SHIP_STRUCTURES_SCHEMA.AXIS2_PLACEMENT_3D'] * TYPEOF(it)) =
        1)) )) = 0);
  WR2: (SIZEOF(QUERY ( it <* SELF.items | (SIZEOF([
       'SHIP_STRUCTURES_SCHEMA.EDGE_BASED_WIREFRAME_MODEL',
       'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM'] * TYPEOF(it)) = 1) ))
       >= 1);
  WR3: (SIZEOF(QUERY ( ebwm <* QUERY ( it <* SELF.items | (
       'SHIP_STRUCTURES_SCHEMA.EDGE_BASED_WIREFRAME_MODEL' IN
       TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( eb <* ebwm\
       edge_based_wireframe_model.ebwm_boundary | (NOT (SIZEOF(
       QUERY ( edges <* eb.ces_edges | (NOT (
       'SHIP_STRUCTURES_SCHEMA.EDGE_CURVE' IN TYPEOF(edges))) )) =
       0)) )) = 0)) )) = 0);
  WR4: (SIZEOF(QUERY ( ebwm <* QUERY ( it <* SELF.items | (
       'SHIP_STRUCTURES_SCHEMA.EDGE_BASED_WIREFRAME_MODEL' IN
       TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( eb <* ebwm\
       edge_based_wireframe_model.ebwm_boundary | (NOT (SIZEOF(
```



```
        QUERY ( pline_edges <* QUERY ( edges <* eb.ces_edges | (
        'SHIP_STRUCTURES_SCHEMA.POLYLINE' IN TYPEOF(edges\edge_curve
        .edge_geometry)) ) | (NOT (SIZEOF(pline_edges\edge_curve.
        edge_geometry\polyline.points) > 2)) )) = 0)) )) = 0)) )) =
        0);
    WR5: (SIZEOF(QUERY ( ebwm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.EDGE_BASED_WIREFRAME_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( eb <* ebwm\
        edge_based_wireframe_model.ebwm_boundary | (NOT (SIZEOF(
        QUERY ( edges <* eb.ces_edges | (NOT ((
        'SHIP_STRUCTURES_SCHEMA.VERTEX_POINT' IN TYPEOF(edges.
        edge_start)) AND ('SHIP_STRUCTURES_SCHEMA.VERTEX_POINT' IN
        TYPEOF(edges.edge_end)))) )) = 0)) )) = 0)) )) = 0);
    WR6: (SIZEOF(QUERY ( ebwm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.EDGE_BASED_WIREFRAME_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( eb <* ebwm\
        edge_based_wireframe_model.ebwm_boundary | (NOT (SIZEOF(
        QUERY ( edges <* eb.ces_edges | (NOT
        valid_wireframe_edge_curve(edges\edge_curve.edge_geometry)) ))
        = 0)) )) = 0)) )) = 0);
    wr7: (SIZEOF(QUERY ( ebwm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.EDGE_BASED_WIREFRAME_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( eb <* ebwm\
        edge_based_wireframe_model.ebwm_boundary | (NOT (SIZEOF(
        QUERY ( edges <* eb.ces_edges | (NOT (
        valid_wireframe_vertex_point(edges.edge_start\vertex_point.
        vertex_geometry) AND valid_wireframe_vertex_point(edges.
        edge_end\vertex_point.vertex_geometry))) )) = 0)) )) = 0)) ))
        = 0);
    wr8: (SIZEOF(QUERY ( mi <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM' IN TYPEOF(it)) ) | (
        NOT (('SHIP_STRUCTURES_SCHEMA.' +
        'EDGE_BASED_WIREFRAME_SHAPE_REPRESENTATION') IN TYPEOF(mi\
        mapped_item.mapping_source.mapped_representation))) )) = 0);
    wr9: (SELF.context_of_items\geometric_representation_context.
        coordinate_space_dimension = 3);
END_ENTITY; -- edge_based_wireframe_shape_representation

ENTITY edge_curve
  SUBTYPE OF (edge, geometric_representation_item);
    edge_geometry : curve;
    same_sense    : BOOLEAN;
END_ENTITY; -- edge_curve

ENTITY edge_loop
  SUBTYPE OF (loop, path);
  DERIVE
    ne : INTEGER := SIZEOF(SELF\path.edge_list);
  WHERE
    WR1: (SELF\path.edge_list[1].edge_start :=: SELF\path.edge_list[ne].
```

```
            edge_end);
  END_ENTITY; -- edge_loop

  ENTITY elementary_surface
    SUPERTYPE OF (ONEOF (plane,cylindrical_surface,conical_surface,
       spherical_surface,toroidal_surface))
    SUBTYPE OF (surface);
     position : axis2_placement_3d;
  END_ENTITY; -- elementary_surface

  ENTITY ellipse
    SUBTYPE OF (conic);
     semi_axis_1 : positive_length_measure;
     semi_axis_2 : positive_length_measure;
  END_ENTITY; -- ellipse

  ENTITY evaluated_degenerate_pcurve
    SUBTYPE OF (degenerate_pcurve);
     equivalent_point : cartesian_point;
  END_ENTITY; -- evaluated_degenerate_pcurve

  ENTITY executed_action
    SUBTYPE OF (action);
  END_ENTITY; -- executed_action

  ENTITY external_identification_assignment
    ABSTRACT SUPERTYPE
    SUBTYPE OF (identification_assignment);
     source : external_source;
  END_ENTITY; -- external_identification_assignment

  ENTITY external_source;
     source_id : source_item;
    DERIVE
     description : text := get_description_value(SELF);
    WHERE
     WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
  END_ENTITY; -- external_source

  ENTITY external_source_relationship;
     name           : label;
     description    : OPTIONAL text;
     relating_source : external_source;
     related_source  : external_source;
  END_ENTITY; -- external_source_relationship

  ENTITY externally_defined_class
    SUBTYPE OF (class, externally_defined_item);
  END_ENTITY; -- externally_defined_class
```



```
ENTITY externally_defined_general_property
  SUBTYPE OF (general_property, externally_defined_item);
END_ENTITY; -- externally_defined_general_property

ENTITY externally_defined_item;
    item_id : source_item;
    source  : external_source;
END_ENTITY; -- externally_defined_item

ENTITY face
  SUPERTYPE OF (ONEOF (face_surface,oriented_face))
  SUBTYPE OF (topological_representation_item);
    bounds : SET [1:?] OF face_bound;
  WHERE
    WR1: (NOT mixed_loop_type_set(list_to_set(list_face_loops(SELF))));
    WR2: (SIZEOF(QUERY ( temp <* bounds | (
           'SHIP_STRUCTURES_SCHEMA.FACE_OUTER_BOUND' IN TYPEOF(temp)) ))
           <= 1);
END_ENTITY; -- face

ENTITY face_based_surface_model
  SUBTYPE OF (geometric_representation_item);
    fbsm_faces : SET [1:?] OF connected_face_set;
END_ENTITY; -- face_based_surface_model

ENTITY face_bound
  SUBTYPE OF (topological_representation_item);
    bound       : loop;
    orientation : BOOLEAN;
END_ENTITY; -- face_bound

ENTITY face_outer_bound
  SUBTYPE OF (face_bound);
END_ENTITY; -- face_outer_bound

ENTITY face_surface
  SUBTYPE OF (face, geometric_representation_item);
    face_geometry : surface;
    same_sense    : BOOLEAN;
  WHERE
    WR1: (NOT ('SHIP_STRUCTURES_SCHEMA.ORIENTED_SURFACE' IN TYPEOF(
           face_geometry)));
END_ENTITY; -- face_surface

ENTITY faceted_brep
  SUBTYPE OF (manifold_solid_brep);
END_ENTITY; -- faceted_brep

ENTITY founded_item;
END_ENTITY; -- founded_item
```

```
ENTITY functionally_defined_transformation;
   name        : label;
   description : OPTIONAL text;
END_ENTITY; -- functionally_defined_transformation

ENTITY general_property;
   id          : identifier;
   name        : label;
   description : OPTIONAL text;
END_ENTITY; -- general_property

ENTITY geometric_curve_set
 SUBTYPE OF (geometric_set);
 WHERE
  WR1: (SIZEOF(QUERY ( temp <* SELF\geometric_set.elements | (
        'SHIP_STRUCTURES_SCHEMA.SURFACE' IN TYPEOF(temp)) )) = 0);
END_ENTITY; -- geometric_curve_set

ENTITY geometric_representation_context
 SUBTYPE OF (representation_context);
  coordinate_space_dimension : dimension_count;
END_ENTITY; -- geometric_representation_context

ENTITY geometric_representation_item
 SUPERTYPE OF (ONEOF (point,direction,vector,placement,
    cartesian_transformation_operator,curve,surface,edge_curve,
    face_surface,poly_loop,vertex_point,solid_model,
    face_based_surface_model,edge_based_wireframe_model,geometric_set))
 SUBTYPE OF (representation_item);
 DERIVE
  dim : dimension_count := dimension_of(SELF);
 WHERE
  WR1: (SIZEOF(QUERY ( using_rep <* using_representations(SELF) | (
        NOT (
        'SHIP_STRUCTURES_SCHEMA.GEOMETRIC_REPRESENTATION_CONTEXT' IN
         TYPEOF(using_rep.context_of_items))) )) = 0);
END_ENTITY; -- geometric_representation_item

ENTITY geometric_set
 SUPERTYPE OF (geometric_curve_set)
 SUBTYPE OF (geometric_representation_item);
  elements : SET [1:?] OF geometric_set_select;
END_ENTITY; -- geometric_set

ENTITY geometrically_bounded_wireframe_shape_representation
 SUBTYPE OF (shape_representation);
 WHERE
  WR1: (SIZEOF(QUERY ( it <* SELF.items | (NOT (SIZEOF(TYPEOF(it) * [
        'SHIP_STRUCTURES_SCHEMA.GEOMETRIC_CURVE_SET',
        'SHIP_STRUCTURES_SCHEMA.AXIS2_PLACEMENT_3D',
```



```
        'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM']) = 1)) )) = 0);
    WR2: (SIZEOF(QUERY ( it <* SELF.items | (SIZEOF(TYPEOF(it) * [
        'SHIP_STRUCTURES_SCHEMA.GEOMETRIC_CURVE_SET',
        'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM']) = 1) )) >= 1);
    WR3: (SIZEOF(QUERY ( gcs <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.GEOMETRIC_CURVE_SET' IN TYPEOF(it)) )
        | (NOT (SIZEOF(QUERY ( crv <* QUERY ( elem <* gcs\
        geometric_set.elements | ('SHIP_STRUCTURES_SCHEMA.CURVE' IN
        TYPEOF(elem)) ) | (NOT valid_geometrically_bounded_wf_curve(
        crv)) )) = 0)) )) = 0);
    WR4: (SIZEOF(QUERY ( gcs <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.GEOMETRIC_CURVE_SET' IN TYPEOF(it)) )
        | (NOT (SIZEOF(QUERY ( pnts <* QUERY ( elem <* gcs\
        geometric_set.elements | ('SHIP_STRUCTURES_SCHEMA.POINT' IN
        TYPEOF(elem)) ) | (NOT valid_geometrically_bounded_wf_point(
        pnts)) )) = 0)) )) = 0);
    WR5: (SIZEOF(QUERY ( gcs <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.GEOMETRIC_CURVE_SET' IN TYPEOF(it)) )
        | (NOT (SIZEOF(QUERY ( cnc <* QUERY ( elem <* gcs\
        geometric_set.elements | ('SHIP_STRUCTURES_SCHEMA.CONIC' IN
        TYPEOF(elem)) ) | (NOT (
        'SHIP_STRUCTURES_SCHEMA.AXIS2_PLACEMENT_3D' IN TYPEOF(cnc\
        conic.position))) )) = 0)) )) = 0);
    WR6: (SIZEOF(QUERY ( gcs <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.GEOMETRIC_CURVE_SET' IN TYPEOF(it)) )
        | (NOT (SIZEOF(QUERY ( pline <* QUERY ( elem <* gcs\
        geometric_set.elements | ('SHIP_STRUCTURES_SCHEMA.POLYLINE'
        IN TYPEOF(elem)) ) | (NOT (SIZEOF(pline\polyline.points) > 2)) ))
        = 0)) )) = 0);
    wr7: (SIZEOF(QUERY ( mi <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM' IN TYPEOF(it)) ) | (
        NOT (('SHIP_STRUCTURES_SCHEMA.' +
        'GEOMETRICALLY_BOUNDED_WIREFRAME_SHAPE_REPRESENTATION') IN
        TYPEOF(mi\mapped_item.mapping_source.mapped_representation))) ))
        = 0);
END_ENTITY; -- geometrically_bounded_wireframe_shape_representation

ENTITY global_uncertainty_assigned_context
  SUBTYPE OF (representation_context);
    uncertainty : SET [1:?] OF uncertainty_measure_with_unit;
END_ENTITY; -- global_uncertainty_assigned_context

ENTITY global_unit_assigned_context
  SUBTYPE OF (representation_context);
    units : SET [1:?] OF unit;
END_ENTITY; -- global_unit_assigned_context

ENTITY group;
    name       : label;
    description : OPTIONAL text;
```

```
  DERIVE
    id : identifier := get_id_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1);
END_ENTITY; -- group

ENTITY group_assignment
  ABSTRACT SUPERTYPE;
    assigned_group : group;
  DERIVE
    role : object_role := get_role(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'ROLE_ASSOCIATION.ITEM_WITH_ROLE')) <= 1);
END_ENTITY; -- group_assignment

ENTITY group_relationship;
    name          : label;
    description   : OPTIONAL text;
    relating_group : group;
    related_group  : group;
END_ENTITY; -- group_relationship

ENTITY hyperbola
  SUBTYPE OF (conic);
    semi_axis     : positive_length_measure;
    semi_imag_axis : positive_length_measure;
END_ENTITY; -- hyperbola

ENTITY id_attribute;
    attribute_value : identifier;
    identified_item : id_attribute_select;
END_ENTITY; -- id_attribute

ENTITY identification_assignment
  ABSTRACT SUPERTYPE;
    assigned_id : identifier;
    role       : identification_role;
END_ENTITY; -- identification_assignment

ENTITY identification_assignment_relationship;
    name                          : label;
    description                   : OPTIONAL text;
    relating_identification_assignment : identification_assignment;
    related_identification_assignment  : identification_assignment;
END_ENTITY; -- identification_assignment_relationship

ENTITY identification_role;
    name       : label;
```



```
    description : OPTIONAL text;
  END_ENTITY; -- identification_role

  ENTITY intersection_curve
    SUBTYPE OF (surface_curve);
    WHERE
      WR1: (SIZEOF(SELF\surface_curve.associated_geometry) = 2);
      WR2: (associated_surface(SELF\surface_curve.associated_geometry[1])
           <> associated_surface(SELF\surface_curve.associated_geometry
           [2]));
  END_ENTITY; -- intersection_curve

  ENTITY item_defined_transformation;
     name            : label;
     description     : OPTIONAL text;
     transform_item_1 : representation_item;
     transform_item_2 : representation_item;
  END_ENTITY; -- item_defined_transformation

  ENTITY known_source
    SUBTYPE OF (pre_defined_item, external_source);
    WHERE
      WR1: (SELF\pre_defined_item.name = 'ISO 13584 library');
      WR2: (SIZEOF(QUERY ( oa <*
USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.APPLIED_ORGANIZATION_ASSIGNMENT.ITE
MS')
           | ((NOT ('SHIP_STRUCTURES_SCHEMA.ORGANIZATION_ASSIGNMENT'
           IN TYPEOF(oa))) AND (oa.role.name = 'library supplier')) ))
           = 0);
  END_ENTITY; -- known_source

  ENTITY length_unit
    SUBTYPE OF (named_unit);
    WHERE
      WR1: ((SELF\named_unit.dimensions.length_exponent = 1) AND (SELF\
           named_unit.dimensions.mass_exponent = 0) AND (SELF\
           named_unit.dimensions.time_exponent = 0) AND (SELF\
           named_unit.dimensions.electric_current_exponent = 0) AND (
           SELF\named_unit.dimensions.
           thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
           .dimensions.amount_of_substance_exponent = 0) AND (SELF\
           named_unit.dimensions.luminous_intensity_exponent = 0));
  END_ENTITY; -- length_unit

  ENTITY library_applied_version_assignment
    SUBTYPE OF (applied_external_identification_assignment);
    WHERE
      WR1: (SELF\identification_assignment.role.name = 'property version');
      WR2: (SIZEOF(SELF\applied_external_identification_assignment.items)
           = 1);
```

```
    WR3: ('SHIP_STRUCTURES_SCHEMA.KNOWN_SOURCE' IN TYPEOF(SELF\
         external_identification_assignment.source));
END_ENTITY; -- library_applied_version_assignment

ENTITY library_class_version_assignment
  SUBTYPE OF (applied_external_identification_assignment);
  WHERE
    WR1: (SELF\identification_assignment.role.name = 'class version');
    WR2: (SIZEOF(SELF\applied_external_identification_assignment.items)
         > 1);
    WR3: ('SHIP_STRUCTURES_SCHEMA.KNOWN_SOURCE' IN TYPEOF(SELF\
         external_identification_assignment.source));
END_ENTITY; -- library_class_version_assignment

ENTITY line
  SUBTYPE OF (curve);
    pnt : cartesian_point;
    dir : vector;
  WHERE
    WR1: (dir.dim = pnt.dim);
END_ENTITY; -- line

ENTITY local_time;
    hour_component   : hour_in_day;
    minute_component : OPTIONAL minute_in_hour;
    second_component : OPTIONAL second_in_minute;
    zone             : coordinated_universal_time_offset;
  WHERE
    WR1: valid_time(SELF);
END_ENTITY; -- local_time

ENTITY loop
  SUPERTYPE OF (ONEOF (vertex_loop,edge_loop,poly_loop))
  SUBTYPE OF (topological_representation_item);
END_ENTITY; -- loop

ENTITY manifold_solid_brep
  SUBTYPE OF (solid_model);
    outer : closed_shell;
END_ENTITY; -- manifold_solid_brep

ENTITY mapped_item
  SUBTYPE OF (representation_item);
    mapping_source : representation_map;
    mapping_target : representation_item;
  WHERE
    WR1: acyclic_mapped_representation(using_representations(SELF),[SELF]);
END_ENTITY; -- mapped_item

ENTITY measure_with_unit;
```



```
    value_component : measure_value;
    unit_component  : unit;
  WHERE
    WR1: valid_units(SELF);
END_ENTITY; -- measure_with_unit

ENTITY name_attribute;
    attribute_value : label;
    named_item      : name_attribute_select;
END_ENTITY; -- name_attribute

ENTITY named_unit
  SUPERTYPE OF (ONEOF (si_unit,conversion_based_unit,
      context_dependent_unit) ANDOR ONEOF (length_unit,time_unit,
      thermodynamic_temperature_unit,plane_angle_unit));
    dimensions : dimensional_exponents;
END_ENTITY; -- named_unit

ENTITY non_manifold_surface_shape_representation
  SUBTYPE OF (shape_representation);
  WHERE
    WR1 : (SIZEOF(QUERY ( it <* SELF.items | (NOT (SIZEOF([
          'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL',
          'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM',
          'SHIP_STRUCTURES_SCHEMA.AXIS2_PLACEMENT_3D'] * TYPEOF(it))
          = 1)) )) = 0);
    WR2 : (SIZEOF(QUERY ( it <* SELF.items | (SIZEOF([
          'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL',
          'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM'] * TYPEOF(it)) = 1) ))
          > 0);
    WR3 : (SIZEOF(QUERY ( mi <* QUERY ( it <* SELF.items | (
          'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM' IN TYPEOF(it)) ) | (
          NOT ((('SHIP_STRUCTURES_SCHEMA.' +
          'NON_MANIFOLD_SURFACE_SHAPE_REPRESENTATION') IN TYPEOF(mi\
          mapped_item.mapping_source.mapped_representation)) AND (
          SIZEOF(QUERY ( mr_it <* mi\mapped_item.mapping_source.
          mapped_representation.items | (
          'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
          TYPEOF(mr_it)) )) > 0))) )) = 0);
    WR4 : (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
          'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
          TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
          face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
          QUERY ( fa <* cfs.cfs_faces | (NOT (SIZEOF([
          'SHIP_STRUCTURES_SCHEMA.FACE_SURFACE',
          'SHIP_STRUCTURES_SCHEMA.ORIENTED_FACE'] * TYPEOF(fa)) = 1)) ))
          = 0)) )) = 0)) )) = 0);
    WR5 : (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
          'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
          TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
```

```
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( f_sf <* QUERY ( fa <* cfs.cfs_faces | (
        'SHIP_STRUCTURES_SCHEMA.FACE_SURFACE' IN TYPEOF(fa)) ) | (
        NOT (('SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(f_sf))
        OR nmsf_surface_check(f_sf\face_surface.face_geometry))) ))
        = 0)) )) = 0)) )) = 0);
    WR6 : (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( o_fa <* QUERY ( fa <* cfs.cfs_faces | (
        'SHIP_STRUCTURES_SCHEMA.ORIENTED_FACE' IN TYPEOF(fa)) ) | (
        NOT (('SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(o_fa
        \oriented_face.face_element)) OR nmsf_surface_check(o_fa\
        oriented_face.face_element\face_surface.face_geometry))) ))
        = 0)) )) = 0)) )) = 0);
    WR7 : (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( fa <* cfs.cfs_faces | (NOT ((
        'SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(fa)) OR (
        SIZEOF(QUERY ( bnds <* fa.bounds | (NOT (SIZEOF([
        'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP',
        'SHIP_STRUCTURES_SCHEMA.VERTEX_LOOP'] * TYPEOF(bnds.bound))
        = 1)) )) = 0))) )) = 0)) )) = 0)) )) = 0);
    WR8 : (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( fa <* cfs.cfs_faces | (NOT ((
        'SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(fa)) OR (
        SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* fa.bounds | (
        'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
         | (NOT (SIZEOF(QUERY ( oe <* elp_fbnds\path.edge_list | (
        NOT ('SHIP_STRUCTURES_SCHEMA.EDGE_CURVE' IN TYPEOF(oe.
        edge_element))) )) = 0)) )) = 0))) )) = 0)) )) = 0)) )) = 0);
    WR9 : (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( fa <* cfs.cfs_faces | (NOT ((
        'SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(fa)) OR (
        SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* fa.bounds | (
        'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
         | (NOT (SIZEOF(QUERY ( oe_cv <* QUERY ( oe <* elp_fbnds\
        path.edge_list | ('SHIP_STRUCTURES_SCHEMA.EDGE_CURVE' IN
        TYPEOF(oe.edge_element)) ) | (NOT (SIZEOF([
        'SHIP_STRUCTURES_SCHEMA.B_SPLINE_CURVE',
        'SHIP_STRUCTURES_SCHEMA.CONIC',
```



```
        'SHIP_STRUCTURES_SCHEMA.CURVE_REPLICA',
        'SHIP_STRUCTURES_SCHEMA.LINE',
        'SHIP_STRUCTURES_SCHEMA.OFFSET_CURVE_3D',
        'SHIP_STRUCTURES_SCHEMA.PCURVE',
        'SHIP_STRUCTURES_SCHEMA.POLYLINE',
        'SHIP_STRUCTURES_SCHEMA.SURFACE_CURVE'] * TYPEOF(oe_cv.
        edge_element\edge_curve.edge_geometry)) = 1)) )) = 0)) )) =
         0))) )) = 0)) )) = 0)) )) = 0);
    WR10: (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( fa <* cfs.cfs_faces | (NOT ((
        'SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(fa)) OR (
        SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* fa.bounds | (
        'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
         | (NOT (SIZEOF(QUERY ( oe <* elp_fbnds\path.edge_list | (
        NOT nmsf_curve_check(oe.edge_element\edge_curve.
        edge_geometry)) )) = 0)) )) = 0))) )) = 0)) )) = 0)) )) = 0);
    WR11: (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( fa <* cfs.cfs_faces | (NOT ((
        'SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(fa)) OR (
        SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* fa.bounds | (
        'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
         | (NOT (SIZEOF(QUERY ( oe <* elp_fbnds\path.edge_list | (
        NOT (('SHIP_STRUCTURES_SCHEMA.VERTEX_POINT' IN TYPEOF(oe.
        edge_element.edge_start)) AND (
        'SHIP_STRUCTURES_SCHEMA.VERTEX_POINT' IN TYPEOF(oe.
        edge_element.edge_end)))) )) = 0)) )) = 0))) )) = 0)) )) =
        0)) )) = 0);
    WR12: (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( fa <* cfs.cfs_faces | (NOT ((
        'SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(fa)) OR (
        SIZEOF(QUERY ( elp_fbnds <* QUERY ( bnds <* fa.bounds | (
        'SHIP_STRUCTURES_SCHEMA.EDGE_LOOP' IN TYPEOF(bnds.bound)) )
         | (NOT (SIZEOF(QUERY ( oe <* elp_fbnds\path.edge_list | (
        NOT ((SIZEOF(['SHIP_STRUCTURES_SCHEMA.CARTESIAN_POINT',
        'SHIP_STRUCTURES_SCHEMA.DEGENERATE_PCURVE',
        'SHIP_STRUCTURES_SCHEMA.POINT_ON_CURVE',
        'SHIP_STRUCTURES_SCHEMA.POINT_ON_SURFACE'] * TYPEOF(oe.
        edge_element.edge_start\vertex_point.vertex_geometry)) = 1)
        AND (SIZEOF(['SHIP_STRUCTURES_SCHEMA.CARTESIAN_POINT',
        'SHIP_STRUCTURES_SCHEMA.DEGENERATE_PCURVE',
        'SHIP_STRUCTURES_SCHEMA.POINT_ON_CURVE',
```

```
        'SHIP_STRUCTURES_SCHEMA.POINT_ON_SURFACE'] * TYPEOF(oe.
        edge_element.edge_end\vertex_point.vertex_geometry)) = 1))) ))
        = 0)) )) = 0))) )) = 0)) )) = 0)) )) = 0);
  WR13: (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( fa <* cfs.cfs_faces | (NOT ((
        'SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(fa)) OR (
        SIZEOF(QUERY ( vlp_fbnds <* QUERY ( bnds <* fa.bounds | (
        'SHIP_STRUCTURES_SCHEMA.VERTEX_LOOP' IN TYPEOF(bnds.bound)) )
         | (NOT ('SHIP_STRUCTURES_SCHEMA.VERTEX_POINT' IN TYPEOF(
        vlp_fbnds\vertex_loop.loop_vertex))) )) = 0))) )) = 0)) ))
        = 0)) )) = 0);
  WR14: (SIZEOF(QUERY ( fbsm <* QUERY ( it <* SELF.items | (
        'SHIP_STRUCTURES_SCHEMA.FACE_BASED_SURFACE_MODEL' IN
        TYPEOF(it)) ) | (NOT (SIZEOF(QUERY ( cfs <* fbsm\
        face_based_surface_model.fbsm_faces | (NOT (SIZEOF(
        QUERY ( fa <* cfs.cfs_faces | (NOT ((
        'SHIP_STRUCTURES_SCHEMA.ADVANCED_FACE' IN TYPEOF(fa)) OR (
        SIZEOF(QUERY ( vlp_fbnds <* QUERY ( bnds <* fa.bounds | (
        'SHIP_STRUCTURES_SCHEMA.VERTEX_LOOP' IN TYPEOF(bnds.bound)) )
         | (NOT (SIZEOF(['SHIP_STRUCTURES_SCHEMA.CARTESIAN_POINT',
        'SHIP_STRUCTURES_SCHEMA.DEGENERATE_PCURVE',
        'SHIP_STRUCTURES_SCHEMA.POINT_ON_CURVE',
        'SHIP_STRUCTURES_SCHEMA.POINT_ON_SURFACE'] * TYPEOF(
        vlp_fbnds\vertex_loop.loop_vertex\vertex_point.
        vertex_geometry)) = 1)) )) = 0))) )) = 0)) )) = 0)) )) = 0);
END_ENTITY; -- non_manifold_surface_shape_representation

ENTITY object_role;
  name       : label;
  description : OPTIONAL text;
END_ENTITY; -- object_role

ENTITY offset_curve_3d
 SUBTYPE OF (curve);
  basis_curve    : curve;
  distance       : length_measure;
  self_intersect : LOGICAL;
  ref_direction  : direction;
 WHERE
  WR1: ((basis_curve.dim = 3) AND (ref_direction.dim = 3));
END_ENTITY; -- offset_curve_3d

ENTITY offset_surface
 SUBTYPE OF (surface);
  basis_surface  : surface;
  distance       : length_measure;
  self_intersect : LOGICAL;
```



```
END_ENTITY; -- offset_surface

ENTITY open_shell
  SUBTYPE OF (connected_face_set);
END_ENTITY; -- open_shell

ENTITY ordinal_date
  SUBTYPE OF (date);
    day_component : day_in_year_number;
  WHERE
    WR1: (((NOT leap_year(SELF.year_component)) AND (1 <= day_component)
          AND (day_component <= 365)) OR (leap_year(SELF.
          year_component) AND (1 <= day_component) AND (day_component
          <= 366)));
END_ENTITY; -- ordinal_date

ENTITY organization;
    id          : OPTIONAL identifier;
    name        : label;
    description : OPTIONAL text;
END_ENTITY; -- organization

ENTITY organization_assignment
  ABSTRACT SUPERTYPE;
    assigned_organization : organization;
    role                  : organization_role;
END_ENTITY; -- organization_assignment

ENTITY organization_role;
    name : label;
  DERIVE
    description : text := get_description_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
END_ENTITY; -- organization_role

ENTITY organizational_address
  SUBTYPE OF (address);
    organizations : SET [1:?] OF organization;
    description   : OPTIONAL text;
END_ENTITY; -- organizational_address

ENTITY organizational_project;
    name                    : label;
    description             : OPTIONAL text;
    responsible_organizations : SET [1:?] OF organization;
  DERIVE
    id : identifier := get_id_value(SELF);
  WHERE
```

```
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1);
 END_ENTITY; -- organizational_project

 ENTITY oriented_closed_shell
  SUBTYPE OF (closed_shell);
   closed_shell_element : closed_shell;
   orientation          : BOOLEAN;
  DERIVE
   SELF\connected_face_set.cfs_faces : SET [1:?] OF face :=
                                conditional_reverse(SELF.
                                orientation,SELF.
                                closed_shell_element.cfs_faces);
  WHERE
   WR1: (NOT ('SHIP_STRUCTURES_SCHEMA.ORIENTED_CLOSED_SHELL' IN TYPEOF(
         SELF.closed_shell_element)));
 END_ENTITY; -- oriented_closed_shell

 ENTITY oriented_edge
  SUBTYPE OF (edge);
   edge_element : edge;
   orientation  : BOOLEAN;
  DERIVE
   SELF\edge.edge_start : vertex := boolean_choose(SELF.orientation,
                       SELF.edge_element.edge_start,SELF.
                       edge_element.edge_end);
   SELF\edge.edge_end   : vertex := boolean_choose(SELF.orientation,
                       SELF.edge_element.edge_end,SELF.
                       edge_element.edge_start);
  WHERE
   WR1: (NOT ('SHIP_STRUCTURES_SCHEMA.ORIENTED_EDGE' IN TYPEOF(SELF.
         edge_element)));
 END_ENTITY; -- oriented_edge

 ENTITY oriented_face
  SUBTYPE OF (face);
   face_element : face;
   orientation  : BOOLEAN;
  DERIVE
   SELF\face.bounds : SET [1:?] OF face_bound := conditional_reverse(
                   SELF.orientation,SELF.face_element.bounds);
  WHERE
   WR1: (NOT ('SHIP_STRUCTURES_SCHEMA.ORIENTED_FACE' IN TYPEOF(SELF.
         face_element)));
 END_ENTITY; -- oriented_face

 ENTITY oriented_open_shell
  SUBTYPE OF (open_shell);
   open_shell_element : open_shell;
   orientation        : BOOLEAN;
```



```
  DERIVE
    SELF\connected_face_set.cfs_faces : SET [1:?] OF face :=
                                  conditional_reverse(SELF.
                                  orientation,SELF.
                                  open_shell_element.cfs_faces);
  WHERE
    WR1: (NOT ('SHIP_STRUCTURES_SCHEMA.ORIENTED_OPEN_SHELL' IN TYPEOF(
          SELF.open_shell_element)));
END_ENTITY; -- oriented_open_shell

ENTITY oriented_path
  SUBTYPE OF (path);
    path_element : path;
    orientation  : BOOLEAN;
  DERIVE
    SELF\path.edge_list : LIST [1:?] OF UNIQUE oriented_edge :=
                        conditional_reverse(SELF.orientation,SELF.
                        path_element.edge_list);
  WHERE
    WR1: (NOT ('SHIP_STRUCTURES_SCHEMA.ORIENTED_PATH' IN TYPEOF(SELF.
          path_element)));
END_ENTITY; -- oriented_path

ENTITY oriented_surface
  SUBTYPE OF (surface);
    orientation : BOOLEAN;
END_ENTITY; -- oriented_surface

ENTITY parabola
  SUBTYPE OF (conic);
    focal_dist : length_measure;
  WHERE
    WR1: (focal_dist <> 0);
END_ENTITY; -- parabola

ENTITY parametric_representation_context
  SUBTYPE OF (representation_context);
END_ENTITY; -- parametric_representation_context

ENTITY path
  SUPERTYPE OF (ONEOF (edge_loop,oriented_path))
  SUBTYPE OF (topological_representation_item);
    edge_list : LIST [1:?] OF UNIQUE oriented_edge;
  WHERE
    WR1: path_head_to_tail(SELF);
END_ENTITY; -- path

ENTITY pcurve
  SUBTYPE OF (curve);
    basis_surface      : surface;
```

```
    reference_to_curve : definitional_representation;
  WHERE
    WR1: (SIZEOF(reference_to_curve\representation.items) = 1);
    WR2: ('SHIP_STRUCTURES_SCHEMA.CURVE' IN TYPEOF(reference_to_curve\
          representation.items[1]));
    WR3: (reference_to_curve\representation.items[1]\
          geometric_representation_item.dim = 2);
END_ENTITY; -- pcurve

ENTITY person;
    id          : identifier;
    last_name    : OPTIONAL label;
    first_name   : OPTIONAL label;
    middle_names  : OPTIONAL LIST [1:?] OF label;
    prefix_titles : OPTIONAL LIST [1:?] OF label;
    suffix_titles : OPTIONAL LIST [1:?] OF label;
  WHERE
    WR1: (EXISTS(last_name) OR EXISTS(first_name));
END_ENTITY; -- person

ENTITY person_and_organization;
    the_person      : person;
    the_organization : organization;
  DERIVE
    name       : label := get_name_value(SELF);
    description : text := get_description_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'NAME_ATTRIBUTE.NAMED_ITEM')) <= 1);
    WR2: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
END_ENTITY; -- person_and_organization

ENTITY person_and_organization_assignment
  ABSTRACT SUPERTYPE;
    assigned_person_and_organization : person_and_organization;
    role                        : person_and_organization_role;
END_ENTITY; -- person_and_organization_assignment

ENTITY person_and_organization_role;
    name : label;
  DERIVE
    description : text := get_description_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
END_ENTITY; -- person_and_organization_role

ENTITY person_assignment
  ABSTRACT SUPERTYPE;
```



```
    assigned_person : person;
    role            : person_role;
END_ENTITY; -- person_assignment

ENTITY person_role;
    name : label;
  DERIVE
    description : text := get_description_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
END_ENTITY; -- person_role

ENTITY personal_address
  SUBTYPE OF (address);
    people      : SET [1:?] OF person;
    description : OPTIONAL text;
END_ENTITY; -- personal_address

ENTITY placement
  SUPERTYPE OF (ONEOF (axis1_placement,axis2_placement_2d,
      axis2_placement_3d))
  SUBTYPE OF (geometric_representation_item);
    location : cartesian_point;
END_ENTITY; -- placement

ENTITY plane
  SUBTYPE OF (elementary_surface);
END_ENTITY; -- plane

ENTITY plane_angle_unit
  SUBTYPE OF (named_unit);
  WHERE
    WR1: ((SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
          named_unit.dimensions.mass_exponent = 0) AND (SELF\
          named_unit.dimensions.time_exponent = 0) AND (SELF\
          named_unit.dimensions.electric_current_exponent = 0) AND (
          SELF\named_unit.dimensions.
          thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
          .dimensions.amount_of_substance_exponent = 0) AND (SELF\
          named_unit.dimensions.luminous_intensity_exponent = 0));
END_ENTITY; -- plane_angle_unit

ENTITY point
  SUPERTYPE OF (ONEOF (cartesian_point,point_on_curve,point_on_surface,
      point_replica,degenerate_pcurve))
  SUBTYPE OF (geometric_representation_item);
END_ENTITY; -- point

ENTITY point_on_curve
```

```
  SUBTYPE OF (point);
    basis_curve     : curve;
    point_parameter : parameter_value;
END_ENTITY; -- point_on_curve

ENTITY point_on_surface
  SUBTYPE OF (point);
    basis_surface     : surface;
    point_parameter_u : parameter_value;
    point_parameter_v : parameter_value;
END_ENTITY; -- point_on_surface

ENTITY point_replica
  SUBTYPE OF (point);
    parent_pt      : point;
    transformation : cartesian_transformation_operator;
  WHERE
    WR1: (transformation.dim = parent_pt.dim);
    WR2: acyclic_point_replica(SELF,parent_pt);
END_ENTITY; -- point_replica

ENTITY poly_loop
  SUBTYPE OF (loop, geometric_representation_item);
    polygon : LIST [3:?] OF UNIQUE cartesian_point;
END_ENTITY; -- poly_loop

ENTITY polyline
  SUBTYPE OF (bounded_curve);
    points : LIST [2:?] OF cartesian_point;
END_ENTITY; -- polyline

ENTITY pre_defined_item;
    name : label;
END_ENTITY; -- pre_defined_item

ENTITY product;
    id                : identifier;
    name              : label;
    description       : OPTIONAL text;
    frame_of_reference : SET [1:?] OF product_context;
END_ENTITY; -- product

ENTITY product_category;
    name       : label;
    description : OPTIONAL text;
  DERIVE
    id : identifier := get_id_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1);
```



```
END_ENTITY; -- product_category

ENTITY product_category_relationship;
   name        : label;
   description  : OPTIONAL text;
   category     : product_category;
   sub_category : product_category;
  WHERE
   WR1: acyclic_product_category_relationship(SELF,[SELF.sub_category]);
END_ENTITY; -- product_category_relationship

ENTITY product_context
  SUBTYPE OF (application_context_element);
   discipline_type : label;
END_ENTITY; -- product_context

ENTITY product_definition;
   id              : identifier;
   description       : OPTIONAL text;
   formation         : product_definition_formation;
   frame_of_reference : product_definition_context;
  DERIVE
   name : label := get_name_value(SELF);
  WHERE
   WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
         'NAME_ATTRIBUTE.NAMED_ITEM')) <= 1);
END_ENTITY; -- product_definition

ENTITY product_definition_context
  SUBTYPE OF (application_context_element);
   life_cycle_stage : label;
END_ENTITY; -- product_definition_context

ENTITY product_definition_formation;
   id         : identifier;
   description : OPTIONAL text;
   of_product  : product;
  UNIQUE
   ur1 : id, of_product;
END_ENTITY; -- product_definition_formation

ENTITY product_definition_relationship;
   id                      : identifier;
   name                      : label;
   description               : OPTIONAL text;
   relating_product_definition : product_definition;
   related_product_definition  : product_definition;
END_ENTITY; -- product_definition_relationship

ENTITY product_definition_shape
```

```
  SUBTYPE OF (property_definition);
  UNIQUE
    ur1 : definition;
  WHERE
    WR1: (SIZEOF([
          'SHIP_STRUCTURES_SCHEMA.CHARACTERIZED_PRODUCT_DEFINITION',
          'SHIP_STRUCTURES_SCHEMA.CHARACTERIZED_OBJECT'] * TYPEOF(SELF
          \property_definition.definition)) > 0);
END_ENTITY; -- product_definition_shape

ENTITY product_definition_with_associated_documents
  SUBTYPE OF (product_definition);
    documentation_ids : SET [1:?] OF document;
END_ENTITY; -- product_definition_with_associated_documents

ENTITY product_related_product_category
  SUBTYPE OF (product_category);
    products : SET [1:?] OF product;
END_ENTITY; -- product_related_product_category

ENTITY property_definition;
    name       : label;
    description : OPTIONAL text;
    definition  : characterized_definition;
  DERIVE
    id : identifier := get_id_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1);
END_ENTITY; -- property_definition

ENTITY property_definition_relationship;
    name                      : label;
    description               : text;
    relating_property_definition : property_definition;
    related_property_definition  : property_definition;
END_ENTITY; -- property_definition_relationship

ENTITY property_definition_representation;
    definition         : represented_definition;
    used_representation : representation;
  DERIVE
    description : text := get_description_value(SELF);
    name       : label := get_name_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
    WR2: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
          'NAME_ATTRIBUTE.NAMED_ITEM')) <= 1);
END_ENTITY; -- property_definition_representation
```



```
ENTITY quasi_uniform_curve
  SUBTYPE OF (b_spline_curve);
END_ENTITY; -- quasi_uniform_curve

ENTITY quasi_uniform_surface
  SUBTYPE OF (b_spline_surface);
END_ENTITY; -- quasi_uniform_surface

ENTITY rational_b_spline_curve
  SUBTYPE OF (b_spline_curve);
    weights_data : LIST [2:?] OF REAL;
  DERIVE
    weights : ARRAY [0:upper_index_on_control_points] OF REAL :=
              list_to_array(weights_data,0,
              upper_index_on_control_points);
  WHERE
    WR1: (SIZEOF(weights_data) = SIZEOF(SELF\b_spline_curve.
         control_points_list));
    WR2: curve_weights_positive(SELF);
END_ENTITY; -- rational_b_spline_curve

ENTITY rational_b_spline_surface
  SUBTYPE OF (b_spline_surface);
    weights_data : LIST [2:?] OF LIST [2:?] OF REAL;
  DERIVE
    weights : ARRAY [0:u_upper] OF ARRAY [0:v_upper] OF REAL :=
              make_array_of_array(weights_data,0,u_upper,0,v_upper);
  WHERE
    WR1: ((SIZEOF(weights_data) = SIZEOF(SELF\b_spline_surface.
         control_points_list)) AND (SIZEOF(weights_data[1]) = SIZEOF(
         SELF\b_spline_surface.control_points_list[1])));
    WR2: surface_weights_positive(SELF);
END_ENTITY; -- rational_b_spline_surface

ENTITY reparametrised_composite_curve_segment
  SUBTYPE OF (composite_curve_segment);
    param_length : parameter_value;
  WHERE
    WR1: (param_length > 0);
END_ENTITY; -- reparametrised_composite_curve_segment

ENTITY representation;
    name            : label;
    items           : SET [1:?] OF representation_item;
    context_of_items : representation_context;
  DERIVE
    id          : identifier := get_id_value(SELF);
    description : text := get_description_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
```

```
        'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1);
    WR2: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
        'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1);
END_ENTITY; -- representation

ENTITY representation_context;
   context_identifier : identifier;
   context_type       : text;
  INVERSE
   representations_in_context : SET [1:?] OF representation FOR
                            context_of_items;
END_ENTITY; -- representation_context

ENTITY representation_item;
   name : label;
  WHERE
   WR1: (SIZEOF(using_representations(SELF)) > 0);
END_ENTITY; -- representation_item

ENTITY representation_map;
   mapping_origin        : representation_item;
   mapped_representation : representation;
  INVERSE
   map_usage : SET [1:?] OF mapped_item FOR mapping_source;
  WHERE
   WR1: item_in_context(SELF.mapping_origin,SELF.mapped_representation.
        context_of_items);
END_ENTITY; -- representation_map

ENTITY representation_relationship;
   name       : label;
   description : OPTIONAL text;
   rep_1      : representation;
   rep_2      : representation;
END_ENTITY; -- representation_relationship

ENTITY role_association;
   role          : object_role;
   item_with_role : role_select;
END_ENTITY; -- role_association

ENTITY seam_curve
  SUBTYPE OF (surface_curve);
  WHERE
   WR1: (SIZEOF(SELF\surface_curve.associated_geometry) = 2);
   WR2: (associated_surface(SELF\surface_curve.associated_geometry[1])
        = associated_surface(SELF\surface_curve.associated_geometry[
        2]));
   WR3: ('SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(SELF\surface_curve.
        associated_geometry[1]));
```



```
    WR4: ('SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(SELF\surface_curve.
         associated_geometry[2]));
END_ENTITY; -- seam_curve

ENTITY shape_aspect;
    name               : label;
    description        : OPTIONAL text;
    of_shape           : product_definition_shape;
    product_definitional : LOGICAL;
  DERIVE
    id : identifier := get_id_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
         'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1);
END_ENTITY; -- shape_aspect

ENTITY shape_aspect_relationship;
    name               : label;
    description        : OPTIONAL text;
    relating_shape_aspect : shape_aspect;
    related_shape_aspect  : shape_aspect;
  DERIVE
    id : identifier := get_id_value(SELF);
  WHERE
    WR1: (SIZEOF(USEDIN(SELF,'SHIP_STRUCTURES_SCHEMA.' +
         'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1);
END_ENTITY; -- shape_aspect_relationship

ENTITY shape_definition_representation
  SUBTYPE OF (property_definition_representation);
  WHERE
    WR1: (('SHIP_STRUCTURES_SCHEMA.PRODUCT_DEFINITION_SHAPE' IN TYPEOF(
         SELF.definition)) OR (
         'SHIP_STRUCTURES_SCHEMA.SHAPE_DEFINITION' IN TYPEOF(SELF.
         definition.definition)));
    WR2: ('SHIP_STRUCTURES_SCHEMA.SHAPE_REPRESENTATION' IN TYPEOF(SELF.
         used_representation));
END_ENTITY; -- shape_definition_representation

ENTITY shape_representation
  SUBTYPE OF (representation);
END_ENTITY; -- shape_representation

ENTITY si_unit
  SUBTYPE OF (named_unit);
    prefix : OPTIONAL si_prefix;
    name   : si_unit_name;
  DERIVE
    SELF\named_unit.dimensions : dimensional_exponents :=
                          dimensions_for_si_unit(name);
```

```
END_ENTITY; -- si_unit

ENTITY solid_model
  SUPERTYPE OF (manifold_solid_brep)
  SUBTYPE OF (geometric_representation_item);
END_ENTITY; -- solid_model

ENTITY spherical_surface
  SUBTYPE OF (elementary_surface);
    radius : positive_length_measure;
END_ENTITY; -- spherical_surface

ENTITY surface
  SUPERTYPE OF (ONEOF (elementary_surface,swept_surface,bounded_surface,
      offset_surface,surface_replica))
  SUBTYPE OF (geometric_representation_item);
END_ENTITY; -- surface

ENTITY surface_curve
  SUPERTYPE OF (ONEOF (intersection_curve,seam_curve) ANDOR
      bounded_surface_curve)
  SUBTYPE OF (curve);
    curve_3d              : curve;
    associated_geometry   : LIST [1:2] OF pcurve_or_surface;
    master_representation : preferred_surface_curve_representation;
  DERIVE
    basis_surface : SET [1:2] OF surface := get_basis_surface(SELF);
  WHERE
    WR1: (curve_3d.dim = 3);
    WR2: (('SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(associated_geometry
          [1])) OR (master_representation <> pcurve_s1));
    WR3: (('SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(associated_geometry
          [2])) OR (master_representation <> pcurve_s2));
    WR4: (NOT ('SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(curve_3d)));
END_ENTITY; -- surface_curve

ENTITY surface_of_linear_extrusion
  SUBTYPE OF (swept_surface);
    extrusion_axis : vector;
END_ENTITY; -- surface_of_linear_extrusion

ENTITY surface_of_revolution
  SUBTYPE OF (swept_surface);
    axis_position : axis1_placement;
  DERIVE
    axis_line : line := representation_item('') ||
                geometric_representation_item() || curve() || line(
                axis_position.location,representation_item('') ||
                geometric_representation_item() || vector(
                axis_position.z,1));
```



```
END_ENTITY; -- surface_of_revolution

ENTITY surface_replica
  SUBTYPE OF (surface);
    parent_surface : surface;
    transformation : cartesian_transformation_operator_3d;
  WHERE
    WR1: acyclic_surface_replica(SELF,parent_surface);
END_ENTITY; -- surface_replica

ENTITY swept_surface
  SUPERTYPE OF (ONEOF (surface_of_linear_extrusion,surface_of_revolution))
  SUBTYPE OF (surface);
    swept_curve : curve;
END_ENTITY; -- swept_surface

ENTITY thermodynamic_temperature_unit
  SUBTYPE OF (named_unit);
  WHERE
    WR1: ((SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
          named_unit.dimensions.mass_exponent = 0) AND (SELF\
          named_unit.dimensions.time_exponent = 0) AND (SELF\
          named_unit.dimensions.electric_current_exponent = 0) AND (
          SELF\named_unit.dimensions.
          thermodynamic_temperature_exponent = 1) AND (SELF\named_unit
          .dimensions.amount_of_substance_exponent = 0) AND (SELF\
          named_unit.dimensions.luminous_intensity_exponent = 0));
END_ENTITY; -- thermodynamic_temperature_unit

ENTITY time_unit
  SUBTYPE OF (named_unit);
  WHERE
    WR1: ((SELF\named_unit.dimensions.length_exponent = 0) AND (SELF\
          named_unit.dimensions.mass_exponent = 0) AND (SELF\
          named_unit.dimensions.time_exponent = 1) AND (SELF\
          named_unit.dimensions.electric_current_exponent = 0) AND (
          SELF\named_unit.dimensions.
          thermodynamic_temperature_exponent = 0) AND (SELF\named_unit
          .dimensions.amount_of_substance_exponent = 0) AND (SELF\
          named_unit.dimensions.luminous_intensity_exponent = 0));
END_ENTITY; -- time_unit

ENTITY topological_representation_item
  SUPERTYPE OF (ONEOF (vertex,edge,face_bound,face,connected_edge_set,
     connected_face_set,loop ANDOR path))
  SUBTYPE OF (representation_item);
END_ENTITY; -- topological_representation_item

ENTITY toroidal_surface
  SUBTYPE OF (elementary_surface);
```

```
    major_radius : positive_length_measure;
    minor_radius : positive_length_measure;
  END_ENTITY; -- toroidal_surface

  ENTITY trimmed_curve
   SUBTYPE OF (bounded_curve);
    basis_curve          : curve;
    trim_1               : SET [1:2] OF trimming_select;
    trim_2               : SET [1:2] OF trimming_select;
    sense_agreement      : BOOLEAN;
    master_representation : trimming_preference;
   WHERE
    WR1: ((HIINDEX(trim_1) = 1) OR (TYPEOF(trim_1[1]) <> TYPEOF(trim_1[2])));
    WR2: ((HIINDEX(trim_2) = 1) OR (TYPEOF(trim_2[1]) <> TYPEOF(trim_2[2])));
  END_ENTITY; -- trimmed_curve

  ENTITY uncertainty_measure_with_unit
   SUBTYPE OF (measure_with_unit);
    name       : label;
    description : OPTIONAL text;
   WHERE
    WR1: valid_measure_value(SELF\measure_with_unit.value_component);
  END_ENTITY; -- uncertainty_measure_with_unit

  ENTITY uniform_curve
   SUBTYPE OF (b_spline_curve);
  END_ENTITY; -- uniform_curve

  ENTITY uniform_surface
   SUBTYPE OF (b_spline_surface);
  END_ENTITY; -- uniform_surface

  ENTITY value_representation_item
   SUBTYPE OF (representation_item);
    value_component : measure_value;
   WHERE
    WR1: (SIZEOF(QUERY ( rep <* using_representations(SELF) | (NOT (
          'SHIP_STRUCTURES_SCHEMA.GLOBAL_UNIT_ASSIGNED_CONTEXT' IN
          TYPEOF(rep.context_of_items))) )) = 0);
  END_ENTITY; -- value_representation_item

  ENTITY vector
   SUBTYPE OF (geometric_representation_item);
    orientation : direction;
    magnitude   : length_measure;
   WHERE
    WR1: (magnitude >= 0);
  END_ENTITY; -- vector

  ENTITY versioned_action_request;
```



```
    id          : identifier;
    version     : label;
    purpose     : text;
    description : OPTIONAL text;
END_ENTITY; -- versioned_action_request

ENTITY vertex
  SUBTYPE OF (topological_representation_item);
END_ENTITY; -- vertex

ENTITY vertex_loop
  SUBTYPE OF (loop);
    loop_vertex : vertex;
END_ENTITY; -- vertex_loop

ENTITY vertex_point
  SUBTYPE OF (vertex, geometric_representation_item);
    vertex_geometry : point;
END_ENTITY; -- vertex_point

ENTITY week_of_year_and_day_date
  SUBTYPE OF (date);
    week_component : week_in_year_number;
    day_component  : OPTIONAL day_in_week_number;
END_ENTITY; -- week_of_year_and_day_date

RULE action_request_solution_connected_to_action FOR (
                    action_request_solution, action);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF action_request_solution := [];
    set_3   : SET OF action_method := [];
    t2_set  : SET OF action := [];
  END_LOCAL;
  t1_set := QUERY ( a <* action_request_solution | VALUE_IN(which_class(
     a),'change plan') );
  t2_set := QUERY ( b <* action | VALUE_IN(which_class(b),'change') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    set_3 := [];
    REPEAT j := 1 TO HIINDEX(t2_set) BY 1;
      set_3 := set_3 + [t2_set[j].chosen_method];
    END_REPEAT;
    violate := VALUE_IN(set_3,t1_set[i].method);
  END_REPEAT;

WHERE
  WR1: (NOT violate);

END_RULE; -- action_request_solution_connected_to_action
```

```
RULE action_request_solution_with_identification_assignment FOR (
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF action_request_solution := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  arg_list  : LIST OF STRING := ['definable object'];
  t2_set    : SET OF action_request_solution := [];
 END_LOCAL;
 REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = arg_list[j]) );
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
  violation := NOT (SIZEOF(t2_set) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- action_request_solution_with_identification_assignment

RULE action_with_identification_assignment FOR (
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF action := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  arg_list  : LIST OF STRING := ['definable object'];
  t2_set    : SET OF action := [];
 END_LOCAL;
 REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = arg_list[j]) );
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
```



```
    t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- action_with_identification_assignment

RULE alternative_version_relationship_has_mandatory_description FOR (
        identification_assignment_relationship);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF identification_assignment_relationship := [];
  END_LOCAL;
  t1_set := QUERY ( i <* identification_assignment_relationship |
     VALUE_IN(which_class(i),'alternative version relationship') );
  violate := SIZEOF(QUERY ( k <* t1_set | (NOT EXISTS(k.description)) ))
     > 0;

WHERE
  WR1: (NOT violate);

END_RULE; -- alternative_version_relationship_has_mandatory_description

RULE alternative_version_relationship_has_unique_versions FOR (
        identification_assignment_relationship);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF identification_assignment_relationship := [];
  END_LOCAL;
  t1_set := QUERY ( a <* identification_assignment_relationship |
     VALUE_IN(which_class(a),'alternative version relationship') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    violate := t1_set[i].relating_identification_assignment.assigned_id
       = t1_set[i].related_identification_assignment.assigned_id;
  END_REPEAT;

WHERE
  WR1: (NOT violate);

END_RULE; -- alternative_version_relationship_has_unique_versions

RULE alternative_version_relationship_versionable_object FOR (
        applied_identification_assignment);

  LOCAL
```

```
    violate  : LOGICAL := FALSE;
    violate1 : LOGICAL := FALSE;
    violate2 : LOGICAL := FALSE;
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_identification_assignment) BY 1
      WHILE NOT violate;
    violate1 := (SIZEOF(USEDIN(applied_identification_assignment[i],
       'SHIP_STRUCTURES_SCHEMA.IDENTIFICATION_ASSIGNMENT_RELATIONSHIP.'
       + 'RELATING_IDENTIFICATION_ASSIGNMENT')) > 0) AND (NOT VALUE_IN(
       which_class(applied_identification_assignment[i]),
       'versionable object'));
    violate2 := (SIZEOF(USEDIN(applied_identification_assignment[i],
       'SHIP_STRUCTURES_SCHEMA.IDENTIFICATION_ASSIGNMENT_RELATIONSHIP.'
       + 'RELATED_IDENTIFICATION_ASSIGNMENT')) > 0) AND (NOT VALUE_IN(
       which_class(applied_identification_assignment[i]),
       'versionable object'));
    violate := violate1 OR violate2;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- alternative_version_relationship_versionable_object

 RULE applied_approval_assignment_has_exactly_one_elements FOR (
         object_role, applied_approval_assignment);

 WHERE
  WR1: (SIZEOF(QUERY ( ass_inst <* applied_approval_assignment | (NOT ((
         ass_inst.role.name = 'subject') AND (SIZEOF(ass_inst.items) =
         1))) )) = 0);

 END_RULE; -- applied_approval_assignment_has_exactly_one_elements

 RULE applied_group_assignment_has_at_least_one_elements FOR (object_role,
         applied_group_assignment);

 WHERE
  WR1: (SIZEOF(QUERY ( ass_inst <* applied_group_assignment | (NOT ((
         ass_inst.role.name = 'approvals') AND (SIZEOF(ass_inst.items)
         >= 1) AND (SIZEOF(QUERY ( item <* ass_inst.items | (NOT (
         'SHIP_STRUCTURES_SCHEMA.APPROVAL' IN TYPEOF(item))) )) = 0))) ))
         = 0);

 END_RULE; -- applied_group_assignment_has_at_least_one_elements

 RULE approval_event_with_approval_date_time FOR (approval,
         approval_date_time);

  LOCAL
```



```
    violate : LOGICAL := FALSE;
    t1_set  : SET OF approval := [];
    t2_set  : SET OF approval_date_time := [];
  END_LOCAL;
  t1_set := QUERY ( i <* approval | VALUE_IN(which_class(i),
     'approval event') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPROVAL_DATE_TIME.' + 'DATED_APPROVAL'));
    violate := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violate);

END_RULE; -- approval_event_with_approval_date_time

RULE approval_event_with_approval_person_organization FOR (approval,
         approval_person_organization);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF approval := [];
    t2_set  : SET OF approval_person_organization := [];
  END_LOCAL;
  t1_set := QUERY ( i <* approval | VALUE_IN(which_class(i),
     'approval event') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPROVAL_PERSON_ORGANIZATION.' +
       'AUTHORIZED_APPROVAL'));
    violate := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violate);

END_RULE; -- approval_event_with_approval_person_organization

RULE approval_history_approves_same_definition FOR (
         applied_group_assignment, applied_approval_assignment);

  LOCAL
    violate : LOGICAL := FALSE;
    t3_set  : SET OF approval := [];
    t4_set  : SET OF group_item := [];
    t5_set  : SET OF applied_approval_assignment := [];
    t2_set  : SET OF applied_group_assignment := [];
  END_LOCAL;
  t2_set := QUERY ( a <* applied_group_assignment | VALUE_IN(
```

```
      which_class(a.assigned_group),'approval history') );
   t3_set := QUERY ( b <* t2_set[1].items | (
      'SHIP_STRUCTURES_SCHEMA.APPROVAL' IN TYPEOF(b)) );
   t4_set := QUERY ( b <* t2_set[1].items | VALUE_IN(which_class(b),
      'DEFINITION') );
   violate := NOT (SIZEOF(t4_set) = 1);
   REPEAT i := 1 TO HIINDEX(t3_set) BY 1 WHILE NOT violate;
    t5_set := QUERY ( a <* applied_approval_assignment | ((a.
       assigned_approval = t3_set[i]) AND (NOT VALUE_IN(a.items,t4_set[1]))) );
    violate := SIZEOF(t5_set) > 0;
   END_REPEAT;

  WHERE
   WR1: (NOT violate);
   WR2: (SIZEOF(t4_set) = 1);

  END_RULE; -- approval_history_approves_same_definition

  RULE approval_history_contains_approval_elements FOR (object_role,
          applied_group_assignment);

   LOCAL
    t3_set         : SET OF approval := [];
    gr_ass_set     : SET OF applied_group_assignment := [];
    violation      : LOGICAL := FALSE;
    valid_items_set : SET OF approval_item := [];
    t1_set         : SET OF applied_approval_assignment := [];
    first_runs     : LOGICAL := TRUE;
   END_LOCAL;
   gr_ass_set := QUERY ( aga <* applied_group_assignment | ((aga.role.
      name = 'approvals') AND ('approval history' IN which_class(aga.
      assigned_group))) );
   REPEAT i := 1 TO HIINDEX(gr_ass_set) BY 1 WHILE NOT violation;
    t3_set := QUERY ( aga_item <* gr_ass_set[i].items | (
       'SHIP_STRUCTURES_SCHEMA.APPROVAL' IN TYPEOF(aga_item)) );
    REPEAT j := 1 TO HIINDEX(t3_set) BY 1;
     t1_set := QUERY ( aga <* bag_to_set(USEDIN(t3_set[j],
        'SHIP_STRUCTURES_SCHEMA.' +
        'APPROVAL_ASSIGNMENT.ASSIGNED_APPROVAL')) | (aga.role.name =
        'subject') );
     REPEAT k := 1 TO HIINDEX(t1_set) BY 1;
      IF first_runs AND (SIZEOF(t1_set[k].items) > 0) THEN
       valid_items_set := t1_set[k].items;
       first_runs := FALSE;
      ELSE
       REPEAT l := 1 TO HIINDEX(t1_set[k].items) BY 1;
        violation := t1_set[k].items[l] :<>: valid_items_set[l];
       END_REPEAT;
      END_IF;
     END_REPEAT;
```



```
    END_REPEAT;
    first_runs := TRUE;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- approval_history_contains_approval_elements

RULE approval_history_has_at_least_one_member FOR (group,
        applied_group_assignment);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF group := [];
    t2_set  : SET OF applied_group_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( i <* group | VALUE_IN(which_class(i),
     'approval history') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    t2_set := QUERY ( a <* applied_group_assignment | (a.assigned_group
       = t1_set[i]) );
    violate := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violate);

END_RULE; -- approval_history_has_at_least_one_member

RULE approvals_references_approval_history FOR (object_role,
        applied_group_assignment, group);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* group | VALUE_IN(which_class(a),
     'approval history') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_group_assignment | (NOT ((b.
       assigned_group = t1_set[i]) AND (b.role.name = 'approvals'))) );
    violate := SIZEOF(a_set) > 0;
  END_REPEAT;

WHERE
  WR1: (NOT violate);

END_RULE; -- approvals_references_approval_history
```

```
RULE assembly_relationship_has_product_definition FOR (group,
        applied_classification_assignment);

 LOCAL
  cl_ass      : SET OF applied_classification_assignment := [];
  clfied_inst : SET OF product_definition_relationship := [];
 END_LOCAL;
 cl_ass := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'assembly relationship') );
 REPEAT i := 1 TO HIINDEX(cl_ass) BY 1;
  clfied_inst := clfied_inst + cl_ass[i].items;
 END_REPEAT;

WHERE
 WR1: ((SIZEOF(QUERY ( pdr <* clfied_inst | (NOT (SIZEOF(which_class(
        pdr.relating_product_definition)) > 0)) )) - SIZEOF(
        QUERY ( pdr <* clfied_inst | (NOT (SIZEOF(which_class(pdr.
        related_product_definition)) > 0)) ))) = 0);

END_RULE; -- assembly_relationship_has_product_definition

RULE author_for_change_plan FOR (
        applied_person_and_organization_assignment,
        person_and_organization_role, action_request_solution);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action_request_solution := [];
  a_set   : SET OF applied_person_and_organization_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* action_request_solution | VALUE_IN(which_class(
    a),'change plan') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_person_and_organization_assignment | (
     VALUE_IN(b.items,t1_set[i]) AND (b.role.name = 'author')) );
  violate := SIZEOF(a_set) <> 1;
 END_REPEAT;

WHERE
 WR1: (NOT violate);

END_RULE; -- author_for_change_plan

RULE author_for_change_realisation FOR (
        applied_person_and_organization_assignment,
        person_and_organization_role, executed_action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF executed_action := [];
```



```
  a_set   : SET OF applied_person_and_organization_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* executed_action | VALUE_IN(which_class(a),
    'change realisation') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_person_and_organization_assignment | (
     VALUE_IN(b.items,t1_set[i]) AND (b.role.name = 'author')) );
  violate := SIZEOF(a_set) <> 1;
 END_REPEAT;

WHERE
 WR1: (NOT violate);

END_RULE; -- author_for_change_realisation

RULE author_for_change_request FOR (
        applied_person_and_organization_assignment,
        person_and_organization_role, versioned_action_request);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF versioned_action_request := [];
  a_set   : SET OF applied_person_and_organization_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* versioned_action_request | VALUE_IN(
    which_class(a),'change request') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_person_and_organization_assignment | (
     VALUE_IN(b.items,t1_set[i]) AND (b.role.name = 'author')) );
  violate := SIZEOF(a_set) <> 1;
 END_REPEAT;

WHERE
 WR1: (NOT violate);

END_RULE; -- author_for_change_request

RULE caused_by_for_check FOR (applied_person_and_organization_assignment,
        person_and_organization_role, action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action := [];
  a_set   : SET OF applied_person_and_organization_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),'check') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_person_and_organization_assignment | (
     VALUE_IN(b.items,t1_set[i]) AND (b.role.name = 'caused by')) );
  violate := SIZEOF(a_set) <> 1;
```



```
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- caused_by_for_check

 RULE caused_by_for_envisaged_version_creation FOR (
          applied_person_and_organization_assignment,
          person_and_organization_role, action);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),
     'envisaged version creation') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_person_and_organization_assignment | (
       VALUE_IN(b.items,t1_set[i]) AND (b.role.name = 'caused by')) );
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- caused_by_for_envisaged_version_creation

 RULE caused_by_for_version_creation FOR (
          applied_person_and_organization_assignment,
          person_and_organization_role, action);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_person_and_organization_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),
     'version creation') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_person_and_organization_assignment | (
       VALUE_IN(b.items,t1_set[i]) AND (b.role.name = 'caused by')) );
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- caused_by_for_version_creation
```



```
RULE caused_by_for_version_deletion FOR (
        applied_person_and_organization_assignment,
        person_and_organization_role, action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action := [];
  a_set   : SET OF applied_person_and_organization_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),
    'version deletion') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_person_and_organization_assignment | (
     VALUE_IN(b.items,t1_set[i]) AND (b.role.name = 'caused by')) );
  violate := SIZEOF(a_set) <> 1;
 END_REPEAT;

WHERE
 WR1: (NOT violate);

END_RULE; -- caused_by_for_version_deletion

RULE caused_by_for_version_modification FOR (
        applied_person_and_organization_assignment,
        person_and_organization_role, action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action := [];
  a_set   : SET OF applied_person_and_organization_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),
    'version modification') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_person_and_organization_assignment | (
     VALUE_IN(b.items,t1_set[i]) AND (b.role.name = 'caused by')) );
  violate := SIZEOF(a_set) <> 1;
 END_REPEAT;

WHERE
 WR1: (NOT violate);

END_RULE; -- caused_by_for_version_modification

RULE caused_when_for_check FOR (applied_date_and_time_assignment,
        date_time_role, action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action := [];
```

```
    a_set   : SET OF applied_date_and_time_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),'check') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_date_and_time_assignment | (VALUE_IN(b
       .items,t1_set[i]) AND (b.role.name = 'caused when')) );
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- caused_when_for_check

 RULE caused_when_for_envisaged_version_creation FOR (
          applied_date_and_time_assignment, date_time_role, action);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_date_and_time_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),
     'envisaged version creation') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_date_and_time_assignment | (VALUE_IN(b
       .items,t1_set[i]) AND (b.role.name = 'caused when')) );
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- caused_when_for_envisaged_version_creation

 RULE caused_when_for_version_creation FOR (
          applied_date_and_time_assignment, date_time_role, action);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF action := [];
    a_set   : SET OF applied_date_and_time_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),
     'version creation') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_date_and_time_assignment | (VALUE_IN(b
       .items,t1_set[i]) AND (b.role.name = 'caused when')) );
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
```



```
WHERE
 WR1: (NOT violate);

END_RULE; -- caused_when_for_version_creation

RULE caused_when_for_version_deletion FOR (
        applied_date_and_time_assignment, date_time_role, action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action := [];
  a_set   : SET OF applied_date_and_time_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),
    'version deletion') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_date_and_time_assignment | (VALUE_IN(b
     .items,t1_set[i]) AND (b.role.name = 'caused when')) );
  violate := SIZEOF(a_set) <> 1;
 END_REPEAT;

WHERE
 WR1: (NOT violate);

END_RULE; -- caused_when_for_version_deletion

RULE caused_when_for_version_modification FOR (
        applied_date_and_time_assignment, date_time_role, action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action := [];
  a_set   : SET OF applied_date_and_time_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* action | VALUE_IN(which_class(a),
    'version modification') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_date_and_time_assignment | (VALUE_IN(b
     .items,t1_set[i]) AND (b.role.name = 'caused when')) );
  violate := SIZEOF(a_set) <> 1;
 END_REPEAT;

WHERE
 WR1: (NOT violate);

END_RULE; -- caused_when_for_version_modification

RULE centre_location_compound_representation_has_specified_name FOR (
        compound_representation_item,
        applied_classification_assignment);
```

```
  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF compound_representation_item := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    arg_list  : LIST OF STRING := ['longitudinal location',
                'transversal location','vertical location'];
    t2_set    : SET OF representation_item := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'centre location') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      t2_set := t1_set[i].item_element;
      violation := SIZEOF(QUERY ( items <* t2_set | (items.name =
         arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- centre_location_compound_representation_has_specified_name

 RULE change_impact_with_versionable_object_change_event FOR (
          applied_action_request_assignment, object_role, action);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF applied_action_request_assignment := [];
    a_set   : SET OF action := [];
  END_LOCAL;
  t1_set := QUERY ( b <* applied_action_request_assignment | (b.role.
     name = 'change impact') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* t1_set[i].items | ((
       'SHIP_STRUCTURES_SCHEMA.ACTION' IN TYPEOF(b)) AND VALUE_IN(
       which_class(b),'versionable object change event')) );
    violate := SIZEOF(a_set) = 0;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- change_impact_with_versionable_object_change_event
```



```
RULE change_plan_has_mandatory_attribute_description FOR (
        action_request_solution);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action_request_solution := [];
 END_LOCAL;
 t1_set := QUERY ( i <* action_request_solution | VALUE_IN(which_class(
    i),'change plan') );
 violate := SIZEOF(QUERY ( k <* t1_set | (NOT EXISTS(k.description)) ))
    > 0;

WHERE
 WR1: (NOT violate);

END_RULE; -- change_plan_has_mandatory_attribute_description

RULE class_and_statutory_designation_has_properties FOR (
        product_definition, property_definition_representation, group,
        applied_classification_assignment);

 LOCAL
  t3_set    : SET OF property_definition := [];
  violation : LOGICAL := FALSE;
  t4_set    : SET OF product_definition := [];
  t1_set    : SET OF product_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'class and statutory designation') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 t2_set := QUERY ( i <* property_definition_representation | (i.name =
    'class and statutory designation') );
 REPEAT i := 1 TO HIINDEX(t2_set) BY 1;
  t3_set := t3_set + t2_set[i].definition;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t3_set) BY 1;
  t4_set := t4_set + t3_set[i].definition;
 END_REPEAT;
 violation := t1_set <> t4_set;

WHERE
 WR1: (NOT violation);

END_RULE; -- class_and_statutory_designation_has_properties
```

```
RULE class_notation_with_named_representation_items FOR (representation,
        representation_item, property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['class notations hull',
              'class notations machinery'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'class notation') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) < 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- class_notation_with_named_representation_items

RULE class_parameters_has_properties FOR (product_definition,
        property_definition_representation, group,
        applied_classification_assignment);

 LOCAL
  t3_set    : SET OF property_definition := [];
  violation : LOGICAL := FALSE;
  t4_set    : SET OF product_definition := [];
  t1_set    : SET OF product_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'class parameters') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 t2_set := QUERY ( i <* property_definition_representation | (i.name =
    'class parameters') );
 REPEAT i := 1 TO HIINDEX(t2_set) BY 1;
  t3_set := t3_set + t2_set[i].definition;
 END_REPEAT;
```



```
  REPEAT i := 1 TO HIINDEX(t3_set) BY 1;
    t4_set := t4_set + t3_set[i].definition;
  END_REPEAT;
  violation := t1_set <> t4_set;

WHERE
  WR1: (NOT violation);

END_RULE; -- class_parameters_has_properties

RULE compatible_dimension FOR (cartesian_point, direction,
        representation_context, geometric_representation_context);

WHERE
  WR1: (SIZEOF(QUERY ( x <* cartesian_point | (SIZEOF(QUERY ( y <*
        geometric_representation_context | (item_in_context(x,y) AND (
        HIINDEX(x.coordinates) <> y.coordinate_space_dimension)) )) >
        0) )) = 0);
  WR2: (SIZEOF(QUERY ( x <* direction | (SIZEOF(QUERY ( y <*
        geometric_representation_context | (item_in_context(x,y) AND (
        HIINDEX(x.direction_ratios) <> y.coordinate_space_dimension)) ))
        > 0) )) = 0);

END_RULE; -- compatible_dimension

RULE corrugated_structure_design_definition_has_at_least_one_class FOR (
        product_definition_shape, shape_aspect, group,
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'corrugated structure design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY ( t2_inst <* t2_set | ('border' IN
       which_class(t2_inst)) )) < 1;
  END_REPEAT;

  WHERE
```

```
  WR1: (NOT violation);

END_RULE; -- corrugated_structure_design_definition_has_at_least_one_class

RULE corrugated_structure_design_definition_has_at_most_one_class FOR (
        product_definition_shape, shape_aspect, group,
        applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'corrugated structure design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
   violation := SIZEOF(QUERY ( t2_inst <* t2_set | ('moulded surface'
      IN which_class(t2_inst)) )) > 1;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- corrugated_structure_design_definition_has_at_most_one_class

RULE date_time_for_change_realisation FOR (
        applied_date_and_time_assignment, date_time_role,
        executed_action);

  LOCAL
   violate : LOGICAL := FALSE;
   t1_set  : SET OF executed_action := [];
   a_set   : SET OF applied_date_and_time_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* executed_action | VALUE_IN(which_class(a),
     'change realisation') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
   a_set := QUERY ( b <* applied_date_and_time_assignment | (VALUE_IN(b
      .items,t1_set[i]) AND (b.role.name = 'date time')) );
   violate := SIZEOF(a_set) <> 1;
  END_REPEAT;
```



```
WHERE
 WR1: (NOT violate);

END_RULE; -- date_time_for_change_realisation

RULE date_time_for_change_request FOR (applied_date_and_time_assignment,
        date_time_role, versioned_action_request);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF versioned_action_request := [];
  a_set   : SET OF applied_date_and_time_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* versioned_action_request | VALUE_IN(
    which_class(a),'change request') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_date_and_time_assignment | (VALUE_IN(b
     .items,t1_set[i]) AND (b.role.name = 'date time')) );
  violate := SIZEOF(a_set) <> 1;
 END_REPEAT;

WHERE
 WR1: (NOT violate);

END_RULE; -- date_time_for_change_request

RULE define_maximum_permissible_still_water_bending_moment_values FOR (
        product_definition, property_definition_representation,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name =
    'maximum permissible still water bending moment') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'maximum permissible still water bending  moment values') )) = 1);
 END_REPEAT;
```

```
 WHERE
  WR1: (NOT violation);

 END_RULE; -- define_maximum_permissible_still_water_bending_moment_values

 RULE define_maximum_permissible_still_water_shear_force_values FOR (
         product_definition, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'maximum permissible still water shear force') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'maximum permissible still water shear force values') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- define_maximum_permissible_still_water_shear_force_values

 RULE definition_for_design_still_water_bending_moment_parameters FOR (
         product_definition, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'design still water bending moment') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
```



```
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'design still water bending moment parameters') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- definition_for_design_still_water_bending_moment_parameters

RULE definition_for_design_vertical_wave_bending_moment_parameters FOR (
          product_definition, property_definition_representation,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'design vertical wave bending moment') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'design vertical wave bending moment parameters') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- definition_for_design_vertical_wave_bending_moment_parameters

RULE definition_for_design_vertical_wave_shear_force_parameters FOR (
          product_definition, property_definition_representation,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
```

```
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'design vertical wave shear force') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'design vertical wave shear force parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- definition_for_design_vertical_wave_shear_force_parameters

 RULE document_has_at_least_one_references FOR (document,
         document_representation_type);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF document := [];
    t2_set  : SET OF document_representation_type := [];
  END_LOCAL;
  t1_set := QUERY ( i <* document | VALUE_IN(which_class(i),'document') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.DOCUMENT_REPRESENTATION_TYPE.' +
      'REPRESENTED_DOCUMENT'));
   violate := SIZEOF(t2_set) < 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- document_has_at_least_one_references

 RULE document_has_exactly_one_author FOR (document);

  LOCAL
    violate : LOGICAL := FALSE;
    bag_3   : BAG OF applied_organization_assignment := [];
    bag_2   : BAG OF applied_person_and_organization_assignment := [];
    bag_1   : BAG OF applied_person_assignment := [];
  END_LOCAL;
```



```
  REPEAT i := 1 TO SIZEOF(document) BY 1 WHILE NOT violate;
    bag_1 := USEDIN(document[i],'SHIP_STRUCTURES_SCHEMA.' +
       'APPLIED_PERSON_ASSIGNMENT.ITEMS');
    bag_2 := USEDIN(document[i],'SHIP_STRUCTURES_SCHEMA.' +
       'APPLIED_PERSON_AND_ORGANIZATION_ASSIGNMENT.ITEMS');
    bag_3 := USEDIN(document[i],'SHIP_STRUCTURES_SCHEMA.' +
       'APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS');
    violate := NOT ((SIZEOF(bag_1) + SIZEOF(bag_2) + SIZEOF(bag_3)) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- document_has_exactly_one_author

 RULE document_reference_has_at_least_one_references FOR (document,
         document_usage_constraint);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF document := [];
    t2_set  : SET OF document_usage_constraint := [];
  END_LOCAL;
  t1_set := QUERY ( i <* document | VALUE_IN(which_class(i),
     'document reference') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.DOCUMENT_USAGE_CONSTRAINT.SOURCE'));
    violate := SIZEOF(t2_set) < 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- document_reference_has_at_least_one_references

 RULE document_reference_reference_with_address_with_references FOR (
         document, document_usage_constraint);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF document := [];
    t2_set  : SET OF document_usage_constraint := [];
  END_LOCAL;
  t1_set := QUERY ( i <* document | VALUE_IN(which_class(i),
     'document reference reference with address') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.DOCUMENT_USAGE_CONSTRAINT.SOURCE'));
    violate := SIZEOF(t2_set) < 1;
```

```
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- document_reference_reference_with_address_with_references

 RULE document_reference_with_address_has_at_least_one_references FOR (
        document, applied_external_identification_assignment);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF document := [];
    t2_set  : SET OF applied_external_identification_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( i <* document | VALUE_IN(which_class(i),
     'document reference with address') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.APPLIED_EXTERNAL_IDENTIFIC
ATION_ASSIGNMENT.ITEMS'));
    violate := SIZEOF(t2_set) < 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- document_reference_with_address_has_at_least_one_references

 RULE envisaged_version_creation_has_mandatory_attribute_description FOR (
        action);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF action := [];
  END_LOCAL;
  t1_set := QUERY ( i <* action | VALUE_IN(which_class(i),
     'envisaged version creation') );
  violate := SIZEOF(QUERY ( k <* t1_set | (NOT EXISTS(k.description)) ))
     > 0;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- envisaged_version_creation_has_mandatory_attribute_description

 RULE executed_action_with_identification_assignment FOR (
        applied_classification_assignment);

  LOCAL
```



```
    violation : LOGICAL := FALSE;
    t1_set    : SET OF executed_action := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    arg_list  : LIST OF STRING := ['definable object'];
    t2_set    : SET OF executed_action := [];
   END_LOCAL;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = arg_list[j]) );
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(t2_set) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- executed_action_with_identification_assignment

  RULE externally_defined_class_with_known_source_requirement FOR (
          externally_defined_class);

  WHERE
   WR1: (SIZEOF(QUERY ( edc <* externally_defined_class | ((
          'SHIP_STRUCTURES_SCHEMA.KNOWN_SOURCE' IN TYPEOF(edc.source))
          AND (SIZEOF(QUERY ( aoa <*
USEDIN(edc,'SHIP_STRUCTURES_SCHEMA.APPLIED_ORGANIZATION_ASSIGNMENT.ITEM
S')
          | (aoa.role.name = 'library supplier') )) = 0)) )) = 0);

  END_RULE; -- externally_defined_class_with_known_source_requirement

  RULE flare_area_parameters_has_exactly_two_representation_items FOR (
          representation, property_definition_representation);

   LOCAL
    violation : LOGICAL := FALSE;
    reps      : BAG OF representation := [];
   END_LOCAL;
   reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
```

```
      'flare area parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    violation := SIZEOF(reps[i].items) > 2;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- flare_area_parameters_has_exactly_two_representation_items

 RULE freeboard_characteristics_has_properties FOR (product_definition,
         property_definition, property_definition_representation,
         applied_classification_assignment);

  LOCAL
    t3_set    : SET OF property_definition := [];
    violation : LOGICAL := FALSE;
    t4_set    : SET OF product_definition := [];
    t1_set    : SET OF product_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'freeboard characteristics') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  t2_set := QUERY ( i <* property_definition_representation | (i.name =
     'freeboard characteristics') );
  REPEAT i := 1 TO HIINDEX(t2_set) BY 1;
    t3_set := t3_set + t2_set[i].definition;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t3_set) BY 1;
    t4_set := t4_set + t3_set[i].definition;
  END_REPEAT;
  violation := t1_set <> t4_set;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- freeboard_characteristics_has_properties

 RULE global_axis_placement_has_properties FOR (
         property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
    t3_set    : SET OF property_definition := [];
```



```
    violation : LOGICAL := FALSE;
    t4_set    : SET OF product_definition := [];
    t1_set    : SET OF product_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'global axis placement') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  t2_set := QUERY ( i <* property_definition_representation | (i.name =
     'global axis placement') );
  REPEAT i := 1 TO HIINDEX(t2_set) BY 1;
    t3_set := t3_set + t2_set[i].definition;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t3_set) BY 1;
    t4_set := t4_set + t3_set[i].definition;
  END_REPEAT;
  violation := t1_set <> t4_set;

WHERE
  WR1: (NOT violation);

END_RULE; -- global_axis_placement_has_properties

RULE global_id_is_unique FOR (applied_identification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    set_1     : SET OF applied_identification_assignment := [];
    bag_2     : BAG OF STRING := [];
  END_LOCAL;
  set_1 := QUERY ( i <* applied_identification_assignment | (i.role.name
     = 'globally unambiguous identifier') );
  REPEAT i := 1 TO HIINDEX(set_1) BY 1;
    bag_2 := bag_2 + [set_1[i].assigned_id];
  END_REPEAT;
  violation := SIZEOF(QUERY ( i <* set_1 | (NOT (i.assigned_id LIKE
     '*:*')) )) = 0;

WHERE
  WR1: VALUE_UNIQUE(bag_2);
  WR2: (NOT violation);

END_RULE; -- global_id_is_unique

RULE group_with_identification_assignment FOR (
```

```
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF group := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    arg_list  : LIST OF STRING := [
                'structural cutout boundary relationship'];
    t2_set    : SET OF group := [];
  END_LOCAL;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = arg_list[j]) );
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- group_with_identification_assignment

 RULE homogeneous_ship_material_property_has_named_references FOR (
        object_role, group, applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF property_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF applied_document_reference := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'homogeneous_ship material property') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_DOCUMENT_REFERENCE.ITEMS'));
    violation := SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.name =
```



```
        'material reference') )) > 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- homogeneous_ship_material_property_has_named_references

RULE hull_cross_section_contains_specified_classes FOR (
          applied_group_assignment, group,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    cl_ass    : SET OF applied_classification_assignment := [];
    gr_ass    : SET OF applied_group_assignment := [];
    pd        : SET OF product_definition := [];
    groups    : SET OF group := [];
  END_LOCAL;
  cl_ass := QUERY ( i <* applied_classification_assignment | ((i.role.
     name = 'class membership') AND (i.assigned_class.name =
     'hull cross section')) );
  REPEAT i := 1 TO HIINDEX(cl_ass) BY 1;
    REPEAT j := 1 TO HIINDEX(cl_ass[i].items) BY 1;
      pd := pd + cl_ass[i].items[j];
    END_REPEAT;
  END_REPEAT;
  gr_ass := QUERY ( i <* applied_group_assignment | (i.role.name =
     'equivalence') );
  REPEAT i := 1 TO HIINDEX(gr_ass) BY 1;
    REPEAT j := 1 TO HIINDEX(gr_ass[i].items) BY 1;
      IF gr_ass[i].items[j] IN pd THEN
        groups := groups + gr_ass[i].assigned_group;
      END_IF;
    END_REPEAT;
  END_REPEAT;
  gr_ass := QUERY ( i <* applied_group_assignment | (((SIZEOF(i.items)
     <> 0) AND (i.role.name = 'item structure')) AND (i.assigned_group
     IN groups)) );
  REPEAT i := 1 TO HIINDEX(gr_ass) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( inst <* gr_ass[i].items | ((
       'compartment' IN which_class(inst)) AND ('structural system' IN
       which_class(inst))) )) > 0;
  END_REPEAT;

 WHERE
  WR1: NOT violation;

 END_RULE; -- hull_cross_section_contains_specified_classes
```

```
 RULE hull_cross_section_design_definition_for_representation_items FOR (
        property_definition_representation, representation);

 LOCAL
  violation : LOGICAL := FALSE;
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'hull cross section design definition parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  violation := SIZEOF(reps[i].items) > 1;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- hull_cross_section_design_definition_for_representation_items

RULE hull_cross_section_design_has_at_least_one_revision_with_context FOR (
        group, applied_classification_assignment);

 LOCAL
  gr_ass_set : SET OF applied_group_assignment := [];
  violation  : LOGICAL := FALSE;
  t1_set     : SET OF product_definition_shape := [];
  c_a_set    : SET OF applied_classification_assignment := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'hull cross section design definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  gr_ass_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS'));
  violation := SIZEOF(QUERY ( aga <* gr_ass_set | ((aga.role.name =
     'members') AND ('revision with context' IN which_class(aga.
     assigned_group)))) )) < 1;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- hull_cross_section_design_has_at_least_one_revision_with_context
```



```
RULE hull_cross_section_design_has_at_most_one_revision_with_context FOR (
        group, applied_classification_assignment);

 LOCAL
  gr_ass_set : SET OF applied_group_assignment := [];
  violation  : LOGICAL := FALSE;
  t1_set     : SET OF product_definition_shape := [];
  c_a_set    : SET OF applied_classification_assignment := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'hull cross section design definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  gr_ass_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS'));
  violation := SIZEOF(QUERY ( aga <* gr_ass_set | ((aga.role.name =
     'members') AND ('revision with context' IN which_class(aga.
     assigned_group))) )) > 1;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- hull_cross_section_design_has_at_most_one_revision_with_context

RULE hull_cross_section_has_at_least_one_revision_with_context FOR (
        object_role, group, applied_classification_assignment);

 LOCAL
  gr_ass_set : SET OF applied_group_assignment := [];
  violation  : LOGICAL := FALSE;
  t1_set     : SET OF product_definition := [];
  c_a_set    : SET OF applied_classification_assignment := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'hull cross section') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  gr_ass_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS'));
  violation := SIZEOF(QUERY ( aga <* gr_ass_set | ((aga.role.name =
     'context of revision') AND ('revision with context' IN
```

```
        which_class(aga.assigned_group))) )) < 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- hull_cross_section_has_at_least_one_revision_with_context

 RULE hull_cross_section_optionally_contains_specified_classes FOR (group,
         applied_classification_assignment, applied_group_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    cl_ass    : SET OF applied_classification_assignment := [];
    gr_ass    : SET OF applied_group_assignment := [];
    pd        : SET OF product_definition := [];
    groups    : SET OF group := [];
  END_LOCAL;
  cl_ass := QUERY ( i <* applied_classification_assignment | ((i.role.
     name = 'class membership') AND (i.assigned_class.name =
     'hull cross section')) );
  REPEAT i := 1 TO HIINDEX(cl_ass) BY 1;
   pd := pd + cl_ass[i].items;
  END_REPEAT;
  gr_ass := QUERY ( i <* applied_group_assignment | (i.role.name =
     'equivalence') );
  REPEAT i := 1 TO HIINDEX(gr_ass) BY 1;
   REPEAT j := 1 TO HIINDEX(gr_ass[i].items) BY 1;
     IF gr_ass[i].items[j] IN pd THEN
       groups := groups + gr_ass[i].assigned_group;
     END_IF;
   END_REPEAT;
  END_REPEAT;
  gr_ass := QUERY ( i <* applied_group_assignment | ((SIZEOF(i.items) <>
      0) AND (i.role.name = 'item structure') AND (i.assigned_group IN
     groups)) );
  REPEAT i := 1 TO HIINDEX(gr_ass) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( inst <* gr_ass[i].items | (
      'structural system' IN which_class(inst)) )) = 0;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- hull_cross_section_optionally_contains_specified_classes

 RULE hull_cross_section_result_for_representation_item FOR (
         representation, representation_item, group,
         applied_classification_assignment);
```



```
  LOCAL
    violation      : LOGICAL := FALSE;
    c_a_set        : SET OF applied_classification_assignment := [];
    arg_list       : LIST OF STRING := [
                     'actual first moment horizontal',
                     'actual first moment vertical',
                     'actual inertia cross','actual inertia horizontal',
                     'actual inertia torsion','actual inertia vertical',
                     'actual neutral axis position horizontal',
                     'actual neutral axis position vertical',
                     'actual section area',
                     'actual section modulus bottom',
                     'actual section modulus coaming',
                     'actual section modulus deck',
                     'actual stress bottom','actual stress coaming',
                     'actual stress deck',
                     'actual twist centre horizontal',
                     'actual twist centre vertical',
                     'direction largest inertia',
                     'direction least inertia','max moment',
                     'max shear force','meter weight','min moment',
                     'min shear force',
                     'required section modulus bottom',
                     'required section modulus coaming',
                     'required section modulus deck',
                     'required stress bottom','required stress coaming',
                     'required stress deck'];
    classed_rep_set : SET OF representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'hull cross section result') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      classed_rep_set := classed_rep_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(classed_rep_set) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* classed_rep_set[i].items
          | (rep_item.name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
   WR1: (NOT violation);

 END_RULE; -- hull_cross_section_result_for_representation_item

 RULE identification_role_optional_attribute_description_required FOR (
         identification_role);
```

```
 WHERE
  WR1: (SIZEOF(QUERY ( i <* identification_role | ((i.name =
         'external reference') AND (NOT EXISTS(i.description))) )) = 0);

 END_RULE; -- identification_role_optional_attribute_description_required

 RULE initiator_for_change_request FOR (
         applied_person_and_organization_assignment,
         person_and_organization_role, versioned_action_request);

  LOCAL
   violate : LOGICAL := FALSE;
   t1_set  : SET OF versioned_action_request := [];
   a_set   : SET OF applied_person_and_organization_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* versioned_action_request | VALUE_IN(
     which_class(a),'change request') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
   a_set := QUERY ( b <* applied_person_and_organization_assignment | (
      VALUE_IN(b.items,t1_set[i]) AND (b.role.name = 'initiator')) );
   violate := SIZEOF(a_set) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- initiator_for_change_request

 RULE library_manufacturing_definition_for_class_references FOR (
         shape_aspect, group, applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'library manufacturing definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
   violation := SIZEOF(QUERY ( t2_inst <* t2_set | ('source' IN
      which_class(t2_inst)) )) < 1;
  END_REPEAT;
```



```
WHERE
 WR1: (NOT violation);

END_RULE; -- library_manufacturing_definition_for_class_references

RULE lightship_definition_has_properties FOR (product_definition,
        property_definition, property_definition_representation,
        applied_classification_assignment);

 LOCAL
  t3_set    : SET OF property_definition := [];
  violation : LOGICAL := FALSE;
  t4_set    : SET OF product_definition := [];
  t1_set    : SET OF product_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'lightship_definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 t2_set := QUERY ( i <* property_definition_representation | (i.name =
    'lightship_definition') );
 REPEAT i := 1 TO HIINDEX(t2_set) BY 1;
  t3_set := t3_set + t2_set[i].definition;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t3_set) BY 1;
  t4_set := t4_set + t3_set[i].definition;
 END_REPEAT;
 violation := t1_set <> t4_set;

WHERE
 WR1: (NOT violation);

END_RULE; -- lightship_definition_has_properties

RULE mandatory_entity_type_for_external_instance_reference FOR (
        external_source, external_source_relationship);

 LOCAL
  violate : LOGICAL := FALSE;
  bag_1   : BAG OF external_source := [];
 END_LOCAL;
 bag_1 := QUERY ( a <* external_source | (a.description = 'schema') );
 REPEAT i := 1 TO SIZEOF(bag_1) BY 1 WHILE NOT violate;
  violate := SIZEOF(QUERY ( a <* external_source_relationship | ((a.
     related_source :=: bag_1[i]) AND (a.relating_source.description =
```

```
        'entity type')) )) = 0;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- mandatory_entity_type_for_external_instance_reference

 RULE maximum_of_one_representation_items_for_class_library_definition FOR (
          representation, applied_classification_assignment,
          classification_role, group);

  LOCAL
    var_representation : BAG OF representation := [];
  END_LOCAL;
  var_representation := QUERY ( temp1 <* representation | (SIZEOF(
     QUERY ( temp2 <* bag_to_set(USEDIN(temp1,
     'SHIP_STRUCTURES_SCHEMA.applied_classification_assignment.ITEMS'))
      | ((temp2\classification_assignment.role.name = 'class membership')
     AND (temp2\classification_assignment.assigned_class\group.name =
     'library definition')) )) > 0) );

 WHERE
  WR1: (SIZEOF(QUERY ( temp3 <* var_representation | (SIZEOF(temp3.items)
         > 1) )) = 0);

 END_RULE; -- maximum_of_one_representation_items_for_class_library_definition

 RULE members_is_referenced_by_at_least_one_revision FOR (
          applied_group_assignment, group);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* group | VALUE_IN(which_class(a),'members') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_group_assignment | ((b.assigned_group
       = t1_set[i]) AND (b.role.name = 'revision')) );
    violate := SIZEOF(a_set) < 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- members_is_referenced_by_at_least_one_revision

 RULE no_approvals_except_in_approval_history FOR (approval);
```



```
  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF approval := [];
    t2_set  : SET OF applied_group_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* approval | VALUE_IN(which_class(a),
     'approval event') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS'));
    violate := SIZEOF(t2_set) = 0;
    REPEAT k := 1 TO HIINDEX(t2_set) BY 1 WHILE NOT violate;
      violate := NOT VALUE_IN(which_class(t2_set[k].assigned_group),
         'approval history');
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violate);

END_RULE; -- no_approvals_except_in_approval_history

RULE panel_system_design_definition_has_at_least_one_class FOR (
         shape_aspect, group, applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'panel system design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY ( t2_inst <* t2_set | ('border' IN
       which_class(t2_inst)) )) < 1;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- panel_system_design_definition_has_at_least_one_class
```

```
RULE panel_system_design_definition_has_at_most_one_class FOR (
        shape_aspect, group, applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition_shape := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF shape_aspect := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'panel system design definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
  violation := SIZEOF(QUERY ( t2_inst <* t2_set | ('moulded surface'
     IN which_class(t2_inst)) )) > 1;
 END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- panel_system_design_definition_has_at_most_one_class

 RULE panel_system_optionally_contains_specified_classes FOR (group,
        applied_classification_assignment, applied_group_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  cl_ass    : SET OF applied_classification_assignment := [];
  gr_ass    : SET OF applied_group_assignment := [];
  pd        : SET OF product_definition := [];
  groups    : SET OF group := [];
 END_LOCAL;
 cl_ass := QUERY ( i <* applied_classification_assignment | ((i.role.
    name = 'class membership') AND (i.assigned_class.name =
    'panel system')) );
 REPEAT i := 1 TO HIINDEX(cl_ass) BY 1;
  pd := pd + cl_ass[i].items;
 END_REPEAT;
 gr_ass := QUERY ( i <* applied_group_assignment | (i.role.name =
    'equivalence') );
 REPEAT i := 1 TO HIINDEX(gr_ass) BY 1;
  REPEAT j := 1 TO HIINDEX(gr_ass[i].items) BY 1;
   IF gr_ass[i].items[j] IN pd THEN
    groups := groups + gr_ass[i].assigned_group;
```



```
    END_IF;
   END_REPEAT;
  END_REPEAT;
  gr_ass := QUERY ( i <* applied_group_assignment | ((SIZEOF(i.items) <>
     0) AND (i.role.name = 'item structure') AND (i.assigned_group IN
    groups)) );
  REPEAT i := 1 TO HIINDEX(gr_ass) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( inst <* gr_ass[i].items | ('plate' IN
      which_class(inst)) )) = 0;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- panel_system_optionally_contains_specified_classes

 RULE plate_design_definition_has_at_least_one_class_references FOR (
         shape_aspect, group, applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'plate design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
   violation := SIZEOF(QUERY ( t2_inst <* t2_set | ('border' IN
      which_class(t2_inst)) )) < 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- plate_design_definition_has_at_least_one_class_references

 RULE plate_design_definition_has_at_most_one_class_references FOR (
         shape_aspect, group, applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
```

```
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'plate design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY ( t2_inst <* t2_set | ('moulded surface'
       IN which_class(t2_inst)) )) > 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- plate_design_definition_has_at_most_one_class_references

 RULE plate_manufacturing_definition_has_class_references FOR (
         shape_aspect, group, applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'plate manufacturing definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := SIZEOF(QUERY ( t2_inst <* t2_set | ('outer contour' IN
       which_class(t2_inst)) )) < 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- plate_manufacturing_definition_has_class_references
```



```
RULE principal_characteristics_has_properties FOR (
        property_definition_representation, group,
        applied_classification_assignment);

 LOCAL
  t3_set    : SET OF property_definition := [];
  violation : LOGICAL := FALSE;
  t4_set    : SET OF product_definition := [];
  t1_set    : SET OF product_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'principal characteristics') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 t2_set := QUERY ( i <* property_definition_representation | (i.name =
    'principal characteristics') );
 REPEAT i := 1 TO HIINDEX(t2_set) BY 1;
  t3_set := t3_set + t2_set[i].definition;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t3_set) BY 1;
  t4_set := t4_set + t3_set[i].definition;
 END_REPEAT;
 violation := t1_set <> t4_set;

WHERE
 WR1: (NOT violation);

END_RULE; -- principal_characteristics_has_properties

RULE product_definition_for_call_sign FOR (product_definition,
        applied_identification_assignment, group,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF applied_identification_assignment := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'ship designation') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
```

```
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.
      name = 'call sign') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_for_call_sign

 RULE product_definition_for_class_notation FOR (property_definition,
         group, applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'class and statutory designation') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (
      'class notation' IN which_class(t2_inst)) )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_for_class_notation

 RULE product_definition_for_design_still_water_shear_force_values FOR (
         product_definition, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
```



```
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'design still water shear  force') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'design still water shear force values') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_for_design_still_water_shear_force_values

 RULE product_definition_for_flag_state FOR (product_definition,
          applied_identification_assignment, group,
          applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF applied_identification_assignment := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'ship designation') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.
      name = 'flag state') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_for_flag_state

 RULE product_definition_for_port_of_registration FOR (product_definition,
```

```
        applied_identification_assignment, group,
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF applied_identification_assignment := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'ship designation') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.
       name = 'port of registration') )) = 1);
  END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- product_definition_for_port_of_registration

  RULE product_definition_for_regulation FOR (product_definition,
          property_definition, group, applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'class and statutory designation') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
         'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | ('regulation'
         IN which_class(t2_inst)) )) = 1);
    END_REPEAT;
```



```
WHERE
  WR1: (NOT violation);

END_RULE; -- product_definition_for_regulation

RULE product_definition_for_shipyard FOR (product_definition,
          applied_organization_assignment, group,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF applied_organization_assignment := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'shipyard designation') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
        'SHIP_STRUCTURES_SCHEMA.APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.
        name = 'shipyard') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- product_definition_for_shipyard

RULE product_definition_has_references_with_class_loadline FOR (
          product_definition, property_definition,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'freeboard characteristics') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
```

```
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | ('loadline' IN
      which_class(t2_inst)) )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_has_references_with_class_loadline

 RULE product_definition_local_coordinate_system_with_station_reference FOR (
          product_definition, property_definition, group,
          applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'hull cross section') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (
      'local co ordinate system with station reference' IN which_class(
      t2_inst)) )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_local_coordinate_system_with_station_reference

 RULE product_definition_relationship_with_definable_object FOR (
          product_definition_relationship, group,
          applied_classification_assignment);

  LOCAL
   clfied_inst : SET OF classification_item := [];
  END_LOCAL;
```



```
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
   IF applied_classification_assignment[i].assigned_class.name =
      'definable object' THEN
     clfied_inst := clfied_inst + applied_classification_assignment[i].
       items;
   END_IF;
  END_REPEAT;

 WHERE
  WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

 END_RULE; -- product_definition_relationship_with_definable_object

 RULE product_definition_relationship_with_identification_assignment FOR (
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_relationship := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   arg_list  : LIST OF STRING := ['assembly relationship',
               'structural part penetration relationship',
               'structural system penetration relationship'];
   t2_set    : SET OF product_definition_relationship := [];
  END_LOCAL;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = arg_list[j]) );
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
   violation := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_relationship_with_identification_assignment

 RULE product_definition_shape_for_assembly_manufacturing FOR (
         property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
```

```
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'assembly manufacturing definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'assembly manufacturing definition parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_assembly_manufacturing

 RULE product_definition_shape_for_bevel_design_parameters FOR (
         product_definition_shape, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'bevel design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'bevel design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);
```



```
  END_RULE; -- product_definition_shape_for_bevel_design_parameters

  RULE product_definition_shape_for_beveled_groove_weld_design FOR (
           product_definition_shape, group,
           applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition_shape := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'beveled groove weld') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
          'beveled groove weld design parameters') )) = 1);
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_beveled_groove_weld_design

  RULE product_definition_shape_for_butt_groove_weld_design FOR (
           product_definition_shape, group,
           applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition_shape := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'butt groove weld') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
```

```
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'butt groove weld design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_butt_groove_weld_design

 RULE product_definition_shape_for_circular_cutout_design_parameters FOR (
         product_definition_shape, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'circular cutout design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'circular cutout design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_circular_cutout_design_parameters

 RULE product_definition_shape_for_class_torch_vector FOR (
         product_definition_shape, shape_aspect, group,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF shape_aspect := [];
```



```
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'weld manufacturing definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | ('torch vector'
     IN which_class(t2_inst)) )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_class_torch_vector

 RULE product_definition_shape_for_continuous_fillet_weld_design FOR (
         product_definition_shape, property_definition_representation,
         group, applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'continuous fillet weld') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'continuous fillet  weld design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_continuous_fillet_weld_design
```

```
  RULE product_definition_shape_for_corrugated_part_design FOR (
          product_definition_shape, property_definition_representation,
          group, applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition_shape := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'corrugated part design definition') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
          'corrugated part design parameters') )) = 1);
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_corrugated_part_design

  RULE product_definition_shape_for_corrugated_structure_design FOR (
          product_definition_shape, property_definition_representation,
          group, applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition_shape := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'corrugated structure design definition') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
```



```
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.I
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        ' corrugated structure design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_corrugated_structure_design

 RULE product_definition_shape_for_corrugation FOR (
         product_definition_shape, shape_aspect, group,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'corrugated part design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | ('corrugation'
      IN which_class(t2_inst)) )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_corrugation

 RULE product_definition_shape_for_drain_hole_cutout_design FOR (
         product_definition_shape, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
```

```
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'drain hole cutout design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'drain hole cutout design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_drain_hole_cutout_design

 RULE product_definition_shape_for_elliptical_cutout_design FOR (
          product_definition_shape, property_definition_representation,
          applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'elliptical cutout design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'elliptical cutout design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_elliptical_cutout_design

 RULE product_definition_shape_for_elongated_oval_cutout_design FOR (
          product_definition_shape, property_definition_representation,
```



```
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'elongated oval cutout l design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'elongated oval cutout design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_elongated_oval_cutout_design

 RULE product_definition_shape_for_fillet_weld_design_parameters FOR (
        product_definition_shape, property_definition_representation,
        group, applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'fillet weld') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'fillet  weld design parameters') )) = 1);
```

```
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_fillet_weld_design_parameters

 RULE product_definition_shape_for_free_form_interior_cutout_design FOR (
          product_definition_shape, property_definition_representation,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'free form interior cutout design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
        'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
         'free form interior cutout design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_free_form_interior_cutout_design

 RULE product_definition_shape_for_groove_weld_design FOR (
          product_definition_shape, property_definition_representation,
          group, applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'groove weld') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
```



```
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'groove weld design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_groove_weld_design

 RULE product_definition_shape_for_hull_cross_section_design_definition FOR (
         product_definition);

  LOCAL
   gr_ass_set : SET OF applied_group_assignment := [];
   violation  : LOGICAL := FALSE;
   t1_set1    : SET OF product_definition_shape := [];
   t1_set2    : SET OF product_definition_shape := [];
   t2_set     : SET OF product_definition := [];
  END_LOCAL;
  t2_set := QUERY ( all_t2_inst <* product_definition | (
    'hull cross section' IN which_class(all_t2_inst)) );
  REPEAT i := 1 TO HIINDEX(t2_set) BY 1 WHILE NOT violation;
   t1_set1 := QUERY ( pd <* bag_to_set(USEDIN(t2_set[i],
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION.DEFINITION')) | (
      'SHIP_STRUCTURES_SCHEMA.PRODUCT_DEFINITION_SHAPE' IN TYPEOF(pd)) );
   t1_set2 := QUERY ( t1_inst1 <* t1_set1 | (
      'hull cross section design definition' IN which_class(t1_inst1)) );
   REPEAT j := 1 TO HIINDEX(t1_set2) BY 1;
    gr_ass_set := QUERY ( aga <* bag_to_set(USEDIN(t1_set2[j],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS')) | ((
       aga.role.name = 'versions') AND ('version history' IN
       which_class(aga.assigned_group))) );
   END_REPEAT;
   violation := SIZEOF(gr_ass_set) > 1;
   gr_ass_set := [];
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_hull_cross_section_design_definition

 RULE product_definition_shape_for_intermittent_fillet_weld_design FOR (
```

```
        product_definition_shape, property_definition_representation,
        group, applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'intermittent fillet weld') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'intermittent fillet  weld design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_intermittent_fillet_weld_design

 RULE product_definition_shape_for_inward_round_corner_design FOR (
        product_definition_shape, property_definition_representation,
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'inward round corner design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
```



```
      'inward round corner design parameters') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- product_definition_shape_for_inward_round_corner_design

RULE product_definition_shape_for_library_definition_parameters FOR (
        product_definition_shape, shape_aspect, group,
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'library definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'library definition parameters') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- product_definition_shape_for_library_definition_parameters

RULE product_definition_shape_for_library_design_parameters FOR (
        product_definition_shape, shape_aspect, group,
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'library design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
```

```
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'library design parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_library_design_parameters

  RULE product_definition_shape_for_library_manufacturing_definition FOR (
          product_definition_shape, shape_aspect, group,
          applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF shape_aspect := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'library manufacturing definition') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'library manufacturing defiinition parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_library_manufacturing_definition

  RULE product_definition_shape_for_manufacturing_definition_parameters FOR (
          product_definition_shape, shape_aspect, group,
          applied_classification_assignment);

   LOCAL
```



```
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition_shape := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF shape_aspect := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'manufacturing definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'manufacturing definition parameters') )) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- product_definition_shape_for_manufacturing_definition_parameters

RULE product_definition_shape_for_name FOR (product_definition_shape,
        shape_aspect, group, applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition_shape := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF shape_aspect := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'library design definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | ('name' IN
     which_class(t2_inst)) )) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);
```

```
END_RULE; -- product_definition_shape_for_name

RULE product_definition_shape_for_outer_contour_representation FOR (
        product_definition_shape, property_definition_representation,
        group, applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition_shape := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'plate manufacturing definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'outer contour representation') )) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- product_definition_shape_for_outer_contour_representation

RULE product_definition_shape_for_outward_round_corner_design FOR (
        product_definition_shape, property_definition_representation,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition_shape := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'outward round corner design definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
```



```
        'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
          'outward round corner design parameters') )) = 1);
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_outward_round_corner_design

  RULE product_definition_shape_for_panel_system_design_parameters FOR (
          product_definition_shape, property_definition_representation,
          group, applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition_shape := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'panel system design definition') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
          'panel system design parameters') )) = 1);
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_panel_system_design_parameters

  RULE product_definition_shape_for_part_edge_cutout_design FOR (
          product_definition_shape, property_definition_representation,
          applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition_shape := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
```

```
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'part edge cutout design definition') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'part edge cutout design parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_part_edge_cutout_design

  RULE product_definition_shape_for_plate_design_parameters FOR (
          product_definition_shape, property_definition_representation,
          group, applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'plate design definition') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'plate design parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_plate_design_parameters

  RULE product_definition_shape_for_profile_cross_section FOR (
```



```
        product_definition_shape, shape_aspect, group,
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'profile design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (
       'profile cross section' IN which_class(t2_inst)) )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_profile_cross_section

 RULE product_definition_shape_for_profile_design_parameters FOR (
        product_definition_shape, property_definition_representation,
        group, applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'profile design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
```

```
        'profile design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_profile_design_parameters

 RULE product_definition_shape_for_rectangular_cutout_design FOR (
          product_definition_shape, property_definition_representation,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'rectangular cutout design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'rectangular cutout design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_rectangular_cutout_design

 RULE product_definition_shape_for_round_corner_rectangular_cutout FOR (
          product_definition_shape, property_definition_representation,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name =
     'round corner rectangular cutout design definition') );
```



```
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'round corner rectangular cutout design parameters') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- product_definition_shape_for_round_corner_rectangular_cutout

RULE product_definition_shape_for_round_edge_rectangular_cutout FOR (
         product_definition_shape, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name =
     'round edge rectangular cutout design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'round edge rectangular cutout design parameters') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- product_definition_shape_for_round_edge_rectangular_cutout

RULE product_definition_shape_for_shear_bevel_design_parameters FOR (
         product_definition_shape, property_definition_representation,
         applied_classification_assignment);
```

```
  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'shear bevel design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
         'shear bevel design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_shear_bevel_design_parameters

 RULE product_definition_shape_for_source FOR (product_definition_shape,
          shape_aspect, group, applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'library definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | ('source' IN
       which_class(t2_inst)) )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);
```



```
  END_RULE; -- product_definition_shape_for_source

  RULE product_definition_shape_for_spot_seam_weld_design FOR (
          product_definition_shape, property_definition_representation,
          group, applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition_shape := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'spot seam weld') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
          'spot seam weld design parameters') )) = 1);
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_spot_seam_weld_design

  RULE product_definition_shape_for_structural_part_joint_design FOR (
          product_definition_shape, property_definition_representation,
          group, applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition_shape := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'structural part joint design definition') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
```

```
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'structural part joint design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_structural_part_joint_design

 RULE product_definition_shape_for_structural_system_design FOR (
         product_definition_shape, property_definition_representation,
         group, applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'structural system design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'structural system design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_for_structural_system_design

 RULE product_definition_shape_for_triangular_cutout_design FOR (
         product_definition_shape, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
```



```
    t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'triangular cutout design definition') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'triangular cutout design parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_triangular_cutout_design

  RULE product_definition_shape_for_welded_joint_design_parameters FOR (
          product_definition_shape, property_definition_representation,
          group, applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'welded joint design definition') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'welded joint design parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- product_definition_shape_for_welded_joint_design_parameters
```



```
RULE product_definition_shape_has_references_for_centre_of_gravity FOR (
        product_definition_shape, shape_aspect, group,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition_shape := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF shape_aspect := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'assembly manufacturing definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (
     'centre of gravity' IN which_class(t2_inst)) )) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- product_definition_shape_has_references_for_centre_of_gravity

RULE product_definition_shape_has_references_with_class_border FOR (
        product_definition_shape, shape_aspect, group,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition_shape := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF shape_aspect := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'corrugated part design definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | ('border' IN
```



```
      which_class(t2_inst)) )) = 2);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- product_definition_shape_has_references_with_class_border

RULE product_definition_shape_has_references_with_name_weld_design_parameters FOR (
         product_definition_shape, property_definition_representation,
         applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'weld design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'weld design parameters') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- product_definition_shape_has_references_with_name_weld_design_parameters

RULE product_definition_shape_of_rectangular_cutback_corner_design FOR (
         product_definition_shape, property_definition_representation,
         applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition_shape := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name =
     'rectangular cutback corner design definition') );
```

```
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'rectangular cutback corner design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_definition_shape_of_rectangular_cutback_corner_design

 RULE product_definition_shape_with_definition_matches_pattern FOR (
         property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
   clfied_inst : SET OF classification_item := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
   IF applied_classification_assignment[i].assigned_class.name =
      'DEFINITION' THEN
     clfied_inst := clfied_inst + applied_classification_assignment[i].
        items;
   END_IF;
  END_REPEAT;

 WHERE
  WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

 END_RULE; -- product_definition_shape_with_definition_matches_pattern

 RULE product_definition_shape_with_identification_assignment FOR (
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition_shape := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   arg_list  : LIST OF STRING := ['explicit feature design definition',
             'assembly manufacturing definition',
             'structural added material feature design definition',
             'structural part manufacturing definition',
             'corrugated part design definition','plate',
             'plate design definition','profile design definition',
```



```
            'weld manufacturing definition',
            'bevel design definition',
            'circular cutout design definition',
            'drain hole cutout design definition',
            'elliptical cutout design definition',
            'elongated oval cutout design definition',
            'free form interior cutout design definition',
            'inward round corner design definition',
            'outward round corner design definition',
            'part edge cutout design definition',
            'rectangular cutback corner design definition',
            'rectangular cutout design definition',
            'round edge rectangular cutout design definition',
            'seam design definition','shear bevel design definition',
            'triangular cutout design definition'];
  t2_set    : SET OF product_definition_shape := [];
 END_LOCAL;
 REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = arg_list[j]) );
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
  violation := NOT (SIZEOF(t2_set) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- product_definition_shape_with_identification_assignment

RULE product_definition_shape_with_versionable_object FOR (group,
       applied_classification_assignment);

 LOCAL
  clfied_inst : SET OF classification_item := [];
 END_LOCAL;
 REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
  IF applied_classification_assignment[i].assigned_class.name =
     'versionable object' THEN
   clfied_inst := clfied_inst + applied_classification_assignment[i].
      items;
  END_IF;
 END_REPEAT;
```

```
 WHERE
  WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

 END_RULE; -- product_definition_shape_with_versionable_object

 RULE product_definition_shape_with_versionable_object_matches_pattern FOR (
         group, applied_classification_assignment);

  LOCAL
   clfied_inst : SET OF classification_item := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
   IF applied_classification_assignment[i].assigned_class.name =
      'versionable object' THEN
     clfied_inst := clfied_inst + applied_classification_assignment[i].
        items;
   END_IF;
  END_REPEAT;

 WHERE
  WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

 END_RULE; -- product_definition_shape_with_versionable_object_matches_pattern

 RULE product_definition_with_date_freeboard_assigned FOR (
         product_definition, applied_date_and_time_assignment,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF applied_date_and_time_assignment := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'freeboard characteristics') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'APPLIED_DATE_AND_TIME_ASSIGNMENT.ITEMS'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.
      name = 'date freeboard assigned') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);
```



```
END_RULE; -- product_definition_with_date_freeboard_assigned

RULE product_definition_with_definable_object_matches_pattern FOR (
        product_definition, group, applied_classification_assignment);

 LOCAL
  clfied_inst : SET OF classification_item := [];
 END_LOCAL;
 REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
  IF applied_classification_assignment[i].assigned_class.name =
     'definable object' THEN
    clfied_inst := clfied_inst + applied_classification_assignment[i].
       items;
  END_IF;
 END_REPEAT;

WHERE
 WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

END_RULE; -- product_definition_with_definable_object_matches_pattern

RULE product_definition_with_freeboard_assigned_by FOR (
        product_definition, applied_organization_assignment,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF applied_organization_assignment := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'freeboard characteristics') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
     'APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.
     name = 'freeboard assigned by') )) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- product_definition_with_freeboard_assigned_by
```

```
RULE product_definition_with_identification_assignment FOR (
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  arg_list  : LIST OF STRING := ['principal characteristics',
              'class and statutory designation',
              'freeboard characteristics','lightship definition',
              'ship designation','shipyard designation',
              'owner designation','assembly','corrugated part',
              'plate boundary relationship','profile',
              'versionable object'];
  t2_set    : SET OF product_definition := [];
 END_LOCAL;
 REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = arg_list[j]) );
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
  violation := NOT (SIZEOF(t2_set) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- product_definition_with_identification_assignment

RULE product_related_product_category_with_identification_assignment FOR (
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_related_product_category := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  arg_list  : LIST OF STRING := ['ship type'];
  t2_set    : SET OF product_related_product_category := [];
 END_LOCAL;
 REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = arg_list[j]) );
 END_REPEAT;
```



```
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
   violation := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_related_product_category_with_identification_assignment

 RULE product_with_identification_assignment FOR (
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF product := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   arg_list  : LIST OF STRING := ['definable object'];
   t2_set    : SET OF product := [];
  END_LOCAL;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = arg_list[j]) );
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
   violation := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- product_with_identification_assignment

 RULE profile_cross_section_has_at_most_one_named_references FOR (
         shape_aspect, property_definition_representation, group,
         applied_classification_assignment);
```

```
    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF shape_aspect := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'profile cross section') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
      violation := SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
         'section properties') )) > 1;
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- profile_cross_section_has_at_most_one_named_references

  RULE profile_design_definition_has_at_most_one_class_references FOR (
           product_definition_shape, shape_aspect, group,
           applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF product_definition_shape := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF shape_aspect := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'profile design definition') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],
         'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
      violation := SIZEOF(QUERY ( t2_inst <* t2_set | (
         'profile curve trace line' IN which_class(t2_inst)) )) > 1;
    END_REPEAT;
```



```
WHERE
 WR1: (NOT violation);

END_RULE; -- profile_design_definition_has_at_most_one_class_references

RULE profile_design_definition_has_zero_or_two_class_references FOR (
        product_definition_shape, shape_aspect, group,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF product_definition_shape := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF shape_aspect := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'profile design definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT.OF_SHAPE'));
  violation := (NOT (SIZEOF(QUERY ( t2_inst <* t2_set | ('border' IN
     which_class(t2_inst)) )) = 0)) AND (NOT (SIZEOF(
     QUERY ( t2_inst <* t2_set | ('border' IN which_class(t2_inst)) ))
     = 2));
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- profile_design_definition_has_zero_or_two_class_references

RULE property_definition_for_class_notation FOR (property_definition,
        property_definition_representation, group,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF property_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'class notation') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
```

```
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'class notation') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- property_definition_for_class_notation

 RULE property_definition_for_class_society FOR (property_definition,
         applied_organization_assignment, group,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF property_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF applied_organization_assignment := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'class notation') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_ORGANIZATION_ASSIGNMENT.ITEMS'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.
     name = 'class society') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- property_definition_for_class_society

 RULE property_definition_for_corrugated_structure_function FOR (
         property_definition, property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
```



```
  violation : LOGICAL := FALSE;
  t1_set    : SET OF property_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'corrugated structure functional definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'corrugated structure function parameters') )) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- property_definition_for_corrugated_structure_function

RULE property_definition_for_edge_cutout_function_parameters FOR (
        property_definition, property_definition_representation,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF property_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'edge cutout functional definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'edge cutout function parameters') )) = 1);
 END_REPEAT;

 WHERE
  WR1: (NOT violation);
```

```
END_RULE; -- property_definition_for_edge_cutout_function_parameters

RULE property_definition_for_edge_feature_function_parameters FOR (
        property_definition, property_definition_representation,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF property_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'edge feature functional definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'edge feature function parameters') )) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- property_definition_for_edge_feature_function_parameters

RULE property_definition_for_homogeneous_ship_material_property FOR (
        property_definition, property_definition_representation, group,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF property_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'homogeneous ship material property') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
```



```
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'homogeneous ship  material property parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- property_definition_for_homogeneous_ship_material_property

 RULE property_definition_for_interior_cutout_function_parameters FOR (
         property_definition, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF property_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'interior cutout functional definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'interior cutout function parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- property_definition_for_interior_cutout_function_parameters

 RULE property_definition_for_local_coordinate_system FOR (
         property_definition, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF property_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
```

```
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'local coordinate system') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION' + '.' + 'DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
     'local coordinate system') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- property_definition_for_local_coordinate_system

 RULE property_definition_for_local_coordinate_system_with_position FOR (
        property_definition, property_definition_representation, group,
        applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF property_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name =
    'local co ordinatesystem with position reference') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
     'local co ordinatesystem with position reference') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- property_definition_for_local_coordinate_system_with_position
```



```
  RULE property_definition_for_panel_system_function_parameters FOR (
          property_definition, property_definition_representation, group,
          applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF property_definition := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'panel system functional definition') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
          'panel system function parameters') )) = 1);
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- property_definition_for_panel_system_function_parameters

  RULE property_definition_for_plate_function_parameters FOR (
          property_definition, property_definition_representation, group,
          applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF property_definition := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'plate functional definition') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
```

```
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESE
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'plate function parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- property_definition_for_plate_function_parameters

 RULE property_definition_for_plate_strake_function_parameters FOR (
         property_definition, property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF property_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'plate strake functional definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'plate strake function parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- property_definition_for_plate_strake_function_parameters

 RULE property_definition_for_profile_function_parameters FOR (
         property_definition, property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF property_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
```



```
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'profile functional definition') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'profile function parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- property_definition_for_profile_function_parameters

  RULE property_definition_for_ship_material_property_parameters FOR (
         property_definition, property_definition_representation, group,
         applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF property_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'ship material property') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'ship material property parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- property_definition_for_ship_material_property_parameters
```

```
RULE property_definition_for_structural_system_function_parameters FOR (
        property_definition, property_definition_representation,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF property_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'structural system functional definition') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'structural system function parameters') )) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- property_definition_for_structural_system_function_parameters

RULE property_definition_for_weld_filler_material_design FOR (
        property_definition, property_definition_representation,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF property_definition := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  t2_set    : SET OF property_definition_representation := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'weld filler material') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
     'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
  violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
```



```
      'weld filler material design parameters') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- property_definition_for_weld_filler_material_design

RULE property_definition_has_references_lightship_weight_item FOR (
         property_definition, property_definition_representation,
         applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF property_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'lightship weight item') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'lightship weight item') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- property_definition_has_references_lightship_weight_item

RULE property_definition_has_references_with_name_loadline FOR (
         property_definition, property_definition_representation,
         applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF property_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'loadline') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
```

```
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'loadline') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- property_definition_has_references_with_name_loadline

  RULE property_definition_structural_weld_shrinkage_allowance_feature FOR (
          property_definition, property_definition_representation,
          applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF property_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name =
      'structural weld shrinkage allowance feature design definition') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'structural weld shrinkage allowance feature design parameters') ))
       = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- property_definition_structural_weld_shrinkage_allowance_feature

  RULE property_definition_to_structural_added_material_feature_design FOR (
          property_definition, property_definition_representation,
          applied_classification_assignment);
```



```
  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF property_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name =
      'structural added material feature design definition') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
        'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'structural added material feature design parameters') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- property_definition_to_structural_added_material_feature_design

RULE property_definition_with_definition_matches_pattern FOR (group,
        applied_classification_assignment);

  LOCAL
    clfied_inst : SET OF classification_item := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
    IF applied_classification_assignment[i].assigned_class.name =
        'DEFINITION' THEN
      clfied_inst := clfied_inst + applied_classification_assignment[i].
          items;
    END_IF;
  END_REPEAT;

WHERE
  WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

END_RULE; -- property_definition_with_definition_matches_pattern

RULE property_definition_with_identification_assignment FOR (
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
```



```
    t1_set    : SET OF property_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    arg_list  : LIST OF STRING := [
              'moulded form characteristics definition',
              'ship overall dimensions','local co ordinate system',
              'spacing table','global axis placement',
              'hydrostatic definition','stability definition',
              'plate functional definition',
              'profile functional definition',
              'edge cutout functional definition',
              'edge feature functional definition',
              'interior cutout functional definition','structural weld shrinkage allowance  feature design
definition'];
    t2_set    : SET OF property_definition := [];
  END_LOCAL;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = arg_list[j]) );
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],
      'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
   violation := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- property_definition_with_identification_assignment

 RULE property_definition_with_versionable_object_matches_pattern FOR (
        group, applied_classification_assignment);

  LOCAL
   clfied_inst : SET OF classification_item := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
   IF applied_classification_assignment[i].assigned_class.name =
      'versionable object' THEN
     clfied_inst := clfied_inst + applied_classification_assignment[i].
        items;
   END_IF;
  END_REPEAT;

 WHERE
```



```
  WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

END_RULE; -- property_definition_with_versionable_object_matches_pattern

RULE property_definition_with_weight_and_centre_of_gravity FOR (
         property_definition, property_definition_representation,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF property_definition := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'weight and centre of gravity') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
      'PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
   violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'weight and centre of gravity') )) = 1);
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- property_definition_with_weight_and_centre_of_gravity

RULE representation_design_vertical_wave_bending_moment_parameters FOR (
         representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['hogging','sagging'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
     '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'design vertical wave bending moment parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
```

```
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_design_vertical_wave_bending_moment_parameters

 RULE representation_for_angle_bar_cross_section_dimensions FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['depth','width','radius','thk'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'angle bar cross section dimensions') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_angle_bar_cross_section_dimensions

 RULE representation_for_assembly_manufacturing_definition FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['assembly method','assembly stage',
                'mass','center of gravity'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'assembly manufacturing definition parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
```



```
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_assembly_manufacturing_definition

 RULE representation_for_bulbflat_cross_section_dimensions FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['depth','bulb width','bulb radius',
             'tw'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'bulbflat cross section dimensions') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_bulbflat_cross_section_dimensions

 RULE representation_for_channel_cross_section_dimensions FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['depth','width','web thk',
             'flange thk','radius','k'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
```

```
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'channel cross section dimensions') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_channel_cross_section_dimensions

 RULE representation_for_circular_cutout_design_parameters FOR (
         representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['radius'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
     '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'circular cutout design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_circular_cutout_design_parameters

 RULE representation_for_circular_hollow_profile_cross_section FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['depth','width','web thk',
            'flange thk','radius','k'];
   reps      : BAG OF representation := [];
  END_LOCAL;
```



```
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'channel cross section dimensions') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_circular_hollow_profile_cross_section

 RULE representation_for_class_and_statutory_designation FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['class number'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'class and statutory designation') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_class_and_statutory_designation

 RULE representation_for_corrugated_part_design_parameters FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['mirrored','offset','repetition',
```

```
              'thickness'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'corrugated part design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_corrugated_part_design_parameters

 RULE representation_for_design_vertical_wave_shear_force_parameters FOR (
          representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['negative value','positive value'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
      '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'design vertical wave shear force parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_design_vertical_wave_shear_force_parameters

 RULE representation_for_drain_hole_cutout_design_parameters FOR (
          representation);

  LOCAL
    violation : LOGICAL := FALSE;
```



```
  arg_list  : LIST OF STRING := ['depth','drain hole radius','gap',
              'gap radius','width'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'drain hole cutout design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_drain_hole_cutout_design_parameters

RULE representation_for_edge_cutout_function_parameters FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['function'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'edge cutout function parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_edge_cutout_function_parameters

RULE representation_for_edge_feature_function_parameters FOR (
        representation);

 LOCAL
```

```
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['function'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
     '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'edge feature function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
         name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_edge_feature_function_parameters

 RULE representation_for_electrode_chemical_composition_design FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['carbon','silicon','manganese',
                'phosphorus','sulphur','nickel','molydenum','aliminium',
                'titanium','zirconium','chromium','copper','vanadium'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'electrode chemical composition design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
         name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_electrode_chemical_composition_design

 RULE representation_for_elliptical_cutout_design_parameters FOR (
```



```
        representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['half axis a','half axis b'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
      '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'elliptical cutout design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_for_elliptical_cutout_design_parameters

RULE representation_for_elongated_oval_cutout_design_parameters FOR (
        representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['distance','r1','r2'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
      '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'elongated oval cutout design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_for_elongated_oval_cutout_design_parameters

RULE representation_for_explicit_profile_cross_section_shape FOR (
```

```
        representation, representation_item,
        property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['cross section geometry',
                'local coordinate system'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'explicit profile cross section shape') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_explicit_profile_cross_section_shape

 RULE representation_for_flanged_plate_cross_section_dimensions FOR (
        representation, representation_item,
        property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['depth','width','radius','thk'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'flanged plate cross section dimensions') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);
```



```
END_RULE; -- representation_for_flanged_plate_cross_section_dimensions

RULE representation_for_flat_bar_cross_section_dimensions FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['depth','width'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'flat bar cross section dimensions') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_flat_bar_cross_section_dimensions

RULE representation_for_global_axis_placement FOR (representation,
        representation_item, property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['global axes and origin',
            'after perpendicular offset','orientation'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'global axis placement') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

 WHERE
```



```
  WR1: (NOT violation);

END_RULE; -- representation_for_global_axis_placement

RULE representation_for_interior_cutout_function_parameters FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['function'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'interior cutout function parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_interior_cutout_function_parameters

RULE representation_for_inward_round_corner_design_parameters FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['radius','x offset','y offset'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'inward round corner design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
```



```
  WR1: (NOT violation);

END_RULE; -- representation_for_inward_round_corner_design_parameters

RULE representation_for_local_co_ordinate_system FOR (representation,
        representation_item, property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['local axes and origin'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'local co ordinatesystem') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_for_local_co_ordinate_system

RULE representation_for_moment_3d_restricts_representation_item FOR (
        representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['longitudinal moment',
              'transverse moment','vertical moment','origin'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( i <* representation | (i.name = 'moment 3d') );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_for_moment_3d_restricts_representation_item
```



```
RULE representation_for_outward_round_corner_design_parameters FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['radius','x offset','y offset'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'outward round corner design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_outward_round_corner_design_parameters

RULE representation_for_part_edge_cutout_design_parameters FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['r left','r right','x a','x b','x c',
           'x d','y b','y c'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'part edge cutout design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_part_edge_cutout_design_parameters
```



```
RULE representation_for_profile_curve_trace_line_design_parameters FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['curve'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'profile curve trace line design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_profile_curve_trace_line_design_parameters

RULE representation_for_profile_function_parameters FOR (representation,
        representation_item, property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['function'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'profile function parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_profile_function_parameters
```

```
RULE representation_for_rectangular_cutback_corner_design FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['radius','x depth','y depth'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'rectangular cutback corner design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_rectangular_cutback_corner_design

RULE representation_for_rectangular_cutout_design_parameters FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['length of','width'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'rectangular cutout design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_rectangular_cutout_design_parameters
```



```
RULE representation_for_round_bar_cross_section_dimensions FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['radius'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'round bar cross section dimensions') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_round_bar_cross_section_dimensions

RULE representation_for_round_corner_rectangular_cutout_design FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['corner radius','length of','width'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'round corner rectangular cutout design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_round_corner_rectangular_cutout_design
```

```
RULE representation_for_round_edge_rectangular_cutout_design FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['distance','edge radius','length of',
              'width'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'round edge rectangular cutout design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_round_edge_rectangular_cutout_design

RULE representation_for_shear_bevel_design_parameters FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['offset','x y angle','x z angle'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'shear bevel design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_shear_bevel_design_parameters
```



```
RULE representation_for_square_tube_cross_section_dimensions FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['depth','width','wall thk'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'square tube cross section dimensions') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_square_tube_cross_section_dimensions

RULE representation_for_structural_added_material_feature_design FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['added_material_length'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name = (
    'structural added material feature ' + 'design parameters')) )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);
```

```
 END_RULE; -- representation_for_structural_added_material_feature_design

 RULE representation_for_structural_part_joint_design FOR (representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['tightness','joint orientation'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'structural part joint design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_structural_part_joint_design

 RULE representation_for_structural_part_symmetry_relationship FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['mirroring plane'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'structural part symmetry relationship design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);
```



```
END_RULE; -- representation_for_structural_part_symmetry_relationship

RULE representation_for_structural_system_symmetry_relationship FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['mirroring plane'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'structural system symmetry relationship design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_for_structural_system_symmetry_relationship

RULE representation_for_t_bar_cross_section_dimensions FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['depth','width','web thk',
            'flange thk','radius','k'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    't bar cross section dimensions') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;
```

```
 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_t_bar_cross_section_dimensions

 RULE representation_for_triangular_cutout_design_parameters FOR (
         representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['c1 radius','c2 radius','c3 radius',
               'c2 x','c3 x','c3 y'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
     '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'triangular cutout design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_for_triangular_cutout_design_parameters

 RULE representation_for_twist_location_design_parameters FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['location','direction'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'twist location design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
```



```
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_for_twist_location_design_parameters

RULE representation_has_global_unit_assigned_context FOR (representation);

  LOCAL
    has_guac : LOGICAL := TRUE;
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(representation) BY 1 WHILE has_guac;
    REPEAT j := 1 TO SIZEOF(representation[i].items) BY 1 WHILE has_guac;
      IF ('SHIP_STRUCTURES_SCHEMA.VALUE_REPRESENTATION_ITEM' IN TYPEOF(
          representation[i].items[j])) OR (
          'SHIP_STRUCTURES_SCHEMA.GEOMETRIC_REPRESENTATION_ITEM' IN
          TYPEOF(representation[i].items[j])) THEN
        has_guac :=
          'SHIP_STRUCTURES_SCHEMA.GLOBAL_UNIT_ASSIGNED_CONTEXT' IN
          TYPEOF(representation[i].context_of_items);
      END_IF;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: has_guac;

END_RULE; -- representation_has_global_unit_assigned_context

RULE representation_has_references_with_name_moment_3d FOR (
         representation, property_definition_representation,
         applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF representation := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'weight and centre of gravity') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'PROPERTY_DEFINITION_REPRESENTATION.USED_REPRESENTATION'));
```

```
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'moment 3d') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_has_references_with_name_moment_3d

 RULE representation_item_corrugated_structure_boundary_relationship FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['displacement'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'corrugated structure boundary relationship design parameters') ))
     > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_item_corrugated_structure_boundary_relationship

 RULE representation_item_for_panel_system_curve_boundary_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['displacement'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'panel system curve boundary design parameters') )) > 0) );
```



```
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT found);

END_RULE; -- representation_item_for_panel_system_curve_boundary_design

RULE representation_item_for_structural_weld_shrinkage_allowance FOR (
        representation, representation_item,
        property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['longitudinal allowance',
           'transverse allowance','vertical allowance'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name = (
     'structural weld shrinkage allowance feature ' +
     'design parameters')) )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT found);

END_RULE; -- representation_item_for_structural_weld_shrinkage_allowance

RULE representation_item_for_transformation_to_parent FOR (
        property_definition, representation, mapped_item, group,
        applied_classification_assignment);

  LOCAL
   t3_set     : SET OF representation := [];
   t4_set     : SET OF representation_map := [];
   t1_set     : SET OF property_definition := [];
   t5_set     : SET OF mapped_item := [];
   c_a_set    : SET OF applied_classification_assignment := [];
```



```
    arg_list   : LIST OF STRING := ['local coordinate system position in global coordinate
system','local coordinate system position in parent local coordinate system','local coordinate system
position in parent local coordinate system with position reference'];
    violation1 : LOGICAL := FALSE;
    violation2 : LOGICAL := FALSE;
    t2_set     : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'local co ordinate system') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation1;
   t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation1 := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.
      used_representation.name = 'local axis representation') )) = 1);
    t3_set := t3_set + t2_set[i].used_representation;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation1;
   t4_set :=
bag_to_set(USEDIN(t3_set[i],'SHIP_STRUCTURES_SCHEMA.REPRESENTATION_MAP.MAPPE
D_REPRESENTATION'));
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation1;
   t5_set := bag_to_set(USEDIN(t4_set[i],
     'SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM.MAPPING_SOURCE'));
   REPEAT j := 1 TO 3 BY 1 WHILE NOT violation2;
    violation2 := NOT (SIZEOF(QUERY ( t2_inst <* t5_set | (t2_inst.
      name = arg_list[j]) )) = 1);
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation1);
  WR2: (NOT violation2);

 END_RULE; -- representation_item_for_transformation_to_parent

 RULE representation_item_to_structural_system_adjacency_relationship FOR (
        representation, representation_item,
        property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['displacement'];
   reps     : BAG OF representation := [];
```



```
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'structural system adjacency relationship design parameters') )) >
     0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_item_to_structural_system_adjacency_relationship

 RULE representation_items_for_corrugated_part_design_parameters FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['mass'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'corrugated part design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_for_corrugated_part_design_parameters

 RULE representation_items_for_corrugated_structure_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
```

```
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['corrugation width',
               'user defined tightness'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'corrugated structure design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
         = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_for_corrugated_structure_design

 RULE representation_items_for_corrugated_structure_function FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['user def function'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'corrugated structure function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
         = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_for_corrugated_structure_function

 RULE representation_items_for_edge_cutout_function_parameters FOR (
          representation);
```



```
  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['user def function'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
      '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'edge cutout function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
          = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT found);

END_RULE; -- representation_items_for_edge_cutout_function_parameters

RULE representation_items_for_edge_feature_function_parameters FOR (
        representation);

  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['user def function'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
      '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'edge feature function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
          = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT found);

END_RULE; -- representation_items_for_edge_feature_function_parameters

RULE representation_items_for_elongated_oval_cutout_design FOR (
        representation);
```

```
  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['r3','r3 x','r3 y'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
      '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'elongated oval cutout design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
          = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_for_elongated_oval_cutout_design

 RULE representation_items_for_interior_cutout_function_parameters FOR (
         representation);

  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['user def function'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
      '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'interior cutout function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
          = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_for_interior_cutout_function_parameters

 RULE representation_items_for_inward_round_corner_design FOR (
         representation);
```



```
  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['z depth'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
      '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'inward round corner design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
         = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT found);

END_RULE; -- representation_items_for_inward_round_corner_design

RULE representation_items_for_maximum_permissible_swbm_values FOR (
         compound_representation_item);

  LOCAL
    violation : LOGICAL := FALSE;
    rep_item  : SET OF representation_item := [];
    arg_list  : LIST OF STRING := ['hogging','sagging','position'];
    creps     : BAG OF compound_representation_item := [];
  END_LOCAL;
  creps := QUERY ( temp_comp_rep_item <* compound_representation_item |
      (temp_comp_rep_item.name = 'maximum permissible swbm values') );
  REPEAT i := 1 TO HIINDEX(creps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      REPEAT k := 1 TO HIINDEX(creps[i].item_element) BY 1;
        rep_item := creps[i].item_element;
        violation := SIZEOF(QUERY ( items <* rep_item | (items.name =
           arg_list[j]) )) = 1;
      END_REPEAT;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_items_for_maximum_permissible_swbm_values

RULE representation_items_for_maximum_permissible_swsf_values FOR (
         compound_representation_item);
```

```
  LOCAL
    violation : LOGICAL := FALSE;
    rep_item  : SET OF representation_item := [];
    arg_list  : LIST OF STRING := ['negative value','positive value',
                'position'];
    creps     : BAG OF compound_representation_item := [];
  END_LOCAL;
  creps := QUERY ( temp_comp_rep_item <* compound_representation_item |
     (temp_comp_rep_item.name = 'maximum permissible swsf values') );
  REPEAT i := 1 TO HIINDEX(creps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      REPEAT k := 1 TO HIINDEX(creps[i].item_element) BY 1;
        rep_item := creps[i].item_element;
        violation := SIZEOF(QUERY ( items <* rep_item | (items.name =
           arg_list[j]) )) = 1;
      END_REPEAT;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_items_for_maximum_permissible_swsf_values

 RULE representation_items_for_outward_round_corner_design FOR (
          representation);

  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['z depth'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
     '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'outward round corner design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
         = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_for_outward_round_corner_design

 RULE representation_items_for_panel_system_boundary_relationship FOR (
```



```
        representation, representation_item,
        property_definition_representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['displacement'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'panel system boundary relationship design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT found);

END_RULE; -- representation_items_for_panel_system_boundary_relationship

RULE representation_items_for_panel_system_function_parameters FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['user def function'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'panel system function parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT found);

END_RULE; -- representation_items_for_panel_system_function_parameters
```

```
RULE representation_items_for_profile_boundary_relationship_design FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['displacement'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'profile boundary relationship design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT found);

END_RULE; -- representation_items_for_profile_boundary_relationship_design

RULE representation_items_for_profile_curve_trace_line_design FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['displacement'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'profile curve trace line design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT found);
```



```
END_RULE; -- representation_items_for_profile_curve_trace_line_design

RULE representation_items_for_profile_trace_line_relationship_design FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['displacement','auxiliary line'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'profile trace line relationship design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT found);

END_RULE; -- representation_items_for_profile_trace_line_relationship_design

RULE representation_items_for_rectangular_cutback_corner_design FOR (
        representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['z depth'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'rectangular cutback corner design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT found);
```

```
 END_RULE; -- representation_items_for_rectangular_cutback_corner_design

 RULE representation_items_for_seam_curve_relationship_parameter FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['displacement'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'seam curve relationship parameter') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_for_seam_curve_relationship_parameter

 RULE representation_items_for_shear_bevel_design_parameters FOR (
         representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['z angle'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
     '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'shear bevel design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);
```



```
END_RULE; -- representation_items_for_shear_bevel_design_parameters

RULE representation_items_optional_for_bevel_design_parameters FOR (
        representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['z angle'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
    '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'bevel design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT found);

END_RULE; -- representation_items_optional_for_bevel_design_parameters

RULE representation_items_optional_for_beveled_groove_weld_design FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['weld backing type','taper angle'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'beveled groove weld design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT found);
```

```
 END_RULE; -- representation_items_optional_for_beveled_groove_weld_design

 RULE representation_items_optional_for_butt_groove_weld_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['weld backing type'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'butt groove weld design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_butt_groove_weld_design

 RULE representation_items_optional_for_class_notation FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['ice class notation','service factor',
            'approval required for heavy cargo'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'class notation') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;
```



```
WHERE
 WR1: (NOT found);

END_RULE; -- representation_items_optional_for_class_notation

RULE representation_items_optional_for_corrugation_dimensions FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['radius 1','radius 2'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'corrugation dimensions') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT found);

END_RULE; -- representation_items_optional_for_corrugation_dimensions

RULE representation_items_optional_for_groove_weld_design FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  found    : LOGICAL := FALSE;
  arg_list : LIST OF STRING := ['weld backing type'];
  reps     : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'groove weld design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
   found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
      = arg_list[j]) )) > 1;
  END_REPEAT;
```

```
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_groove_weld_design

 RULE representation_items_optional_for_owner_designation FOR (
         representation);

  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['owner approval'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'owner designation') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
         = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_owner_designation

 RULE representation_items_optional_for_panel_system_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['stiffener spacing',
             'user defined tightness'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'panel system design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
```



```
        = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_panel_system_design

 RULE representation_items_optional_for_plate_design_parameters FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['mass'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'plate design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
      found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
          = arg_list[j]) )) > 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_plate_design_parameters

 RULE representation_items_optional_for_plate_function_parameters FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['user def function'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'plate function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
```

```
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
     found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
        = arg_list[j]) )) > 1;
    END_REPEAT;
   END_REPEAT;

  WHERE
   WR1: (NOT found);

  END_RULE; -- representation_items_optional_for_plate_function_parameters

  RULE representation_items_optional_for_plate_strake_function FOR (
          representation, representation_item,
          property_definition_representation);

   LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['user def function'];
    reps     : BAG OF representation := [];
   END_LOCAL;
   reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'plate strake function parameters') )) > 0) );
   REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
     found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
        = arg_list[j]) )) > 1;
    END_REPEAT;
   END_REPEAT;

  WHERE
   WR1: (NOT found);

  END_RULE; -- representation_items_optional_for_plate_strake_function

  RULE representation_items_optional_for_principal_characteristics FOR (
          representation, representation_item,
          property_definition_representation);

   LOCAL
    found    : LOGICAL := FALSE;
    arg_list : LIST OF STRING := ['block coefficient'];
    reps     : BAG OF representation := [];
   END_LOCAL;
   reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
```



```
    'principal characteristics') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_principal_characteristics

 RULE representation_items_optional_for_profile_design_parameters FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['mass'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'profile design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_profile_design_parameters

 RULE representation_items_optional_for_profile_function_parameters FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['user def function'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
```

```
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'profile function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_profile_function_parameters

 RULE representation_items_optional_for_seam_design_parameters FOR (
         representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['seam curve'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
     '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'seam design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_seam_design_parameters

 RULE representation_items_optional_for_structural_system_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['user def  tightness'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
```



```
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'structural system design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_structural_system_design

 RULE representation_items_optional_for_structural_system_function FOR (
        representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['user def function'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'structural system function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_structural_system_function

 RULE representation_items_optional_for_weld_design_parameters FOR (
        representation, representation_item,
        property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['start suspension','end suspension',
            'weld cross section area'];
   reps     : BAG OF representation := [];
```

```
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'weld design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_weld_design_parameters

 RULE representation_items_optional_for_weld_filler_material_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   found    : LOGICAL := FALSE;
   arg_list : LIST OF STRING := ['chemical composition id',
              'electrode id','material strength id',
              'notch impact work id'];
   reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'weld filler material design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT found;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT found;
    found := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.name
       = arg_list[j]) )) > 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT found);

 END_RULE; -- representation_items_optional_for_weld_filler_material_design

 RULE representation_items_specified_for_design_swsf_values FOR (
         compound_representation_item, representation_item);

  LOCAL
```



```
  violation : LOGICAL := FALSE;
  rep_item  : SET OF representation_item := [];
  arg_list  : LIST OF STRING := ['negative value','positive value',
              'position'];
  creps     : BAG OF compound_representation_item := [];
 END_LOCAL;
 creps := QUERY ( temp_comp_rep_item <* compound_representation_item |
    (temp_comp_rep_item.name = 'design swsf values') );
 REPEAT i := 1 TO HIINDEX(creps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   REPEAT k := 1 TO HIINDEX(creps[i].item_element) BY 1;
    rep_item := creps[i].item_element;
    violation := SIZEOF(QUERY ( items <* rep_item | (items.name =
       arg_list[j]) )) = 1;
   END_REPEAT;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_items_specified_for_design_swsf_values

RULE representation_local_coordinate_system_with_position_reference FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['local axes and origin'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'local co ordinatesystem with position reference') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_local_coordinate_system_with_position_reference

RULE representation_maximum_permissible_still_water_bending_moment FOR (
```

```
        representation);

  LOCAL
    violation : LOGICAL := FALSE;
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (temp_rep.name =
      'maximum permissible still water bending moment') );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = 'maximum permissible swbm values') )) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_maximum_permissible_still_water_bending_moment

 RULE representation_of_design_still_water_bending_moment_parameters FOR (
        representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['hogging amidship','sagging amidship'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
      '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'design still water bending moment parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_of_design_still_water_bending_moment_parameters

 RULE representation_of_maximum_permissible_still_water_shear_force FOR (
        representation);

  LOCAL
    violation : LOGICAL := FALSE;
    reps      : BAG OF representation := [];
  END_LOCAL;
```



```
  reps := QUERY ( temp_rep <* representation | (temp_rep.name =
     'maximum permissible still water shear force') );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = 'maximum permissible swsf values') )) <> 1;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_of_maximum_permissible_still_water_shear_force

RULE representation_restricted_by_name_bevel_design_parameters FOR (
         representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['offset','x y angle'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION' +
     '.USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'bevel design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
         name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_by_name_bevel_design_parameters

RULE representation_restricted_by_name_class_notation FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['service area',
                'approval required for oil cargo',
                'approval required for loading unloading aground',
                'approval required for unloading grabs'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
```

```
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'class notation') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_by_name_class_notation

 RULE representation_restricted_by_name_class_parameters FOR (
        representation, representation_item,
        property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['length class','length solas',
           'scantlings draught','block coefficient class',
           'design speed ahead','design speed astern'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'class parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_by_name_class_parameters

 RULE representation_restricted_by_name_corrugation_dimensions FOR (
        representation, representation_item,
        property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
```



```
  arg_list  : LIST OF STRING := ['depth','flat width 1','flat width 2',
              'slope width'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'corrugation dimensions') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_by_name_corrugation_dimensions

RULE representation_restricted_by_name_flare_area_parameters FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['area','height'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'flare area parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_by_name_flare_area_parameters

RULE representation_restricted_by_name_freeboard_characteristics FOR (
        representation);
```

```
  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['assigned code','freeboard'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'freeboard characteristics') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_by_name_freeboard_characteristics

 RULE representation_restricted_by_name_lightship_definition FOR (
         representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['lightship weight',
              'lightship centre of gravity'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'lightship_definition') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_by_name_lightship_definition

 RULE representation_restricted_by_name_lightship_weight_item FOR (
         representation);
```



```
  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['aft weight extent',
                'fwd weight extent'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'lightship weight item') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_by_name_lightship_weight_item

RULE representation_restricted_by_name_loadline FOR (representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['load line length','load line depth',
                'load line displacement','load line block coefficient',
                'load line regulation'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name = 'loadline') ))
      > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_by_name_loadline

RULE representation_restricted_by_name_plate_design_parameters FOR (
```

```
        representation, representation_item,
        property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['material offset','thickness'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'plate design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_by_name_plate_design_parameters

 RULE representation_restricted_by_name_plate_function_parameters FOR (
        representation, representation_item,
        property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['function'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'plate function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_by_name_plate_function_parameters
```



```
RULE representation_restricted_by_name_principal_characteristics FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['length between perpendiculars',
              'moulded breadth','moulded depth','design draught',
              'design deadweight','min draught at fp',
              'max draught at fp','min draught at ap',
              'max draught at ap'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'principal characteristics') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_by_name_principal_characteristics

RULE representation_restricted_by_name_profile_design_parameters FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['mirrored'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'profile design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;
```



```
 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_by_name_profile_design_parameters

 RULE representation_restricted_by_name_section_properties FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['nominal mass per length','area',
                'na u','na v','moi u','moi v','moi uv','tr'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'section properties') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_by_name_section_properties

 RULE representation_restricted_by_name_table_of_shear_force_values FOR (
          representation);

  LOCAL
    violation : LOGICAL := FALSE;
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (temp_rep.name =
      'table of shear force values') );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = 'design swsf values') )) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_by_name_table_of_shear_force_values
```



```
RULE representation_restricted_by_name_w_shape_cross_section FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['depth','width','web thk',
              'flange thk','radius','k'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'w shape cross section') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_by_name_w_shape_cross_section

RULE representation_restricted_for_assembly_manufacturing_position FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['assembly footprint','orientation'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'assembly manufacturing position parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);
```

```
 END_RULE; -- representation_restricted_for_assembly_manufacturing_position

 RULE representation_restricted_for_beveled_groove_weld_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['endcut shape','taper','sideness',
               'weld joint spacer','configuration','penetration',
               'joint_orientation','tightness'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'beveled groove weld design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_beveled_groove_weld_design

 RULE representation_restricted_for_butt_groove_weld_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['face shape','sideness',
               'weld joint spacer','configuration','penetration',
               'joint_orientation','tightness'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'butt groove weld design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
```



```
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_for_butt_groove_weld_design

RULE representation_restricted_for_composite_feature_design FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['composed of','description','name'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'composite feature design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_for_composite_feature_design

RULE representation_restricted_for_continuous_fillet_weld_design FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['endcut shape type','sideness',
                'configuration','penetration','joint_orientation',
                'tightness'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'continuous fillet weld design parameters') )) > 0) );
```

```
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_continuous_fillet_weld_design

 RULE representation_restricted_for_corrugated_structure_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['thickness','tightness'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'corrugated structure design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_corrugated_structure_design

 RULE representation_restricted_for_corrugated_structure_function FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['function'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
```



```
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'corrugated structure function parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
     name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_for_corrugated_structure_function

RULE representation_restricted_for_fillet_weld_design_parameters FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['endcut shape type','sideness',
            'configuration','penetration','joint_orientation',
            'tightness'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'fillet weld design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
     name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_for_fillet_weld_design_parameters

RULE representation_restricted_for_groove_weld_design FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['sideness','weld joint spacer',
            'configuration','penetration','joint_orientation',
```

```
            'tightness'];
    reps     : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'groove weld design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_groove_weld_design

 RULE representation_restricted_for_homogeneous_ship_material_property FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['density','poisson ratio',
              'stress of fracture','thermal expansion coefficient',
              'yield point','youngs module'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'homogeneous ship material property parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_homogeneous_ship_material_property

 RULE representation_restricted_for_hull_cross_section_design_definition FOR (
         representation, representation_item,
```



```
        property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['mirrored symmetry'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'hull cross section design definition parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_for_hull_cross_section_design_definition

RULE representation_restricted_for_intermittent_fillet_weld_design FOR (
        representation, representation_item,
        property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['end rules','cutout rules',
                'fillet alignment','fillet weld length',
                'fillet weld spacing','penetration rules',
                'endcut shape type','sideness','configuration',
                'penetration','joint_orientation','tightness'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'intermittent fillet weld design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

  WHERE
```

```
  WR1: (NOT violation);

END_RULE; -- representation_restricted_for_intermittent_fillet_weld_design

RULE representation_restricted_for_panel_system_curve_boundary FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['curve'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'panel system curve boundary design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_for_panel_system_curve_boundary

RULE representation_restricted_for_panel_system_design_parameters FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['material offset','thickness',
              'tightness'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'panel system design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
```



```
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_for_panel_system_design_parameters

RULE representation_restricted_for_panel_system_function FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['function'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'panel system function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
          name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_for_panel_system_function

RULE representation_restricted_for_panel_system_plane_boundary FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['plane'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'panel system plane boundary design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
```

```
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_panel_system_plane_boundary

 RULE representation_restricted_for_plate_renewal_definition_attributes FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['rule thickness','renewal thickness'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'plate renewal definition attributes') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_plate_renewal_definition_attributes

 RULE representation_restricted_for_plate_strake_function FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['function'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'plate strake function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
```



```
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
     violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- representation_restricted_for_plate_strake_function

  RULE representation_restricted_for_ship_material_property FOR (
          representation, representation_item,
          property_definition_representation);

   LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['density'];
    reps      : BAG OF representation := [];
   END_LOCAL;
   reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'ship material property parameters') )) > 0) );
   REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
     violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
        name = arg_list[j]) )) <> 1;
    END_REPEAT;
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- representation_restricted_for_ship_material_property

  RULE representation_restricted_for_spot_seam_weld_design FOR (
          representation, representation_item,
          property_definition_representation);

   LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['weld context','configuration',
              'penetration','joint_orientation','tightness'];
    reps      : BAG OF representation := [];
   END_LOCAL;
   reps := QUERY ( temp_rep <* representation | (SIZEOF(
      QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
      'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
```

```
      'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
      'spot seam weld design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_spot_seam_weld_design

 RULE representation_restricted_for_structural_part_joint_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['tightness','joint orientation'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'structural part joint design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
       name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_structural_part_joint_design

 RULE representation_restricted_for_structural_system_design FOR (
         representation, representation_item,
         property_definition_representation);

  LOCAL
   violation : LOGICAL := FALSE;
   arg_list  : LIST OF STRING := ['thightness'];
   reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
```



```
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'structural system design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_for_structural_system_design

RULE representation_restricted_for_structural_system_function FOR (
       representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['function'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'structural system function parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
   REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
   END_REPEAT;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- representation_restricted_for_structural_system_function

RULE representation_restricted_for_weld_design_parameters FOR (
       representation, representation_item,
       property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['shape of weld surface',
          'weld geometry','connection angle','rotation angle',
          'inclination angle','shrinkage','weld size'];
```

```
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'weld design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
         name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_weld_design_parameters

 RULE representation_restricted_for_weld_filler_material_design FOR (
          representation);

  LOCAL
    violation : LOGICAL := FALSE;
    arg_list  : LIST OF STRING := ['density'];
    reps      : BAG OF representation := [];
  END_LOCAL;
  reps := QUERY ( temp_rep <* representation | (SIZEOF(
     QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
     'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
     'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
     'weld filler material design parameters') )) > 0) );
  REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
      violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
         name = arg_list[j]) )) <> 1;
    END_REPEAT;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- representation_restricted_for_weld_filler_material_design

 RULE representation_restricted_for_weld_manufacturing_definition FOR (
          representation, representation_item,
          property_definition_representation);

  LOCAL
    violation : LOGICAL := FALSE;
```



```
  arg_list  : LIST OF STRING := ['position','process',
              'degree of automations','number of weld passes',
              'welding deposition sequences','welding environment',
              'tack weld used'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'weld manufacturing definition parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_for_weld_manufacturing_definition

RULE representation_restricted_for_welded_joint_design_parameters FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['configuration','penetration',
              'joint_orientation','tightness'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'welded joint design parameters') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_for_welded_joint_design_parameters
```

```
RULE representation_restricted_for_welding_sequence_representation FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['name','start offset','end offset',
              'direction'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'welding sequence representation') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricted_for_welding_sequence_representation

RULE representation_restricted_weight_and_centre_of_gravity FOR (
        representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['mass','centre of gravity'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'weight and centre of gravity') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);
```



```
END_RULE; -- representation_restricted_weight_and_centre_of_gravity

RULE representation_restricts_profile_renewal_definition_attributes FOR (
        representation, representation_item,
        property_definition_representation);

 LOCAL
  violation : LOGICAL := FALSE;
  arg_list  : LIST OF STRING := ['renewal modulus','rule inertia',
            'rule section modulus'];
  reps      : BAG OF representation := [];
 END_LOCAL;
 reps := QUERY ( temp_rep <* representation | (SIZEOF(
    QUERY ( temp_prop_def_rep <* bag_to_set(USEDIN(temp_rep,
    'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRESENTATION.' +
    'USED_REPRESENTATION')) | (temp_prop_def_rep.name =
    'profile renewal definition attributes') )) > 0) );
 REPEAT i := 1 TO HIINDEX(reps) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
   violation := SIZEOF(QUERY ( rep_item <* reps[i].items | (rep_item.
      name = arg_list[j]) )) <> 1;
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- representation_restricts_profile_renewal_definition_attributes

RULE representation_with_versionable_object_matches_pattern FOR (group,
        applied_classification_assignment);

 LOCAL
  clfied_inst : SET OF classification_item := [];
 END_LOCAL;
 REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
  IF applied_classification_assignment[i].assigned_class.name =
     'versionable object' THEN
   clfied_inst := clfied_inst + applied_classification_assignment[i].
      items;
  END_IF;
 END_REPEAT;

WHERE
 WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

END_RULE; -- representation_with_versionable_object_matches_pattern

RULE restrict_name_for_known_source FOR (known_source);
```



```
 WHERE
  WR1: (SIZEOF(QUERY ( ks <* known_source | (ks.name <>
         'ISO 13584 library') )) = 0);

 END_RULE; -- restrict_name_for_known_source

 RULE revision_has_mandatory_attribute_description FOR (group);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF group := [];
  END_LOCAL;
  t1_set := QUERY ( i <* group | VALUE_IN(which_class(i),'revision') );
  violate := SIZEOF(QUERY ( k <* t1_set | (NOT EXISTS(k.description)) ))
     > 0;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- revision_has_mandatory_attribute_description

 RULE revision_with_context_referenced_for_context_of_revision FOR (
         applied_group_assignment, group);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* group | VALUE_IN(which_class(a),
     'revision with context') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_group_assignment | ((b.assigned_group
       = t1_set[i]) AND (b.role.name = 'context of revision')) );
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- revision_with_context_referenced_for_context_of_revision

 RULE seam_has_at_most_one_class_references FOR (shape_aspect,
         shape_aspect_relationship, group,
         applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF shape_aspect := [];
    c_a_set   : SET OF applied_classification_assignment := [];
```



```
    t2_set    : SET OF shape_aspect_relationship := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'seam') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT_RELATIONSHIP.
RELATING_SHAPE_ASPECT'));
    violation := SIZEOF(QUERY ( t2_inst <* t2_set | (
      'seam curve relationship' IN which_class(t2_inst)) )) > 1;
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- seam_has_at_most_one_class_references

  RULE shape_aspect_for_angle_bar_cross_section_dimensions FOR (
         shape_aspect, property_definition_representation, group,
         applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF shape_aspect := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'angle bar cross section') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'angle bar cross section dimensions') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);
```

```
  END_RULE; -- shape_aspect_for_angle_bar_cross_section_dimensions

  RULE shape_aspect_for_assembly_manufacturing_position_parameters FOR (
           shape_aspect, property_definition_representation, group,
           applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF shape_aspect := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'assembly manufacturing position') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
          'assembly manufacturing position parameters') )) = 1);
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_assembly_manufacturing_position_parameters

  RULE shape_aspect_for_border_corner_cutout_boundary_relationship FOR (
           shape_aspect, applied_group_assignment, group,
           applied_classification_assignment);

    LOCAL
      gr_ass_set : BAG OF applied_group_assignment := [];
      violation  : LOGICAL := FALSE;
      t1_set     : SET OF shape_aspect := [];
      c_a_set    : SET OF applied_classification_assignment := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'border') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
```



```
    gr_ass_set := USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS');
    violation := SIZEOF(QUERY ( aga <* gr_ass_set | ((aga.role.name =
       'boundary index 2') AND ('corner cutout boundary relationship' IN
        which_class(aga.assigned_group))) )) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_border_corner_cutout_boundary_relationship

 RULE shape_aspect_for_bulbflat_cross_section_dimensions FOR (
         shape_aspect, property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF shape_aspect := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'bulbflat cross section') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'bulbflat cross section dimensions') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_bulbflat_cross_section_dimensions

 RULE shape_aspect_for_channel_cross_section_dimensions FOR (shape_aspect,
         property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF shape_aspect := [];
   c_a_set   : SET OF applied_classification_assignment := [];
```

```
    t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'channel cross section') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'channel cross section dimensions') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_channel_cross_section_dimensions

  RULE shape_aspect_for_circular_hollow_cross_section_dimensions FOR (
         shape_aspect, property_definition_representation, group,
         applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF shape_aspect := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'circular hollow cross section') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'circular hollow cross section dimensions') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);
```



```
  END_RULE; -- shape_aspect_for_circular_hollow_cross_section_dimensions

  RULE shape_aspect_for_composite_feature_design_parameters FOR (
          shape_aspect, property_definition_representation, group,
          applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF shape_aspect := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'composite feature') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
          'composite feature design parameters') )) = 1);
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_composite_feature_design_parameters

  RULE shape_aspect_for_composite_feature_has_at_least_two_instances FOR (
          shape_aspect, applied_classification_assignment);

    LOCAL
      violation       : LOGICAL := FALSE;
      c_a_set         : SET OF applied_classification_assignment := [];
      t2_set          : SET OF shape_aspect := [];
      classed_s_a_set : SET OF shape_aspect := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'composite feature') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        classed_s_a_set := classed_s_a_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    c_a_set := [];
    REPEAT i := 1 TO HIINDEX(classed_s_a_set) BY 1 WHILE NOT violation;
```

```
    t2_set := bag_to_set(USEDIN(classed_s_a_set[1],
       'SHIP_STRUCTURES_SCHEMA.SHAPE_ASPECT_RELATIONSHIP.' +
       'RELATING_SHAPE_ASPECT'));
    violation := NOT (SIZEOF(t2_set) = 2);
  END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_composite_feature_has_at_least_two_instances

  RULE shape_aspect_for_corner_cutout FOR (shape_aspect,
          applied_group_assignment, group,
          applied_classification_assignment);

   LOCAL
    gr_ass_set : BAG OF applied_group_assignment := [];
    violation  : LOGICAL := FALSE;
    t1_set     : SET OF shape_aspect := [];
    c_a_set    : SET OF applied_classification_assignment := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'corner cutout') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    gr_ass_set := USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS');
    violation := SIZEOF(QUERY ( aga <* gr_ass_set | ((aga.role.name =
       'item 1') AND ('structural cutout boundary relationship' IN
       which_class(aga.assigned_group))) )) <> 1;
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_corner_cutout

  RULE shape_aspect_for_corner_cutout_boundary_relationship FOR (
          shape_aspect, applied_group_assignment, group,
          applied_classification_assignment);

   LOCAL
    gr_ass_set : BAG OF applied_group_assignment := [];
    violation  : LOGICAL := FALSE;
    t1_set     : SET OF shape_aspect := [];
    c_a_set    : SET OF applied_classification_assignment := [];
```



```
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'corner cutout') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   gr_ass_set := USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS');
   violation := SIZEOF(QUERY ( aga <* gr_ass_set | ((aga.role.name =
     'item 1') AND ('corner cutout boundary relationship' IN
     which_class(aga.assigned_group))) )) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_corner_cutout_boundary_relationship

 RULE shape_aspect_for_corrugation_dimensions FOR (shape_aspect,
        property_definition_representation, group,
        applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF shape_aspect := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'corrugation') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
      'corrugation dimensions') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_corrugation_dimensions
```

```
  RULE shape_aspect_for_explicit_profile_cross_section_dimensions FOR (
           shape_aspect, property_definition_representation, group,
           applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF shape_aspect := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'explicit profile cross section') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
      violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
          ' explicit profile cross section dimensions') )) = 1);
    END_REPEAT;

  WHERE
    WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_explicit_profile_cross_section_dimensions

  RULE shape_aspect_for_flanged_plate_cross_section_dimensions FOR (
           shape_aspect, property_definition_representation, group,
           applied_classification_assignment);

    LOCAL
      violation : LOGICAL := FALSE;
      t1_set    : SET OF shape_aspect := [];
      c_a_set   : SET OF applied_classification_assignment := [];
      t2_set    : SET OF property_definition_representation := [];
    END_LOCAL;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = 'flanged plate cross section') );
    REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
      REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
        t1_set := t1_set + c_a_set[i].items[j];
      END_REPEAT;
    END_REPEAT;
    REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
      t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.
PROPERTY_DEFINITION_REPRESENTATION.DEFINITION'));
```



```
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'flanged plate cross section dimensions') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_flanged_plate_cross_section_dimensions

 RULE shape_aspect_for_flare_area_parameters FOR (shape_aspect,
        property_definition_representation, group,
        applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF shape_aspect := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'flare area') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'flare area parameters') )) = 2);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_flare_area_parameters

 RULE shape_aspect_for_flat_bar_cross_section_dimensions FOR (
        shape_aspect, property_definition_representation, group,
        applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF shape_aspect := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
```

```
      assigned_class.name = 'flat bar cross section') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'flat  bar cross section dimensions') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_flat_bar_cross_section_dimensions

  RULE shape_aspect_for_panel_system_curve_boundary_design FOR (
          shape_aspect, property_definition_representation, group,
          applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF shape_aspect := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'panel system curve boundary') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'panel system curve boundary design parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_panel_system_curve_boundary_design

  RULE shape_aspect_for_panel_system_plane_boundary_design FOR (
```



```
        shape_aspect, property_definition_representation, group,
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF shape_aspect := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'panel system plane boundary') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       'panel system plane boundary design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_panel_system_plane_boundary_design

 RULE shape_aspect_for_profile_curve_trace_line_design_parameters FOR (
        shape_aspect, property_definition_representation, group,
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF shape_aspect := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'profile curve trace line') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_
REPRESENTATION.DEFINITION'));
```



```
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'profile curve trace line design parameters') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_profile_curve_trace_line_design_parameters

 RULE shape_aspect_for_round_bar_cross_section_dimensions FOR (
         shape_aspect, property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF shape_aspect := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'round bar cross section') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
   REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
    t1_set := t1_set + c_a_set[i].items[j];
   END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
   t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'round bar cross section dimensions') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_round_bar_cross_section_dimensions

 RULE shape_aspect_for_square_tube_cross_section_dimensions FOR (
         shape_aspect, property_definition_representation, group,
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF shape_aspect := [];
   c_a_set   : SET OF applied_classification_assignment := [];
   t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
```



```
      assigned_class.name = 'square tube cross section') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'square tube cross section dimensions') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_square_tube_cross_section_dimensions

  RULE shape_aspect_for_structural_cutout_boundary_relationship FOR (
          shape_aspect, applied_group_assignment, group,
          applied_classification_assignment);

   LOCAL
    gr_ass_set : BAG OF applied_group_assignment := [];
    violation  : LOGICAL := FALSE;
    t1_set     : SET OF shape_aspect := [];
    c_a_set    : SET OF applied_classification_assignment := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'structural cutout') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    gr_ass_set := USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_GROUP_ASSIGNMENT.ITEMS');
    violation := SIZEOF(QUERY ( aga <* gr_ass_set | ((aga.role.name =
       'item 1') AND ('structural cutout boundary relationship' IN
       which_class(aga.assigned_group))) )) <> 1;
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_structural_cutout_boundary_relationship

  RULE shape_aspect_for_t_bar_cross_section_dimensions FOR (shape_aspect,
```

```
          property_definition_representation, group,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF shape_aspect := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 't bar cross section') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
       't bar cross section dimensions') )) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_for_t_bar_cross_section_dimensions

 RULE shape_aspect_for_twist_location_design_parameters FOR (shape_aspect,
          property_definition_representation, group,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF shape_aspect := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF property_definition_representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'twist location') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_
DEFINITION_REPRESENTATION.DEFINITION'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
```



```
        'twist location design parameters') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_twist_location_design_parameters

  RULE shape_aspect_for_w_shape_cross_section_dimensions FOR (shape_aspect,
          property_definition_representation, group,
          applied_classification_assignment);

   LOCAL
     violation : LOGICAL := FALSE;
     t1_set    : SET OF shape_aspect := [];
     c_a_set   : SET OF applied_classification_assignment := [];
     t2_set    : SET OF property_definition_representation := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'w shape cross section') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set :=
bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.PROPERTY_DEFINITION_REPRES
ENTATION.DEFINITION'));
     violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.name =
        'w shape cross section dimensions') )) = 1);
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- shape_aspect_for_w_shape_cross_section_dimensions

  RULE shape_aspect_relationship_attributes_references_are_distinct FOR (
          shape_aspect_relationship);

   LOCAL
     cyclic_relationship : LOGICAL := FALSE;
   END_LOCAL;
   REPEAT i := 1 TO HIINDEX(shape_aspect_relationship) BY 1 WHILE NOT
      cyclic_relationship;
     cyclic_relationship := shape_aspect_relationship[i].
        relating_shape_aspect :=: shape_aspect_relationship[i].
        related_shape_aspect;
   END_REPEAT;
```

```
 WHERE
  WR1: (NOT cyclic_relationship);

 END_RULE; -- shape_aspect_relationship_attributes_references_are_distinct

 RULE shape_aspect_relationship_with_definable_object FOR (group,
         applied_classification_assignment);

  LOCAL
   clfied_inst : SET OF classification_item := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
   IF applied_classification_assignment[i].assigned_class.name =
      'definable object' THEN
     clfied_inst := clfied_inst + applied_classification_assignment[i].
        items;
   END_IF;
  END_REPEAT;

 WHERE
  WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

 END_RULE; -- shape_aspect_relationship_with_definable_object

 RULE shape_aspect_relationship_with_definition_matches_pattern FOR (
         group, applied_classification_assignment);

  LOCAL
   clfied_inst : SET OF classification_item := [];
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
   IF applied_classification_assignment[i].assigned_class.name =
      'DEFINITION' THEN
     clfied_inst := clfied_inst + applied_classification_assignment[i].
        items;
   END_IF;
  END_REPEAT;

 WHERE
  WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

 END_RULE; -- shape_aspect_relationship_with_definition_matches_pattern

 RULE shape_aspect_relationship_with_identification_assignment FOR (
         applied_classification_assignment);

  LOCAL
   violation : LOGICAL := FALSE;
   t1_set    : SET OF shape_aspect_relationship := [];
   c_a_set   : SET OF applied_classification_assignment := [];
```



```
  arg_list  : LIST OF STRING := ['composite feature',
              'seam curve relationship'];
  t2_set    : SET OF shape_aspect_relationship := [];
 END_LOCAL;
 REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = arg_list[j]) );
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  t2_set := bag_to_set(USEDIN(t1_set[i],
     'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
  violation := NOT (SIZEOF(t2_set) = 1);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- shape_aspect_relationship_with_identification_assignment

RULE shape_aspect_with_definable_object_matches_pattern FOR (group,
        applied_classification_assignment);

 LOCAL
  clfied_inst : SET OF classification_item := [];
 END_LOCAL;
 REPEAT i := 1 TO HIINDEX(applied_classification_assignment) BY 1;
  IF applied_classification_assignment[i].assigned_class.name =
     'definable object' THEN
    clfied_inst := clfied_inst + applied_classification_assignment[i].
       items;
  END_IF;
 END_REPEAT;

WHERE
 WR1: (SIZEOF(QUERY ( i <* clfied_inst | (NOT (i.id LIKE '*.*')) )) = 0);

END_RULE; -- shape_aspect_with_definable_object_matches_pattern

RULE shape_aspect_with_identification_assignment FOR (
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF shape_aspect := [];
  c_a_set   : SET OF applied_classification_assignment := [];
```

```
    arg_list  : LIST OF STRING := ['composite feature','seam',
                'structural cutout'];
    t2_set    : SET OF shape_aspect := [];
  END_LOCAL;
  REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
    c_a_set := QUERY ( i <* applied_classification_assignment | (i.
       assigned_class.name = arg_list[j]) );
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- shape_aspect_with_identification_assignment

 RULE ship_designation_has_one_specified_names FOR (product_definition,
          applied_identification_assignment, group,
          applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF product_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF applied_identification_assignment := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'ship designation') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(QUERY ( t2_inst <* t2_set | ((t2_inst.role.
       name = 'imo number') OR (t2_inst.role.name =
       'pennant hull number')) )) = 1);
  END_REPEAT;

 WHERE
```



```
  WR1: (NOT violation);

END_RULE; -- ship_designation_has_one_specified_names

RULE ship_material_property_has_at_most_one_named_references FOR (
        property_definition, applied_document_reference,
        applied_classification_assignment);

  LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF property_definition := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF applied_document_reference := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'ship material property') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],'SHIP_STRUCTURES_SCHEMA.' +
       'APPLIED_DOCUMENT_REFERENCE.ITEMS'));
    violation := SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.name =
       'material reference') )) > 1;
  END_REPEAT;

WHERE
  WR1: (NOT violation);

END_RULE; -- ship_material_property_has_at_most_one_named_references

RULE source_for_library_definition FOR (representation,
        representation_item, group, applied_classification_assignment);

  LOCAL
    violation      : LOGICAL := FALSE;
    c_a_set        : SET OF applied_classification_assignment := [];
    arg_list       : LIST OF STRING := ['library definition'];
    classed_rep_set : SET OF representation := [];
  END_LOCAL;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = 'source') );
  REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
      classed_rep_set := classed_rep_set + c_a_set[i].items[j];
    END_REPEAT;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(classed_rep_set) BY 1 WHILE NOT violation;
```

```
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
     violation := SIZEOF(QUERY ( rep_item <* classed_rep_set[i].items
         | (rep_item.name = arg_list[j]) )) <> 1;
    END_REPEAT;
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- source_for_library_definition

  RULE spacing_position_compound_representation_has_name FOR (
          compound_representation_item, group,
          applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF compound_representation_item := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    arg_list  : LIST OF STRING := ['position number','position'];
    t2_set    : SET OF representation_item := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'spacing position') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
     t2_set := t1_set[i].item_element;
     violation := SIZEOF(QUERY ( items <* t2_set | (items.name =
        arg_list[j]) )) <> 1;
    END_REPEAT;
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- spacing_position_compound_representation_has_name

  RULE spacing_position_with_offset_compound_representation_has_class FOR (
          compound_representation_item, group,
          applied_classification_assignment);

   LOCAL
    t3_set    : SET OF representation_item := [];
    violation : LOGICAL := FALSE;
    t1_set    : SET OF compound_representation_item := [];
```



```
  c_a_set   : SET OF applied_classification_assignment := [];
  c_a_set2  : SET OF applied_classification_assignment := [];
  t2_set    : SET OF compound_representation_item := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'spacing position with offset') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 c_a_set2 := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'spacing position') );
 REPEAT i := 1 TO HIINDEX(c_a_set2) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set2[i].items) BY 1;
   t2_set := t2_set + c_a_set2[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
  REPEAT j := 1 TO HIINDEX(t1_set[i].item_element) BY 1;
   t3_set := t3_set + t1_set[i].item_element;
  END_REPEAT;
  violation := SIZEOF(t3_set * t2_set) <> 1;
  t3_set := [];
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- spacing_position_with_offset_compound_representation_has_class

RULE spacing_position_with_offset_compound_representation_has_name FOR (
        compound_representation_item, group,
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF compound_representation_item := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  arg_list  : LIST OF STRING := ['offset'];
  t2_set    : SET OF representation_item := [];
 END_LOCAL;
 c_a_set := QUERY ( i <* applied_classification_assignment | (i.
    assigned_class.name = 'spacing position with offset') );
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
```

```
    REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
     t2_set := t1_set[i].item_element;
     violation := SIZEOF(QUERY ( items <* t2_set | (items.name =
        arg_list[j]) )) <> 1;
    END_REPEAT;
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- spacing_position_with_offset_compound_representation_has_name

  RULE structural_class_rejection_has_at_least_one_role_references FOR (
          group, applied_document_reference,
          applied_classification_assignment);

   LOCAL
    violation : LOGICAL := FALSE;
    t1_set    : SET OF group := [];
    c_a_set   : SET OF applied_classification_assignment := [];
    t2_set    : SET OF applied_document_reference := [];
   END_LOCAL;
   c_a_set := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'structural class rejection') );
   REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
    REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
     t1_set := t1_set + c_a_set[i].items[j];
    END_REPEAT;
   END_REPEAT;
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_DOCUMENT_REFERENCE.ITEMS'));
    violation := SIZEOF(QUERY ( t2_inst <* t2_set | (t2_inst.role.name =
        'explanations') )) < 1;
   END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- structural_class_rejection_has_at_least_one_role_references

  RULE structural_part_symmetry_relationship_has_product_definition FOR (
          group, applied_classification_assignment);

   LOCAL
    cl_ass      : SET OF applied_classification_assignment := [];
    clfied_inst : SET OF product_definition_relationship := [];
   END_LOCAL;
   cl_ass := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'structural part symmetry relationship') );
```



```
  REPEAT i := 1 TO HIINDEX(cl_ass) BY 1;
    clfied_inst := clfied_inst + cl_ass[i].items;
  END_REPEAT;

WHERE
  WR1: ((SIZEOF(QUERY ( pdr <* clfied_inst | (NOT (SIZEOF(which_class(
         pdr.relating_product_definition)) > 0)) )) - SIZEOF(
         QUERY ( pdr <* clfied_inst | (NOT (SIZEOF(which_class(pdr.
         related_product_definition)) > 0)) ))) = 0);

END_RULE; -- structural_part_symmetry_relationship_has_product_definition

RULE structural_system_symmetry_relationship_has_product_definition FOR (
         group, applied_classification_assignment);

  LOCAL
    cl_ass      : SET OF applied_classification_assignment := [];
    clfied_inst : SET OF product_definition_relationship := [];
  END_LOCAL;
  cl_ass := QUERY ( i <* applied_classification_assignment | (i.
      assigned_class.name = 'structural system symmetry relationship') );
  REPEAT i := 1 TO HIINDEX(cl_ass) BY 1;
    clfied_inst := clfied_inst + cl_ass[i].items;
  END_REPEAT;

WHERE
  WR1: ((SIZEOF(QUERY ( pdr <* clfied_inst | (NOT (SIZEOF(which_class(
         pdr.relating_product_definition)) > 0)) )) - SIZEOF(
         QUERY ( pdr <* clfied_inst | (NOT (SIZEOF(which_class(pdr.
         related_product_definition)) > 0)) ))) = 0);

END_RULE; -- structural_system_symmetry_relationship_has_product_definition

RULE unique_approvals_in_approval_history FOR (group,
         applied_group_assignment);

  LOCAL
    violate : LOGICAL := FALSE;
    t3_set  : SET OF approval := [];
    t1_set  : SET OF group := [];
    t2_set  : SET OF applied_group_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( i <* group | VALUE_IN(which_class(i),
      'APPROVAL HISTORY') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    t2_set := QUERY ( a <* applied_group_assignment | (a.assigned_group
        = t1_set[i]) );
    t3_set := QUERY ( b <* t2_set[1].items | (
        'SHIP_STRUCTURES_SCHEMA.APPROVAL' IN TYPEOF(b)) );
    violate := NOT VALUE_UNIQUE(t3_set);
```

```
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- unique_approvals_in_approval_history

 RULE user_def_function_description_required FOR (representation,
         representation_item);

  LOCAL
   violation : LOGICAL := FALSE;
  END_LOCAL;
  REPEAT i := 1 TO HIINDEX(representation) BY 1 WHILE NOT violation;
   violation := (SIZEOF(QUERY ( r <* representation[i].items | ((
      'SHIP_STRUCTURES_SCHEMA.DESCRIPTIVE_REPRESENTATION_ITEM' IN
      TYPEOF(r)) AND (r.name = 'function') AND (r\
      descriptive_representation_item.description = 'user defined')) ))
      > 0) AND (SIZEOF(QUERY ( r <* representation[i].items | (r.name =
       'user def function') )) = 0);
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- user_def_function_description_required

 RULE user_defined_requires_user_defined_class FOR (product_definition,
         group, applied_classification_assignment);

  LOCAL
   violation   : LOGICAL := FALSE;
   cl_ass      : SET OF applied_classification_assignment := [];
   clfied_inst : SET OF classification_item := [];
  END_LOCAL;
  cl_ass := QUERY ( i <* applied_classification_assignment | ((i.role.
     name = 'class membership') AND (i.assigned_class.name =
     'user defined')) );
  REPEAT i := 1 TO HIINDEX(cl_ass) BY 1;
   clfied_inst := clfied_inst + cl_ass[i].items;
  END_REPEAT;
  REPEAT i := 1 TO HIINDEX(clfied_inst) BY 1 WHILE NOT violation;
   violation := NOT ('user defined class' IN which_class(clfied_inst[i]));
  END_REPEAT;

 WHERE
  WR1: (NOT violation);

 END_RULE; -- user_defined_requires_user_defined_class
```



```
RULE user_defined_tightness_description_required FOR (representation,
        representation_item);

 LOCAL
  violation : LOGICAL := FALSE;
 END_LOCAL;
 REPEAT i := 1 TO HIINDEX(representation) BY 1 WHILE NOT violation;
  violation := (SIZEOF(QUERY ( r <* representation[i].items | ((
     'SHIP_STRUCTURES_SCHEMA.DESCRIPTIVE_REPRESENTATION_ITEM' IN
     TYPEOF(r)) AND (r.name = 'tightness') AND (r\
     descriptive_representation_item.description = 'user defined')) ))
     > 0) AND (SIZEOF(QUERY ( r <* representation[i].items | (r.name =
      'user defined tightness') )) = 0);
 END_REPEAT;

WHERE
 WR1: (NOT violation);

END_RULE; -- user_defined_tightness_description_required

RULE version_creation_has_mandatory_attribute_description FOR (action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action := [];
 END_LOCAL;
 t1_set := QUERY ( i <* action | VALUE_IN(which_class(i),
    'version creation') );
 violate := SIZEOF(QUERY ( k <* t1_set | (NOT EXISTS(k.description)) ))
    > 0;

WHERE
 WR1: (NOT violate);

END_RULE; -- version_creation_has_mandatory_attribute_description

RULE version_deletion_has_mandatory_attribute_description FOR (action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action := [];
 END_LOCAL;
 t1_set := QUERY ( i <* action | VALUE_IN(which_class(i),
    'version deletion') );
 violate := SIZEOF(QUERY ( k <* t1_set | (NOT EXISTS(k.description)) ))
    > 0;

WHERE
 WR1: (NOT violate);
```

```
 END_RULE; -- version_deletion_has_mandatory_attribute_description

 RULE version_history_has_exactly_one_assigned_group FOR (
         applied_group_assignment, group);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF group := [];
    set_1   : SET OF applied_group_assignment := [];
    set_3   : SET OF group_item := [];
    set_2   : SET OF applied_group_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* group | VALUE_IN(which_class(a),
     'version history') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    set_1 := QUERY ( b <* applied_group_assignment | ((b.assigned_group
       = t1_set[i]) AND (b.role.name = 'current version')) );
    set_2 := QUERY ( c <* applied_group_assignment | ((c.assigned_group
       = t1_set[i]) AND (c.role.name = 'members')) );
    violate := (SIZEOF(set_1) <> 1) OR (SIZEOF(set_2) <> 1);
    IF NOT violate THEN
      set_3 := set_1[1].items * set_2[1].items;
      violate := (SIZEOF(set_3) <> 1) OR (NOT VALUE_IN(which_class(set_3
         [1]),'versionable object'));
    END_IF;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- version_history_has_exactly_one_assigned_group

 RULE version_history_referenced_by_exactly_one_current_version FOR (
         applied_group_assignment, group);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
  END_LOCAL;
  t1_set := QUERY ( a <* group | VALUE_IN(which_class(a),
     'version history') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_group_assignment | ((b.assigned_group
       = t1_set[i]) AND (b.role.name = 'current version')) );
    violate := SIZEOF(a_set) <> 1;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);
```



```
END_RULE; -- version_history_referenced_by_exactly_one_current_version

RULE version_history_referenced_by_multiple_roles FOR (
        applied_group_assignment, group);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF group := [];
  a_set   : SET OF applied_group_assignment := [];
 END_LOCAL;
 t1_set := QUERY ( a <* group | VALUE_IN(which_class(a),
    'version history') );
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
  a_set := QUERY ( b <* applied_group_assignment | ((b.assigned_group
     = t1_set[i]) AND (NOT (b.role.name IN ['versions',
     'current version','relationships'])))  );
  violate := SIZEOF(a_set) > 0;
 END_REPEAT;

WHERE
 WR1: (NOT violate);

END_RULE; -- version_history_referenced_by_multiple_roles

RULE version_modification_has_mandatory_attribute_description FOR (
        action);

 LOCAL
  violate : LOGICAL := FALSE;
  t1_set  : SET OF action := [];
 END_LOCAL;
 t1_set := QUERY ( i <* action | VALUE_IN(which_class(i),
    'version modification') );
 violate := SIZEOF(QUERY ( k <* t1_set | (NOT EXISTS(k.description)) ))
    > 0;

WHERE
 WR1: (NOT violate);

END_RULE; -- version_modification_has_mandatory_attribute_description

RULE version_relationship_associates_with_versionable_object FOR (
        applied_identification_assignment);

 LOCAL
  violate  : LOGICAL := FALSE;
  violate1 : LOGICAL := FALSE;
  violate2 : LOGICAL := FALSE;
 END_LOCAL;
 REPEAT i := 1 TO HIINDEX(applied_identification_assignment) BY 1
```

```
      WHILE NOT violate;
    violate1 := (SIZEOF(USEDIN(applied_identification_assignment[i],
      'SHIP_STRUCTURES_SCHEMA.IDENTIFICATION_ASSIGNMENT_RELATIONSHIP.'
      + 'RELATING_IDENTIFICATION_ASSIGNMENT')) > 0) AND (NOT VALUE_IN(
      which_class(applied_identification_assignment[i]),
      'versionable object'));
    violate2 := (SIZEOF(USEDIN(applied_identification_assignment[i],
      'SHIP_STRUCTURES_SCHEMA.IDENTIFICATION_ASSIGNMENT_RELATIONSHIP.'
      + 'RELATED_IDENTIFICATION_ASSIGNMENT')) > 0) AND (NOT VALUE_IN(
      which_class(applied_identification_assignment[i]),
      'versionable object'));
    violate := violate1 OR violate2;
  END_REPEAT;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- version_relationship_associates_with_versionable_object

 RULE version_relationship_has_mandatory_attribute_description FOR (
         identification_assignment_relationship);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF identification_assignment_relationship := [];
  END_LOCAL;
  t1_set := QUERY ( i <* identification_assignment_relationship |
     VALUE_IN(which_class(i),'version relationship') );
  violate := SIZEOF(QUERY ( k <* t1_set | (NOT EXISTS(k.description)) ))
     > 0;

 WHERE
  WR1: (NOT violate);

 END_RULE; -- version_relationship_has_mandatory_attribute_description

 RULE version_relationship_has_unique_versions FOR (
         identification_assignment_relationship);

  LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF identification_assignment_relationship := [];
  END_LOCAL;
  t1_set := QUERY ( a <* identification_assignment_relationship |
     VALUE_IN(which_class(a),'version relationship') );
  REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    violate := t1_set[i].relating_identification_assignment.assigned_id
       = t1_set[i].related_identification_assignment.assigned_id;
  END_REPEAT;
```



```
WHERE
 WR1: (NOT violate);

END_RULE; -- version_relationship_has_unique_versions

RULE versionable_object_has_one_version_id FOR (
        applied_identification_assignment);

 LOCAL
  version_ids        : SET OF applied_identification_assignment := [];
  duplicate          : LOGICAL := FALSE;
  versionable_objects : BAG OF identification_item := [];
 END_LOCAL;
 version_ids := QUERY ( i <* applied_identification_assignment | (i.
    role.name = 'version identifier') );
 REPEAT i := 1 TO HIINDEX(version_ids) BY 1;
  versionable_objects := versionable_objects + version_ids[i].items;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(versionable_objects) BY 1 WHILE NOT duplicate;
  REPEAT j := i + 1 TO HIINDEX(versionable_objects) BY 1 WHILE NOT
     duplicate;
    duplicate := versionable_objects[i] :=: versionable_objects[j];
  END_REPEAT;
 END_REPEAT;

WHERE
 WR1: (NOT duplicate);

END_RULE; -- versionable_object_has_one_version_id

RULE versioned_action_request_with_identification_assignment FOR (
        applied_classification_assignment);

 LOCAL
  violation : LOGICAL := FALSE;
  t1_set    : SET OF versioned_action_request := [];
  c_a_set   : SET OF applied_classification_assignment := [];
  arg_list  : LIST OF STRING := ['definable object'];
  t2_set    : SET OF versioned_action_request := [];
 END_LOCAL;
 REPEAT j := 1 TO HIINDEX(arg_list) BY 1 WHILE NOT violation;
  c_a_set := QUERY ( i <* applied_classification_assignment | (i.
     assigned_class.name = arg_list[j]) );
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(c_a_set) BY 1;
  REPEAT j := 1 TO HIINDEX(c_a_set[i].items) BY 1;
   t1_set := t1_set + c_a_set[i].items[j];
  END_REPEAT;
 END_REPEAT;
 REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violation;
```

```
    t2_set := bag_to_set(USEDIN(t1_set[i],
       'SHIP_STRUCTURES_SCHEMA.APPLIED_IDENTIFICATION_ASSIGNMENT.ITEMS'));
    violation := NOT (SIZEOF(t2_set) = 1);
  END_REPEAT;

  WHERE
   WR1: (NOT violation);

  END_RULE; -- versioned_action_request_with_identification_assignment

  RULE versions_is_referenced_by_at_least_one_version_history FOR (
          applied_group_assignment, group);

   LOCAL
    violate : LOGICAL := FALSE;
    t1_set  : SET OF group := [];
    a_set   : SET OF applied_group_assignment := [];
   END_LOCAL;
   t1_set := QUERY ( a <* group | VALUE_IN(which_class(a),'versions') );
   REPEAT i := 1 TO HIINDEX(t1_set) BY 1 WHILE NOT violate;
    a_set := QUERY ( b <* applied_group_assignment | ((b.assigned_group
       = t1_set[i]) AND (b.role.name = 'version history')) );
    violate := SIZEOF(a_set) < 1;
   END_REPEAT;

  WHERE
   WR1: (NOT violate);

  END_RULE; -- versions_is_referenced_by_at_least_one_version_history

  FUNCTION acyclic_curve_replica(
            rep: curve_replica;
            parent: curve
    ): BOOLEAN;
   IF NOT ('SHIP_STRUCTURES_SCHEMA.CURVE_REPLICA' IN TYPEOF(parent))
       THEN
    RETURN(TRUE);
   END_IF;
   IF parent :=: rep THEN
    RETURN(FALSE);
   ELSE
    RETURN(acyclic_curve_replica(rep,parent\curve_replica.parent_curve));
   END_IF;

  END_FUNCTION; -- acyclic_curve_replica

  FUNCTION acyclic_mapped_representation(
            parent_set: SET OF representation;
            children_set: SET OF representation_item
    ): BOOLEAN;
```



```
  LOCAL
    x : SET OF representation_item;
    y : SET OF representation_item;
  END_LOCAL;
  x := QUERY ( z <* children_set | ('SHIP_STRUCTURES_SCHEMA.MAPPED_ITEM'
     IN TYPEOF(z)) );
  IF SIZEOF(x) > 0 THEN
    REPEAT i := 1 TO HIINDEX(x) BY 1;
      IF x[i]\mapped_item.mapping_source.mapped_representation IN
         parent_set THEN
        RETURN(FALSE);
      END_IF;
      IF NOT acyclic_mapped_representation(parent_set + x[i]\mapped_item
         .mapping_source.mapped_representation,x[i]\mapped_item.
         mapping_source.mapped_representation.items) THEN
        RETURN(FALSE);
      END_IF;
    END_REPEAT;
  END_IF;
  x := children_set - x;
  IF SIZEOF(x) > 0 THEN
    REPEAT i := 1 TO HIINDEX(x) BY 1;
      y := QUERY ( z <* bag_to_set(USEDIN(x[i],'')) | (
         'SHIP_STRUCTURES_SCHEMA.REPRESENTATION_ITEM' IN TYPEOF(z)) );
      IF NOT acyclic_mapped_representation(parent_set,y) THEN
        RETURN(FALSE);
      END_IF;
    END_REPEAT;
  END_IF;
  RETURN(TRUE);

END_FUNCTION; -- acyclic_mapped_representation

FUNCTION acyclic_point_replica(
           rep: point_replica;
           parent: point
   ): BOOLEAN;
  IF NOT ('SHIP_STRUCTURES_SCHEMA.POINT_REPLICA' IN TYPEOF(parent))
      THEN
    RETURN(TRUE);
  END_IF;
  IF parent :=: rep THEN
    RETURN(FALSE);
  ELSE
    RETURN(acyclic_point_replica(rep,parent\point_replica.parent_pt));
  END_IF;

END_FUNCTION; -- acyclic_point_replica

FUNCTION acyclic_product_category_relationship(
```

```
            relation: product_category_relationship;
            children: SET OF product_category
    ): BOOLEAN;

   LOCAL
    x            : SET OF product_category_relationship;
    local_children : SET OF product_category;
   END_LOCAL;
   REPEAT i := 1 TO HIINDEX(children) BY 1;
    IF relation.category :=: children[i] THEN
     RETURN(FALSE);
    END_IF;
   END_REPEAT;
   x := bag_to_set(USEDIN(relation.category,'SHIP_STRUCTURES_SCHEMA.' +
      'PRODUCT_CATEGORY_RELATIONSHIP.SUB_CATEGORY'));
   local_children := children + relation.category;
   IF SIZEOF(x) > 0 THEN
    REPEAT i := 1 TO HIINDEX(x) BY 1;
     IF NOT acyclic_product_category_relationship(x[i],local_children)
         THEN
      RETURN(FALSE);
     END_IF;
    END_REPEAT;
   END_IF;
   RETURN(TRUE);

  END_FUNCTION; -- acyclic_product_category_relationship

  FUNCTION acyclic_surface_replica(
            rep: surface_replica;
            parent: surface
    ): BOOLEAN;
   IF NOT ('SHIP_STRUCTURES_SCHEMA.SURFACE_REPLICA' IN TYPEOF(parent))
       THEN
    RETURN(TRUE);
   END_IF;
   IF parent :=: rep THEN
    RETURN(FALSE);
   ELSE
    RETURN(acyclic_surface_replica(rep,parent\surface_replica.
       parent_surface));
   END_IF;

  END_FUNCTION; -- acyclic_surface_replica

  FUNCTION associated_surface(
            arg: pcurve_or_surface
    ): surface;

   LOCAL
```



```
    surf : surface;
  END_LOCAL;
  IF 'SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(arg) THEN
    surf := arg.basis_surface;
  ELSE
    surf := arg;
  END_IF;
  RETURN(surf);

END_FUNCTION; -- associated_surface

FUNCTION bag_to_set(
           the_bag: BAG OF GENERIC:intype
   ): SET OF GENERIC:intype;

  LOCAL
    the_set : SET OF GENERIC:intype := [];
  END_LOCAL;
  IF SIZEOF(the_bag) > 0 THEN
    REPEAT i := 1 TO HIINDEX(the_bag) BY 1;
      the_set := the_set + the_bag[i];
    END_REPEAT;
  END_IF;
  RETURN(the_set);

END_FUNCTION; -- bag_to_set

FUNCTION base_axis(
           dim: INTEGER;
           axis1, axis2, axis3: direction
   ): LIST [2:3] OF direction;

  LOCAL
    u      : LIST [2:3] OF direction;
    d1     : direction;
    d2     : direction;
    factor : REAL;
  END_LOCAL;
  IF dim = 3 THEN
    d1 := NVL(normalise(axis3),dummy_gri || direction([0,0,1]));
    d2 := first_proj_axis(d1,axis1);
    u := [d2,second_proj_axis(d1,d2,axis2),d1];
  ELSE
    IF EXISTS(axis1) THEN
      d1 := normalise(axis1);
      u := [d1,orthogonal_complement(d1)];
      IF EXISTS(axis2) THEN
        factor := dot_product(axis2,u[2]);
        IF factor < 0 THEN
          u[2].direction_ratios[1] := -u[2].direction_ratios[1];
```

```
        u[2].direction_ratios[2] := -u[2].direction_ratios[2];
       END_IF;
      END_IF;
     ELSE
      IF EXISTS(axis2) THEN
       d1 := normalise(axis2);
       u := [orthogonal_complement(d1),d1];
       u[1].direction_ratios[1] := -u[1].direction_ratios[1];
       u[1].direction_ratios[2] := -u[1].direction_ratios[2];
      ELSE
       u := [dummy_gri || direction([1,0]),dummy_gri || direction([0,1])];
      END_IF;
     END_IF;
    END_IF;
    RETURN(u);

  END_FUNCTION; -- base_axis

  FUNCTION boolean_choose(
            b: BOOLEAN;
            choice1: GENERIC:item;
            choice2: GENERIC:item
    ): GENERIC:item;
    IF b THEN
     RETURN(choice1);
    ELSE
     RETURN(choice2);
    END_IF;

  END_FUNCTION; -- boolean_choose

  FUNCTION build_2axes(
            ref_direction: direction
    ): LIST [2:2] OF direction;

    LOCAL
     d : direction := NVL(normalise(ref_direction),dummy_gri ||
         direction([1,0]));
    END_LOCAL;
    RETURN([d,orthogonal_complement(d)]);

  END_FUNCTION; -- build_2axes

  FUNCTION build_axes(
            axis, ref_direction: direction
    ): LIST [3:3] OF direction;

    LOCAL
     d1 : direction;
     d2 : direction;
```



```
  END_LOCAL;
  d1 := NVL(normalise(axis),dummy_gri || direction([0,0,1]));
  d2 := first_proj_axis(d1,ref_direction);
  RETURN([d2,normalise(cross_product(d1,d2)).orientation,d1]);

END_FUNCTION; -- build_axes

FUNCTION closed_shell_reversed(
          a_shell: closed_shell
   ): oriented_closed_shell;

  LOCAL
   the_reverse : oriented_closed_shell;
  END_LOCAL;
  IF 'SHIP_STRUCTURES_SCHEMA.ORIENTED_CLOSED_SHELL' IN TYPEOF(a_shell)
     THEN
   the_reverse := dummy_tri || connected_face_set(a_shell\
      connected_face_set.cfs_faces) || closed_shell() ||
      oriented_closed_shell(a_shell\oriented_closed_shell.
      closed_shell_element,NOT a_shell\oriented_closed_shell.
      orientation);
  ELSE
   the_reverse := dummy_tri || connected_face_set(a_shell\
      connected_face_set.cfs_faces) || closed_shell() ||
      oriented_closed_shell(a_shell,FALSE);
  END_IF;
  RETURN(the_reverse);

END_FUNCTION; -- closed_shell_reversed

FUNCTION conditional_reverse(
          p: BOOLEAN;
          an_item: reversible_topology
   ): reversible_topology;
  IF p THEN
   RETURN(an_item);
  ELSE
   RETURN(topology_reversed(an_item));
  END_IF;

END_FUNCTION; -- conditional_reverse

FUNCTION constraints_composite_curve_on_surface(
          c: composite_curve_on_surface
   ): BOOLEAN;

  LOCAL
   n_segments : INTEGER := SIZEOF(c.segments);
  END_LOCAL;
  REPEAT k := 1 TO n_segments BY 1;
```

```
    IF (NOT ('SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(c\composite_curve
       .segments[k].parent_curve))) AND (NOT (
       'SHIP_STRUCTURES_SCHEMA.SURFACE_CURVE' IN TYPEOF(c\
       composite_curve.segments[k].parent_curve))) AND (NOT (
       'SHIP_STRUCTURES_SCHEMA.COMPOSITE_CURVE_ON_SURFACE' IN TYPEOF(c\
       composite_curve.segments[k].parent_curve))) THEN
      RETURN(FALSE);
    END_IF;
  END_REPEAT;
  RETURN(TRUE);

 END_FUNCTION; -- constraints_composite_curve_on_surface

 FUNCTION constraints_param_b_spline(
           degree: INTEGER;
           up_knots: INTEGER;
           up_cp: INTEGER;
           knot_mult: LIST OF INTEGER;
           knots: LIST OF parameter_value
    ): BOOLEAN;

  LOCAL
    k      : INTEGER;
    sum    : INTEGER;
    result : BOOLEAN := TRUE;
  END_LOCAL;
  sum := knot_mult[1];
  REPEAT i := 2 TO up_knots BY 1;
    sum := sum + knot_mult[i];
  END_REPEAT;
  IF (degree < 1) OR (up_knots < 2) OR (up_cp < degree) OR (sum <> (
      degree + up_cp + 2)) THEN
    result := FALSE;
    RETURN(result);
  END_IF;
  k := knot_mult[1];
  IF (k < 1) OR (k > (degree + 1)) THEN
    result := FALSE;
    RETURN(result);
  END_IF;
  REPEAT i := 2 TO up_knots BY 1;
    IF (knot_mult[i] < 1) OR (knots[i] <= knots[i - 1]) THEN
      result := FALSE;
      RETURN(result);
    END_IF;
    k := knot_mult[i];
    IF (i < up_knots) AND (k > degree) THEN
      result := FALSE;
      RETURN(result);
    END_IF;
```



```
    IF (i = up_knots) AND (k > (degree + 1)) THEN
      result := FALSE;
      RETURN(result);
    END_IF;
  END_REPEAT;
  RETURN(result);

END_FUNCTION; -- constraints_param_b_spline

FUNCTION cross_product(
          arg1, arg2: direction
    ): vector;

  LOCAL
    v2     : LIST [3:3] OF REAL;
    v1     : LIST [3:3] OF REAL;
    mag    : REAL;
    res    : direction;
    result : vector;
  END_LOCAL;
  IF (NOT EXISTS(arg1)) OR (arg1.dim = 2) OR (NOT EXISTS(arg2)) OR (arg2
     .dim = 2) THEN
    RETURN(?);
  ELSE
    BEGIN
      v1 := normalise(arg1).direction_ratios;
      v2 := normalise(arg2).direction_ratios;
      res := dummy_gri || direction([(v1[2] * v2[3]) - (v1[3] * v2[2]),(
         v1[3] * v2[1]) - (v1[1] * v2[3]),(v1[1] * v2[2]) - (v1[2] * v2[
         1])]);
      mag := 0;
      REPEAT i := 1 TO 3 BY 1;
        mag := mag + (res.direction_ratios[i] * res.direction_ratios[i]);
      END_REPEAT;
      IF mag > 0 THEN
        result := dummy_gri || vector(res,SQRT(mag));
      ELSE
        result := dummy_gri || vector(arg1,0);
      END_IF;
      RETURN(result);
    END;
  END_IF;

END_FUNCTION; -- cross_product

FUNCTION curve_weights_positive(
          b: rational_b_spline_curve
    ): BOOLEAN;

  LOCAL
```

```
    result : BOOLEAN := TRUE;
  END_LOCAL;
  REPEAT i := 0 TO b.upper_index_on_control_points BY 1;
   IF b.weights[i] <= 0 THEN
    result := FALSE;
    RETURN(result);
   END_IF;
  END_REPEAT;
  RETURN(result);

 END_FUNCTION; -- curve_weights_positive

 FUNCTION derive_dimensional_exponents(
           x: unit
   ): dimensional_exponents;

  LOCAL
   result : dimensional_exponents := dimensional_exponents(0,0,0,0,0,0,
            0);
  END_LOCAL;
  IF 'SHIP_STRUCTURES_SCHEMA.DERIVED_UNIT' IN TYPEOF(x) THEN
   REPEAT i := LOINDEX(x.elements) TO HIINDEX(x.elements) BY 1;
    result.length_exponent := result.length_exponent + (x.elements[i].
       exponent * x.elements[i].unit.dimensions.length_exponent);
    result.mass_exponent := result.mass_exponent + (x.elements[i].
       exponent * x.elements[i].unit.dimensions.mass_exponent);
    result.time_exponent := result.time_exponent + (x.elements[i].
       exponent * x.elements[i].unit.dimensions.time_exponent);
    result.electric_current_exponent := result.
       electric_current_exponent + (x.elements[i].exponent * x.
       elements[i].unit.dimensions.electric_current_exponent);
    result.thermodynamic_temperature_exponent := result.
       thermodynamic_temperature_exponent + (x.elements[i].exponent *
       x.elements[i].unit.dimensions.
       thermodynamic_temperature_exponent);
    result.amount_of_substance_exponent := result.
       amount_of_substance_exponent + (x.elements[i].exponent * x.
       elements[i].unit.dimensions.amount_of_substance_exponent);
    result.luminous_intensity_exponent := result.
       luminous_intensity_exponent + (x.elements[i].exponent * x.
       elements[i].unit.dimensions.luminous_intensity_exponent);
   END_REPEAT;
  ELSE
   result := x.dimensions;
  END_IF;
  RETURN(result);

 END_FUNCTION; -- derive_dimensional_exponents

 FUNCTION dimension_of(
```



```
        item: geometric_representation_item
  ): dimension_count;

  LOCAL
    x   : SET OF representation;
    y   : representation_context;
    dim : dimension_count;
  END_LOCAL;
  IF 'SHIP_STRUCTURES_SCHEMA.CARTESIAN_POINT' IN TYPEOF(item) THEN
    dim := SIZEOF(item\cartesian_point.coordinates);
    RETURN(dim);
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.DIRECTION' IN TYPEOF(item) THEN
    dim := SIZEOF(item\direction.direction_ratios);
    RETURN(dim);
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.VECTOR' IN TYPEOF(item) THEN
    dim := SIZEOF(item\vector.orientation\direction.direction_ratios);
    RETURN(dim);
  END_IF;
  x := using_representations(item);
  y := x[1].context_of_items;
  dim := y\geometric_representation_context.coordinate_space_dimension;
  RETURN(dim);

END_FUNCTION; -- dimension_of

FUNCTION dimensions_for_si_unit(
        n: si_unit_name
  ): dimensional_exponents;
  CASE n OF
    metre      :  RETURN(dimensional_exponents(1,0,0,0,0,0,0));
    gram       :  RETURN(dimensional_exponents(0,1,0,0,0,0,0));
    second     :  RETURN(dimensional_exponents(0,0,1,0,0,0,0));
    ampere     :  RETURN(dimensional_exponents(0,0,0,1,0,0,0));
    kelvin     :  RETURN(dimensional_exponents(0,0,0,0,1,0,0));
    mole       :  RETURN(dimensional_exponents(0,0,0,0,0,1,0));
    candela    :  RETURN(dimensional_exponents(0,0,0,0,0,0,1));
    radian     :  RETURN(dimensional_exponents(0,0,0,0,0,0,0));
    steradian  :  RETURN(dimensional_exponents(0,0,0,0,0,0,0));
    hertz      :  RETURN(dimensional_exponents(0,0,-1,0,0,0,0));
    newton     :  RETURN(dimensional_exponents(1,1,-2,0,0,0,0));
    pascal     :  RETURN(dimensional_exponents(-1,1,-2,0,0,0,0));
    joule      :  RETURN(dimensional_exponents(2,1,-2,0,0,0,0));
    watt       :  RETURN(dimensional_exponents(2,1,-3,0,0,0,0));
    coulomb    :  RETURN(dimensional_exponents(0,0,1,1,0,0,0));
    volt       :  RETURN(dimensional_exponents(2,1,-3,-1,0,0,0));
    farad      :  RETURN(dimensional_exponents(-2,-1,4,1,0,0,0));
    ohm        :  RETURN(dimensional_exponents(2,1,-3,-2,0,0,0));
    siemens    :  RETURN(dimensional_exponents(-2,-1,3,2,0,0,0));
```

```
    weber       :       RETURN(dimensional_exponents(2,1,-2,-1,0,0,0));
    tesla       :       RETURN(dimensional_exponents(0,1,-2,-1,0,0,0));
    henry       :       RETURN(dimensional_exponents(2,1,-2,-2,0,0,0));
    degree_celsius :    RETURN(dimensional_exponents(0,0,0,0,1,0,0));
    lumen       :       RETURN(dimensional_exponents(0,0,0,0,0,0,1));
    lux         :       RETURN(dimensional_exponents(-2,0,0,0,0,0,1));
    becquerel   :       RETURN(dimensional_exponents(0,0,-1,0,0,0,0));
    gray        :       RETURN(dimensional_exponents(2,0,-2,0,0,0,0));
    sievert     :       RETURN(dimensional_exponents(2,0,-2,0,0,0,0));
    OTHERWISE   :       RETURN(?);
    END_CASE;

 END_FUNCTION; -- dimensions_for_si_unit

 FUNCTION dot_product(
            arg1, arg2: direction
    ): REAL;

   LOCAL
    ndim   : INTEGER;
    scalar : REAL;
    vec1   : direction;
    vec2   : direction;
   END_LOCAL;
   IF (NOT EXISTS(arg1)) OR (NOT EXISTS(arg2)) THEN
    scalar := ?;
   ELSE
    IF arg1.dim <> arg2.dim THEN
     scalar := ?;
    ELSE
     BEGIN
      vec1 := normalise(arg1);
      vec2 := normalise(arg2);
      ndim := arg1.dim;
      scalar := 0;
      REPEAT i := 1 TO ndim BY 1;
       scalar := scalar + (vec1.direction_ratios[i] * vec2.
          direction_ratios[i]);
      END_REPEAT;
     END;
    END_IF;
   END_IF;
   RETURN(scalar);

 END_FUNCTION; -- dot_product

 FUNCTION edge_reversed(
            an_edge: edge
    ): oriented_edge;
```



```
  LOCAL
    the_reverse : oriented_edge;
  END_LOCAL;
  IF 'SHIP_STRUCTURES_SCHEMA.ORIENTED_EDGE' IN TYPEOF(an_edge) THEN
    the_reverse := dummy_tri || edge(an_edge.edge_end,an_edge.edge_start)
       || oriented_edge(an_edge\oriented_edge.edge_element,NOT an_edge\
       oriented_edge.orientation);
  ELSE
    the_reverse := dummy_tri || edge(an_edge.edge_end,an_edge.edge_start)
       || oriented_edge(an_edge,FALSE);
  END_IF;
  RETURN(the_reverse);

END_FUNCTION; -- edge_reversed

FUNCTION face_bound_reversed(
           a_face_bound: face_bound
    ): face_bound;

  LOCAL
    the_reverse : face_bound;
  END_LOCAL;
  IF 'SHIP_STRUCTURES_SCHEMA.FACE_OUTER_BOUND' IN TYPEOF(a_face_bound)
      THEN
    the_reverse := dummy_tri || face_bound(a_face_bound\face_bound.bound,
       NOT a_face_bound\face_bound.orientation) || face_outer_bound();
  ELSE
    the_reverse := dummy_tri || face_bound(a_face_bound.bound,NOT
       a_face_bound.orientation);
  END_IF;
  RETURN(the_reverse);

END_FUNCTION; -- face_bound_reversed

FUNCTION face_reversed(
           a_face: face
    ): oriented_face;

  LOCAL
    the_reverse : oriented_face;
  END_LOCAL;
  IF 'SHIP_STRUCTURES_SCHEMA.ORIENTED_FACE' IN TYPEOF(a_face) THEN
    the_reverse := dummy_tri || face(set_of_topology_reversed(a_face.
       bounds)) || oriented_face(a_face\oriented_face.face_element,NOT
       a_face\oriented_face.orientation);
  ELSE
    the_reverse := dummy_tri || face(set_of_topology_reversed(a_face.
       bounds)) || oriented_face(a_face,FALSE);
  END_IF;
  RETURN(the_reverse);
```

```
END_FUNCTION; -- face_reversed

FUNCTION first_proj_axis(
            z_axis, arg: direction
  ): direction;

 LOCAL
  x_vec  : vector;
  v      : direction;
  z      : direction;
  x_axis : direction;
 END_LOCAL;
 IF NOT EXISTS(z_axis) THEN
  RETURN(?);
 ELSE
  z := normalise(z_axis);
  IF NOT EXISTS(arg) THEN
   IF z.direction_ratios <> [1,0,0] THEN
    v := dummy_gri || direction([1,0,0]);
   ELSE
    v := dummy_gri || direction([0,1,0]);
   END_IF;
  ELSE
   IF arg.dim <> 3 THEN
    RETURN(?);
   END_IF;
   IF cross_product(arg,z).magnitude = 0 THEN
    RETURN(?);
   ELSE
    v := normalise(arg);
   END_IF;
  END_IF;
  x_vec := scalar_times_vector(dot_product(v,z),z);
  x_axis := vector_difference(v,x_vec).orientation;
  x_axis := normalise(x_axis);
 END_IF;
 RETURN(x_axis);

END_FUNCTION; -- first_proj_axis

FUNCTION get_basis_surface(
            c: curve_on_surface
  ): SET [0:2] OF surface;

 LOCAL
  surfs : SET [0:2] OF surface;
  n     : INTEGER;
 END_LOCAL;
 surfs := [];
 IF 'SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(c) THEN
```



```
    surfs := [c\pcurve.basis_surface];
  ELSE
    IF 'SHIP_STRUCTURES_SCHEMA.SURFACE_CURVE' IN TYPEOF(c) THEN
      n := SIZEOF(c\surface_curve.associated_geometry);
      REPEAT i := 1 TO n BY 1;
        surfs := surfs + associated_surface(c\surface_curve.
            associated_geometry[i]);
      END_REPEAT;
    END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.COMPOSITE_CURVE_ON_SURFACE' IN TYPEOF(c)
      THEN
    n := SIZEOF(c\composite_curve.segments);
    surfs := get_basis_surface(c\composite_curve.segments[1].
        parent_curve);
    IF n > 1 THEN
      REPEAT i := 2 TO n BY 1;
        surfs := surfs * get_basis_surface(c\composite_curve.segments[i]
            .parent_curve);
      END_REPEAT;
    END_IF;
  END_IF;
  RETURN(surfs);

END_FUNCTION; -- get_basis_surface

FUNCTION get_description_value(
            obj: description_attribute_select
    ): text;

  LOCAL
    description_bag : BAG OF description_attribute := USEDIN(obj,
                  'SHIP_STRUCTURES_SCHEMA.' +
                  'DESCRIPTION_ATTRIBUTE.' + 'DESCRIBED_ITEM');
  END_LOCAL;
  IF SIZEOF(description_bag) = 1 THEN
    RETURN(description_bag[1].attribute_value);
  ELSE
    RETURN(?);
  END_IF;

END_FUNCTION; -- get_description_value

FUNCTION get_id_value(
            obj: id_attribute_select
    ): identifier;

  LOCAL
    id_bag : BAG OF id_attribute := USEDIN(obj,'SHIP_STRUCTURES_SCHEMA.'
          + 'ID_ATTRIBUTE.' + 'IDENTIFIED_ITEM');
```

```
  END_LOCAL;
  IF SIZEOF(id_bag) = 1 THEN
   RETURN(id_bag[1].attribute_value);
  ELSE
   RETURN(?);
  END_IF;

 END_FUNCTION; -- get_id_value

 FUNCTION get_name_value(
          obj: name_attribute_select
   ): label;

  LOCAL
   name_bag : BAG OF name_attribute := USEDIN(obj,
            'SHIP_STRUCTURES_SCHEMA.' + 'NAME_ATTRIBUTE.' +
            'NAMED_ITEM');
  END_LOCAL;
  IF SIZEOF(name_bag) = 1 THEN
   RETURN(name_bag[1].attribute_value);
  ELSE
   RETURN(?);
  END_IF;

 END_FUNCTION; -- get_name_value

 FUNCTION get_role(
          obj: role_select
   ): object_role;

  LOCAL
   role_bag : BAG OF role_association := USEDIN(obj,
            'SHIP_STRUCTURES_SCHEMA.' + 'ROLE_ASSOCIATION.' +
            'ITEM_WITH_ROLE');
  END_LOCAL;
  IF SIZEOF(role_bag) = 1 THEN
   RETURN(role_bag[1].role);
  ELSE
   RETURN(?);
  END_IF;

 END_FUNCTION; -- get_role

 FUNCTION item_in_context(
          item: representation_item;
          cntxt: representation_context
   ): BOOLEAN;

  LOCAL
   y : BAG OF representation_item;
```



```
  END_LOCAL;
  IF SIZEOF(USEDIN(item,'SHIP_STRUCTURES_SCHEMA.REPRESENTATION.ITEMS') *
      cntxt.representations_in_context) > 0 THEN
    RETURN(TRUE);
  ELSE
    y := QUERY ( z <* USEDIN(item,'') | (
       'SHIP_STRUCTURES_SCHEMA.REPRESENTATION_ITEM' IN TYPEOF(z)) );
    IF SIZEOF(y) > 0 THEN
      REPEAT i := 1 TO HIINDEX(y) BY 1;
        IF item_in_context(y[i],cntxt) THEN
          RETURN(TRUE);
        END_IF;
      END_REPEAT;
    END_IF;
  END_IF;
  RETURN(FALSE);

END_FUNCTION; -- item_in_context

FUNCTION leap_year(
            year: year_number
    ): BOOLEAN;
  IF (((year MOD 4) = 0) AND ((year MOD 100) <> 0)) OR ((year MOD 400) =
      0) THEN
    RETURN(TRUE);
  ELSE
    RETURN(FALSE);
  END_IF;

END_FUNCTION; -- leap_year

FUNCTION list_face_loops(
            f: face
    ): LIST [0:?] OF loop;

  LOCAL
    loops : LIST [0:?] OF loop := [];
  END_LOCAL;
  REPEAT i := 1 TO SIZEOF(f.bounds) BY 1;
    loops := loops + f.bounds[i].bound;
  END_REPEAT;
  RETURN(loops);

END_FUNCTION; -- list_face_loops

FUNCTION list_of_topology_reversed(
            a_list: list_of_reversible_topology_item
    ): list_of_reversible_topology_item;

  LOCAL
```

```
    the_reverse : list_of_reversible_topology_item;
  END_LOCAL;
  the_reverse := [];
  REPEAT i := 1 TO SIZEOF(a_list) BY 1;
    the_reverse := topology_reversed(a_list[i]) + the_reverse;
  END_REPEAT;
  RETURN(the_reverse);

END_FUNCTION; -- list_of_topology_reversed

FUNCTION list_to_array(
            lis: LIST [0:?] OF GENERIC:t;
            low: INTEGER;
            u: INTEGER
    ): ARRAY OF GENERIC:t;

  LOCAL
    n   : INTEGER;
    res : ARRAY [low:u] OF GENERIC:t;
  END_LOCAL;
  n := SIZEOF(lis);
  IF n <> ((u - low) + 1) THEN
    RETURN(?);
  ELSE
    res := [lis[1]];
    REPEAT i := 2 TO n BY 1;
      res[(low + i) - 1] := lis[i];
    END_REPEAT;
    RETURN(res);
  END_IF;

END_FUNCTION; -- list_to_array

FUNCTION list_to_set(
            l: LIST [0:?] OF GENERIC:t
    ): SET OF GENERIC:t;

  LOCAL
    s : SET OF GENERIC:t := [];
  END_LOCAL;
  REPEAT i := 1 TO SIZEOF(l) BY 1;
    s := s + l[i];
  END_REPEAT;
  RETURN(s);

END_FUNCTION; -- list_to_set

FUNCTION make_array_of_array(
            lis: LIST [1:?] OF LIST [1:?] OF GENERIC:t;
            low1: INTEGER;
```



```
        u1: INTEGER;
        low2: INTEGER;
        u2: INTEGER
  ): ARRAY OF ARRAY OF GENERIC:t;

  LOCAL
   res : ARRAY [low1:u1] OF ARRAY [low2:u2] OF GENERIC:t;
  END_LOCAL;
  IF ((u1 - low1) + 1) <> SIZEOF(lis) THEN
   RETURN(?);
  END_IF;
  IF ((u2 - low2) + 1) <> SIZEOF(lis[1]) THEN
   RETURN(?);
  END_IF;
  res := [list_to_array(lis[1],low2,u2)];
  REPEAT i := 2 TO HIINDEX(lis) BY 1;
   IF ((u2 - low2) + 1) <> SIZEOF(lis[i]) THEN
    RETURN(?);
   END_IF;
   res[(low1 + i) - 1] := list_to_array(lis[i],low2,u2);
  END_REPEAT;
  RETURN(res);

 END_FUNCTION; -- make_array_of_array

 FUNCTION mixed_loop_type_set(
        l: SET [0:?] OF loop
  ): LOGICAL;

  LOCAL
   poly_loop_type : LOGICAL;
  END_LOCAL;
  IF SIZEOF(l) <= 1 THEN
   RETURN(FALSE);
  END_IF;
  poly_loop_type := 'SHIP_STRUCTURES_SCHEMA.POLY_LOOP' IN TYPEOF(l[1]);
  REPEAT i := 2 TO SIZEOF(l) BY 1;
   IF ('SHIP_STRUCTURES_SCHEMA.POLY_LOOP' IN TYPEOF(l[i])) <>
      poly_loop_type THEN
    RETURN(TRUE);
   END_IF;
  END_REPEAT;
  RETURN(FALSE);

 END_FUNCTION; -- mixed_loop_type_set

 FUNCTION msb_shells(
        brep: manifold_solid_brep
  ): SET [1:?] OF closed_shell;
  IF SIZEOF(QUERY ( msbtype <* TYPEOF(brep) | (msbtype LIKE
```

```
    '*BREP_WITH_VOIDS') )) >= 1 THEN
   RETURN(brep\brep_with_voids.voids + brep.outer);
  ELSE
   RETURN([brep.outer]);
  END_IF;

 END_FUNCTION; -- msb_shells

 FUNCTION nmsf_curve_check(
           cv: representation_item
   ): BOOLEAN;
  IF SIZEOF(['SHIP_STRUCTURES_SCHEMA.BOUNDED_CURVE',
     'SHIP_STRUCTURES_SCHEMA.CONIC',
     'SHIP_STRUCTURES_SCHEMA.CURVE_REPLICA',
     'SHIP_STRUCTURES_SCHEMA.LINE',
     'SHIP_STRUCTURES_SCHEMA.OFFSET_CURVE_3D'] * TYPEOF(cv)) > 1 THEN
   RETURN(FALSE);
  ELSE
   IF (('SHIP_STRUCTURES_SCHEMA.B_SPLINE_CURVE' IN TYPEOF(cv)) AND (cv\
      b_spline_curve.self_intersect = FALSE)) OR (cv\b_spline_curve.
      self_intersect = UNKNOWN) THEN
    RETURN(TRUE);
   ELSE
    IF SIZEOF(['SHIP_STRUCTURES_SCHEMA.CONIC',
       'SHIP_STRUCTURES_SCHEMA.LINE'] * TYPEOF(cv)) = 1 THEN
     RETURN(TRUE);
    ELSE
     IF 'SHIP_STRUCTURES_SCHEMA.CURVE_REPLICA' IN TYPEOF(cv) THEN
      RETURN(nmsf_curve_check(cv\curve_replica.parent_curve));
     ELSE
      IF ('SHIP_STRUCTURES_SCHEMA.OFFSET_CURVE_3D' IN TYPEOF(cv))
         AND ((cv\offset_curve_3d.self_intersect = FALSE) OR (cv\
         offset_curve_3d.self_intersect = UNKNOWN)) AND (NOT (
         'SHIP_STRUCTURES_SCHEMA.POLYLINE' IN TYPEOF(cv\
         offset_curve_3d.basis_curve))) THEN
       RETURN(nmsf_curve_check(cv\offset_curve_3d.basis_curve));
      ELSE
       IF 'SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(cv) THEN
        RETURN(nmsf_curve_check(cv\pcurve.reference_to_curve\
           representation.items[1]) AND nmsf_surface_check(cv\
           pcurve.basis_surface));
       ELSE
        IF 'SHIP_STRUCTURES_SCHEMA.SURFACE_CURVE' IN TYPEOF(cv)
           THEN
         IF nmsf_curve_check(cv\surface_curve.curve_3d) THEN
          REPEAT i := 1 TO SIZEOF(cv\surface_curve.
             associated_geometry) BY 1;
           IF 'SHIP_STRUCTURES_SCHEMA.SURFACE' IN TYPEOF(cv\
              surface_curve.associated_geometry[i]) THEN
            IF NOT nmsf_surface_check(cv\surface_curve.
```



```
                associated_geometry[i]) THEN
              RETURN(FALSE);
             END_IF;
            ELSE
             IF 'SHIP_STRUCTURES_SCHEMA.PCURVE' IN TYPEOF(cv\
               surface_curve.associated_geometry[i]) THEN
              IF NOT nmsf_curve_check(cv\surface_curve.
                associated_geometry[i]) THEN
               RETURN(FALSE);
              END_IF;
             END_IF;
            END_IF;
           END_REPEAT;
           RETURN(TRUE);
          END_IF;
         ELSE
          IF 'SHIP_STRUCTURES_SCHEMA.POLYLINE' IN TYPEOF(cv) THEN
           IF SIZEOF(cv\polyline.points) >= 3 THEN
            RETURN(TRUE);
           END_IF;
          END_IF;
         END_IF;
        END_IF;
       END_IF;
      END_IF;
     END_IF;
    END_IF;
   END_IF;
  RETURN(FALSE);

END_FUNCTION; -- nmsf_curve_check

FUNCTION nmsf_surface_check(
          surf: surface
  ): BOOLEAN;
  IF 'SHIP_STRUCTURES_SCHEMA.ELEMENTARY_SURFACE' IN TYPEOF(surf) THEN
   RETURN(TRUE);
  ELSE
   IF 'SHIP_STRUCTURES_SCHEMA.SWEPT_SURFACE' IN TYPEOF(surf) THEN
    RETURN(nmsf_curve_check(surf\swept_surface.swept_curve));
   ELSE
    IF (('SHIP_STRUCTURES_SCHEMA.OFFSET_SURFACE' IN TYPEOF(surf)) AND
      (surf\offset_surface.self_intersect = FALSE)) OR (surf\
      offset_surface.self_intersect = UNKNOWN) THEN
     RETURN(nmsf_surface_check(surf\offset_surface.basis_surface));
    ELSE
     IF 'SHIP_STRUCTURES_SCHEMA.SURFACE_REPLICA' IN TYPEOF(surf)
        THEN
      RETURN(nmsf_surface_check(surf\surface_replica.parent_surface));
     ELSE
```

```
        IF (('SHIP_STRUCTURES_SCHEMA.B_SPLINE_SURFACE' IN TYPEOF(surf))
           AND (surf\b_spline_surface.self_intersect = FALSE)) OR (
           surf\b_spline_surface.self_intersect = UNKNOWN) THEN
          RETURN(TRUE);
        END_IF;
       END_IF;
     END_IF;
   END_IF;
 END_IF;
 RETURN(FALSE);

END_FUNCTION; -- nmsf_surface_check

FUNCTION normalise(
          arg: vector_or_direction
  ): vector_or_direction;

 LOCAL
  ndim   : INTEGER;
  v      : direction;
  vec    : vector;
  mag    : REAL;
  result : vector_or_direction;
 END_LOCAL;
 IF NOT EXISTS(arg) THEN
  result := ?;
 ELSE
  ndim := arg.dim;
  IF 'SHIP_STRUCTURES_SCHEMA.VECTOR' IN TYPEOF(arg) THEN
   BEGIN
    v := dummy_gri || direction(arg.orientation.direction_ratios);
    IF arg.magnitude = 0 THEN
     RETURN(?);
    ELSE
     vec := dummy_gri || vector(v,1);
    END_IF;
   END;
  ELSE
   v := dummy_gri || direction(arg.direction_ratios);
  END_IF;
  mag := 0;
  REPEAT i := 1 TO ndim BY 1;
   mag := mag + (v.direction_ratios[i] * v.direction_ratios[i]);
  END_REPEAT;
  IF mag > 0 THEN
   mag := SQRT(mag);
   REPEAT i := 1 TO ndim BY 1;
    v.direction_ratios[i] := v.direction_ratios[i] / mag;
   END_REPEAT;
   IF 'SHIP_STRUCTURES_SCHEMA.VECTOR' IN TYPEOF(arg) THEN
```



```
      vec.orientation := v;
      result := vec;
     ELSE
      result := v;
     END_IF;
    ELSE
     RETURN(?);
    END_IF;
  END_IF;
  RETURN(result);

END_FUNCTION; -- normalise

FUNCTION open_shell_reversed(
          a_shell: open_shell
   ): oriented_open_shell;

  LOCAL
   the_reverse : oriented_open_shell;
  END_LOCAL;
  IF 'SHIP_STRUCTURES_SCHEMA.ORIENTED_OPEN_SHELL' IN TYPEOF(a_shell)
     THEN
   the_reverse := dummy_tri || connected_face_set(a_shell\
      connected_face_set.cfs_faces) || open_shell() ||
      oriented_open_shell(a_shell\oriented_open_shell.
      open_shell_element,NOT a_shell\oriented_open_shell.orientation);
  ELSE
   the_reverse := dummy_tri || connected_face_set(a_shell\
      connected_face_set.cfs_faces) || open_shell() ||
      oriented_open_shell(a_shell,FALSE);
  END_IF;
  RETURN(the_reverse);

END_FUNCTION; -- open_shell_reversed

FUNCTION orthogonal_complement(
          vec: direction
   ): direction;

  LOCAL
   result : direction;
  END_LOCAL;
  IF (vec.dim <> 2) OR (NOT EXISTS(vec)) THEN
   RETURN(?);
  ELSE
   result := dummy_gri || direction([-vec.direction_ratios[2],vec.
      direction_ratios[1]]);
   RETURN(result);
  END_IF;
```

```
END_FUNCTION; -- orthogonal_complement

FUNCTION path_head_to_tail(
          a_path: path
  ): BOOLEAN;

 LOCAL
  n : INTEGER;
  p : BOOLEAN := TRUE;
 END_LOCAL;
 n := SIZEOF(a_path.edge_list);
 REPEAT i := 2 TO n BY 1;
  p := p AND (a_path.edge_list[i - 1].edge_end :=: a_path.edge_list[i]
     .edge_start);
 END_REPEAT;
 RETURN(p);

END_FUNCTION; -- path_head_to_tail

FUNCTION path_reversed(
          a_path: path
  ): oriented_path;

 LOCAL
  the_reverse : oriented_path;
 END_LOCAL;
 IF 'SHIP_STRUCTURES_SCHEMA.ORIENTED_PATH' IN TYPEOF(a_path) THEN
  the_reverse := dummy_tri || path(list_of_topology_reversed(a_path.
     edge_list)) || oriented_path(a_path\oriented_path.path_element,
     NOT a_path\oriented_path.orientation);
 ELSE
  the_reverse := dummy_tri || path(list_of_topology_reversed(a_path.
     edge_list)) || oriented_path(a_path,FALSE);
 END_IF;
 RETURN(the_reverse);

END_FUNCTION; -- path_reversed

FUNCTION scalar_times_vector(
          scalar: REAL;
          vec: vector_or_direction
  ): vector;

 LOCAL
  v      : direction;
  mag    : REAL;
  result : vector;
 END_LOCAL;
 IF (NOT EXISTS(scalar)) OR (NOT EXISTS(vec)) THEN
  RETURN(?);
```



```
  ELSE
    IF 'SHIP_STRUCTURES_SCHEMA.VECTOR' IN TYPEOF(vec) THEN
      v := dummy_gri || direction(vec.orientation.direction_ratios);
      mag := scalar * vec.magnitude;
    ELSE
      v := dummy_gri || direction(vec.direction_ratios);
      mag := scalar;
    END_IF;
    IF mag < 0 THEN
      REPEAT i := 1 TO SIZEOF(v.direction_ratios) BY 1;
        v.direction_ratios[i] := -v.direction_ratios[i];
      END_REPEAT;
      mag := -mag;
    END_IF;
    result := dummy_gri || vector(normalise(v),mag);
  END_IF;
  RETURN(result);

END_FUNCTION; -- scalar_times_vector

FUNCTION second_proj_axis(
            z_axis, x_axis, arg: direction
    ): direction;

  LOCAL
    temp   : vector;
    v      : direction;
    y_axis : vector;
  END_LOCAL;
  IF NOT EXISTS(arg) THEN
    v := dummy_gri || direction([0,1,0]);
  ELSE
    v := arg;
  END_IF;
  temp := scalar_times_vector(dot_product(v,z_axis),z_axis);
  y_axis := vector_difference(v,temp);
  temp := scalar_times_vector(dot_product(v,x_axis),x_axis);
  y_axis := vector_difference(y_axis,temp);
  y_axis := normalise(y_axis);
  RETURN(y_axis.orientation);

END_FUNCTION; -- second_proj_axis

FUNCTION set_of_topology_reversed(
            a_set: set_of_reversible_topology_item
    ): set_of_reversible_topology_item;

  LOCAL
    the_reverse : set_of_reversible_topology_item;
  END_LOCAL;
```

```
  the_reverse := [];
  REPEAT i := 1 TO SIZEOF(a_set) BY 1;
    the_reverse := the_reverse + topology_reversed(a_set[i]);
  END_REPEAT;
  RETURN(the_reverse);

END_FUNCTION; -- set_of_topology_reversed

FUNCTION shell_reversed(
          a_shell: shell
    ): shell;
  IF 'SHIP_STRUCTURES_SCHEMA.OPEN_SHELL' IN TYPEOF(a_shell) THEN
    RETURN(open_shell_reversed(a_shell));
  ELSE
    IF 'SHIP_STRUCTURES_SCHEMA.CLOSED_SHELL' IN TYPEOF(a_shell) THEN
      RETURN(closed_shell_reversed(a_shell));
    ELSE
      RETURN(?);
    END_IF;
  END_IF;

END_FUNCTION; -- shell_reversed

FUNCTION surface_weights_positive(
          b: rational_b_spline_surface
    ): BOOLEAN;

  LOCAL
    result : BOOLEAN := TRUE;
  END_LOCAL;
  REPEAT i := 0 TO b.u_upper BY 1;
    REPEAT j := 0 TO b.v_upper BY 1;
      IF b.weights[i][j] <= 0 THEN
        result := FALSE;
        RETURN(result);
      END_IF;
    END_REPEAT;
  END_REPEAT;
  RETURN(result);

END_FUNCTION; -- surface_weights_positive

FUNCTION topology_reversed(
          an_item: reversible_topology
    ): reversible_topology;
  IF 'SHIP_STRUCTURES_SCHEMA.EDGE' IN TYPEOF(an_item) THEN
    RETURN(edge_reversed(an_item));
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.PATH' IN TYPEOF(an_item) THEN
    RETURN(path_reversed(an_item));
```



```
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.FACE_BOUND' IN TYPEOF(an_item) THEN
    RETURN(face_bound_reversed(an_item));
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.FACE' IN TYPEOF(an_item) THEN
    RETURN(face_reversed(an_item));
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.SHELL' IN TYPEOF(an_item) THEN
    RETURN(shell_reversed(an_item));
  END_IF;
  IF 'SET' IN TYPEOF(an_item) THEN
    RETURN(set_of_topology_reversed(an_item));
  END_IF;
  IF 'LIST' IN TYPEOF(an_item) THEN
    RETURN(list_of_topology_reversed(an_item));
  END_IF;
  RETURN(?);

END_FUNCTION; -- topology_reversed

FUNCTION using_items(
            item: founded_item_select;
            checked_items: SET OF founded_item_select
    ): SET OF founded_item_select;

  LOCAL
    next_items     : SET OF founded_item_select;
    new_check_items : SET OF founded_item_select;
    result_items    : SET OF founded_item_select;
  END_LOCAL;
  result_items := [];
  new_check_items := checked_items + item;
  next_items := QUERY ( z <* bag_to_set(USEDIN(item,")) | ((
     'SHIP_STRUCTURES_SCHEMA.REPRESENTATION_ITEM' IN TYPEOF(z)) OR (
     'SHIP_STRUCTURES_SCHEMA.FOUNDED_ITEM' IN TYPEOF(z))) );
  IF SIZEOF(next_items) > 0 THEN
    REPEAT i := 1 TO HIINDEX(next_items) BY 1;
      IF NOT (next_items[i] IN new_check_items) THEN
        result_items := result_items + next_items[i] + using_items(
            next_items[i],new_check_items);
      END_IF;
    END_REPEAT;
  END_IF;
  RETURN(result_items);

END_FUNCTION; -- using_items

FUNCTION using_representations(
            item: founded_item_select
    ): SET OF representation;
```

```
  LOCAL
    results           : SET OF representation;
    intermediate_items : SET OF founded_item_select;
    result_bag        : BAG OF representation;
  END_LOCAL;
  results := [];
  result_bag := USEDIN(item,
     'SHIP_STRUCTURES_SCHEMA.REPRESENTATION.ITEMS');
  IF SIZEOF(result_bag) > 0 THEN
    REPEAT i := 1 TO HIINDEX(result_bag) BY 1;
      results := results + result_bag[i];
    END_REPEAT;
  END_IF;
  intermediate_items := using_items(item,[]);
  IF SIZEOF(intermediate_items) > 0 THEN
    REPEAT i := 1 TO HIINDEX(intermediate_items) BY 1;
      result_bag := USEDIN(intermediate_items[i],
         'SHIP_STRUCTURES_SCHEMA.REPRESENTATION.ITEMS');
      IF SIZEOF(result_bag) > 0 THEN
        REPEAT j := 1 TO HIINDEX(result_bag) BY 1;
          results := results + result_bag[j];
        END_REPEAT;
      END_IF;
    END_REPEAT;
  END_IF;
  RETURN(results);

END_FUNCTION; -- using_representations

FUNCTION valid_calendar_date(
           date: calendar_date
    ): LOGICAL;
  CASE date.month_component OF
    1 :       RETURN((1 <= date.day_component) AND (date.day_component
          <= 31));
    2 :       BEGIN
        IF leap_year(date.year_component) THEN
          RETURN((1 <= date.day_component) AND (date.day_component <= 29));
        ELSE
          RETURN((1 <= date.day_component) AND (date.day_component <= 28));
        END_IF;
      END;
    3 :       RETURN((1 <= date.day_component) AND (date.day_component
          <= 31));
    4 :       RETURN((1 <= date.day_component) AND (date.day_component
          <= 30));
    5 :       RETURN((1 <= date.day_component) AND (date.day_component
          <= 31));
    6 :       RETURN((1 <= date.day_component) AND (date.day_component
          <= 30));
```



```
  7 :       RETURN((1 <= date.day_component) AND (date.day_component
      <= 31));
  8 :       RETURN((1 <= date.day_component) AND (date.day_component
      <= 31));
  9 :       RETURN((1 <= date.day_component) AND (date.day_component
      <= 30));
  10 :      RETURN((1 <= date.day_component) AND (date.
      day_component <= 31));
  11 :      RETURN((1 <= date.day_component) AND (date.
      day_component <= 30));
  12 :      RETURN((1 <= date.day_component) AND (date.
      day_component <= 31));
  END_CASE;
 RETURN(FALSE);

END_FUNCTION; -- valid_calendar_date

FUNCTION valid_geometrically_bounded_wf_curve(
          crv: curve
  ): BOOLEAN;
 IF SIZEOF(['SHIP_STRUCTURES_SCHEMA.POLYLINE',
    'SHIP_STRUCTURES_SCHEMA.B_SPLINE_CURVE',
    'SHIP_STRUCTURES_SCHEMA.ELLIPSE','SHIP_STRUCTURES_SCHEMA.CIRCLE'] *
     TYPEOF(crv)) = 1 THEN
  RETURN(TRUE);
 ELSE
  IF 'SHIP_STRUCTURES_SCHEMA.TRIMMED_CURVE' IN TYPEOF(crv) THEN
   IF SIZEOF(['SHIP_STRUCTURES_SCHEMA.LINE',
      'SHIP_STRUCTURES_SCHEMA.PARABOLA',
      'SHIP_STRUCTURES_SCHEMA.HYPERBOLA'] * TYPEOF(crv\trimmed_curve.
      basis_curve)) = 1 THEN
    RETURN(TRUE);
   ELSE
    RETURN(valid_geometrically_bounded_wf_curve(crv\trimmed_curve.
       basis_curve));
   END_IF;
  ELSE
   IF 'SHIP_STRUCTURES_SCHEMA.OFFSET_CURVE_3D' IN TYPEOF(crv) THEN
    RETURN(valid_geometrically_bounded_wf_curve(crv\offset_curve_3d.
       basis_curve));
   ELSE
    IF 'SHIP_STRUCTURES_SCHEMA.CURVE_REPLICA' IN TYPEOF(crv) THEN
     RETURN(valid_geometrically_bounded_wf_curve(crv\curve_replica.
        parent_curve));
    ELSE
     IF 'SHIP_STRUCTURES_SCHEMA.COMPOSITE_CURVE' IN TYPEOF(crv)
        THEN
      RETURN(SIZEOF(QUERY ( ccs <* crv\composite_curve.segments |
         (NOT valid_geometrically_bounded_wf_curve(ccs.
         parent_curve)) )) = 0);
```

```
        END_IF;
       END_IF;
     END_IF;
   END_IF;
  END_IF;
  RETURN(FALSE);

 END_FUNCTION; -- valid_geometrically_bounded_wf_curve

 FUNCTION valid_geometrically_bounded_wf_point(
           pnt: point
    ): BOOLEAN;
  IF 'SHIP_STRUCTURES_SCHEMA.CARTESIAN_POINT' IN TYPEOF(pnt) THEN
   RETURN(TRUE);
  ELSE
   IF 'SHIP_STRUCTURES_SCHEMA.POINT_ON_CURVE' IN TYPEOF(pnt) THEN
    RETURN(valid_geometrically_bounded_wf_curve(pnt\point_on_curve.
       basis_curve));
   ELSE
    IF 'SHIP_STRUCTURES_SCHEMA.POINT_REPLICA' IN TYPEOF(pnt) THEN
     RETURN(valid_geometrically_bounded_wf_point(pnt\point_replica.
        parent_pt));
    END_IF;
   END_IF;
  END_IF;
  RETURN(FALSE);

 END_FUNCTION; -- valid_geometrically_bounded_wf_point

 FUNCTION valid_measure_value(
           m: measure_value
    ): BOOLEAN;
  IF 'REAL' IN TYPEOF(m) THEN
   RETURN(m > 0);
  ELSE
   IF 'INTEGER' IN TYPEOF(m) THEN
    RETURN(m > 0);
   ELSE
    RETURN(TRUE);
   END_IF;
  END_IF;

 END_FUNCTION; -- valid_measure_value

 FUNCTION valid_time(
           time: local_time
    ): BOOLEAN;
  IF EXISTS(time.second_component) THEN
   RETURN(EXISTS(time.minute_component));
  ELSE
```



```
    RETURN(TRUE);
  END_IF;

END_FUNCTION; -- valid_time

FUNCTION valid_units(
          m: measure_with_unit
  ): BOOLEAN;
  IF 'SHIP_STRUCTURES_SCHEMA.LENGTH_MEASURE' IN TYPEOF(m.value_component)
      THEN
    IF derive_dimensional_exponents(m.unit_component) <>
       dimensional_exponents(1,0,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.MASS_MEASURE' IN TYPEOF(m.value_component)
      THEN
    IF derive_dimensional_exponents(m.unit_component) <>
       dimensional_exponents(0,1,0,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.TIME_MEASURE' IN TYPEOF(m.value_component)
      THEN
    IF derive_dimensional_exponents(m.unit_component) <>
       dimensional_exponents(0,0,1,0,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.ELECTRIC_CURRENT_MEASURE' IN TYPEOF(m.
     value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
       dimensional_exponents(0,0,0,1,0,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.THERMODYNAMIC_TEMPERATURE_MEASURE' IN
     TYPEOF(m.value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
       dimensional_exponents(0,0,0,0,1,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.CELSIUS_TEMPERATURE_MEASURE' IN TYPEOF(m.
     value_component) THEN
    IF derive_dimensional_exponents(m.unit_component) <>
       dimensional_exponents(0,0,0,0,1,0,0) THEN
      RETURN(FALSE);
    END_IF;
  END_IF;
```

```
  IF 'SHIP_STRUCTURES_SCHEMA.AMOUNT_OF_SUBSTANCE_MEASURE' IN TYPEOF(m.
     value_component) THEN
   IF derive_dimensional_exponents(m.unit_component) <>
      dimensional_exponents(0,0,0,0,0,1,0) THEN
    RETURN(FALSE);
   END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.LUMINOUS_INTENSITY_MEASURE' IN TYPEOF(m.
     value_component) THEN
   IF derive_dimensional_exponents(m.unit_component) <>
      dimensional_exponents(0,0,0,0,0,0,1) THEN
    RETURN(FALSE);
   END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.PLANE_ANGLE_MEASURE' IN TYPEOF(m.
     value_component) THEN
   IF derive_dimensional_exponents(m.unit_component) <>
      dimensional_exponents(0,0,0,0,0,0,0) THEN
    RETURN(FALSE);
   END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.SOLID_ANGLE_MEASURE' IN TYPEOF(m.
     value_component) THEN
   IF derive_dimensional_exponents(m.unit_component) <>
      dimensional_exponents(0,0,0,0,0,0,0) THEN
    RETURN(FALSE);
   END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.AREA_MEASURE' IN TYPEOF(m.value_component)
      THEN
   IF derive_dimensional_exponents(m.unit_component) <>
      dimensional_exponents(2,0,0,0,0,0,0) THEN
    RETURN(FALSE);
   END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.VOLUME_MEASURE' IN TYPEOF(m.value_component)
      THEN
   IF derive_dimensional_exponents(m.unit_component) <>
      dimensional_exponents(3,0,0,0,0,0,0) THEN
    RETURN(FALSE);
   END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.RATIO_MEASURE' IN TYPEOF(m.value_component)
      THEN
   IF derive_dimensional_exponents(m.unit_component) <>
      dimensional_exponents(0,0,0,0,0,0,0) THEN
    RETURN(FALSE);
   END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.POSITIVE_LENGTH_MEASURE' IN TYPEOF(m.
```



```
    value_component) THEN
   IF derive_dimensional_exponents(m.unit_component) <>
     dimensional_exponents(1,0,0,0,0,0,0) THEN
    RETURN(FALSE);
   END_IF;
  END_IF;
  IF 'SHIP_STRUCTURES_SCHEMA.POSITIVE_PLANE_ANGLE_MEASURE' IN TYPEOF(m.
    value_component) THEN
   IF derive_dimensional_exponents(m.unit_component) <>
     dimensional_exponents(0,0,0,0,0,0,0) THEN
    RETURN(FALSE);
   END_IF;
  END_IF;
  RETURN(TRUE);

END_FUNCTION; -- valid_units

FUNCTION valid_wireframe_edge_curve(
          crv: curve
   ): BOOLEAN;
  IF SIZEOF(['SHIP_STRUCTURES_SCHEMA.LINE',
    'SHIP_STRUCTURES_SCHEMA.CONIC',
    'SHIP_STRUCTURES_SCHEMA.B_SPLINE_CURVE',
    'SHIP_STRUCTURES_SCHEMA.POLYLINE'] * TYPEOF(crv)) = 1 THEN
   RETURN(TRUE);
  ELSE
   IF 'SHIP_STRUCTURES_SCHEMA.CURVE_REPLICA' IN TYPEOF(crv) THEN
    RETURN(valid_wireframe_edge_curve(crv\curve_replica.parent_curve));
   ELSE
    IF 'SHIP_STRUCTURES_SCHEMA.OFFSET_CURVE_3D' IN TYPEOF(crv) THEN
     RETURN(valid_wireframe_edge_curve(crv\offset_curve_3d.
       basis_curve));
    END_IF;
   END_IF;
  END_IF;
  RETURN(FALSE);

END_FUNCTION; -- valid_wireframe_edge_curve

FUNCTION valid_wireframe_vertex_point(
          pnt: point
   ): BOOLEAN;
  IF 'SHIP_STRUCTURES_SCHEMA.CARTESIAN_POINT' IN TYPEOF(pnt) THEN
   RETURN(TRUE);
  ELSE
   IF 'SHIP_STRUCTURES_SCHEMA.POINT_REPLICA' IN TYPEOF(pnt) THEN
    RETURN(valid_wireframe_vertex_point(pnt\point_replica.parent_pt));
   END_IF;
  END_IF;
  RETURN(FALSE);
```

```
END_FUNCTION; -- valid_wireframe_vertex_point

FUNCTION vector_difference(
          arg1, arg2: vector_or_direction
  ): vector;

 LOCAL
  ndim   : INTEGER;
  mag2   : REAL;
  mag1   : REAL;
  mag    : REAL;
  res    : direction;
  vec1   : direction;
  vec2   : direction;
  result : vector;
 END_LOCAL;
 IF (NOT EXISTS(arg1)) OR (NOT EXISTS(arg2)) OR (arg1.dim <> arg2.dim)
    THEN
  RETURN(?);
 ELSE
  BEGIN
   IF 'SHIP_STRUCTURES_SCHEMA.VECTOR' IN TYPEOF(arg1) THEN
    mag1 := arg1.magnitude;
    vec1 := arg1.orientation;
   ELSE
    mag1 := 1;
    vec1 := arg1;
   END_IF;
   IF 'SHIP_STRUCTURES_SCHEMA.VECTOR' IN TYPEOF(arg2) THEN
    mag2 := arg2.magnitude;
    vec2 := arg2.orientation;
   ELSE
    mag2 := 1;
    vec2 := arg2;
   END_IF;
   vec1 := normalise(vec1);
   vec2 := normalise(vec2);
   ndim := SIZEOF(vec1.direction_ratios);
   mag := 0;
   res := dummy_gri || direction(vec1.direction_ratios);
   REPEAT i := 1 TO ndim BY 1;
    res.direction_ratios[i] := (mag1 * vec1.direction_ratios[i]) + (
       mag2 * vec2.direction_ratios[i]);
    mag := mag + (res.direction_ratios[i] * res.direction_ratios[i]);
   END_REPEAT;
   IF mag > 0 THEN
    result := dummy_gri || vector(res,SQRT(mag));
   ELSE
    result := dummy_gri || vector(vec1,0);
   END_IF;
```



```
    END;
  END_IF;
  RETURN(result);

END_FUNCTION; -- vector_difference

FUNCTION which_class(
          t: GENERIC
  ): LIST OF STRING;

  LOCAL
    class_list : LIST OF STRING := [];
    elements   : BAG OF applied_classification_assignment;
  END_LOCAL;
  elements := USEDIN(t,
     'SHIP_STRUCTURES_SCHEMA.applied_classification_assignment.ITEMS');
  REPEAT i := 1 TO HIINDEX(elements) BY 1;
    IF elements[i]\classification_assignment.role.name =
       'class membership' THEN
      class_list := class_list + elements[i]\classification_assignment.
         assigned_class\group.name;
    END_IF;
  END_REPEAT;
  RETURN(class_list);

END_FUNCTION; -- which_class

END_SCHEMA; -- ship_structures_schema
```

Annex B
(normative)

AIM short names

The following table provides the short names of entities specified in the AIM of this part of ISO 10303. Requirements on the use of short names are found in the implementation methods included in ISO 10303.

Entity name	Short name
ACTION	ACTION
ACTION_METHOD	ACTMTH
ACTION_REQUEST_ASSIGNMENT	ACRQAS
ACTION_REQUEST_SOLUTION	ACRQSL
ADDRESS	ADDRSS
ADVANCED_BREP_SHAPE_REPRESENTATION	ABSR
ADVANCED_FACE	ADVFC
APPLICATION_CONTEXT	APPCNT
APPLICATION_CONTEXT_ELEMENT	APCNEL
APPLICATION_PROTOCOL_DEFINITION	APPRDF
APPLIED_ACTION_REQUEST_ASSIGNMENT	AARA
APPLIED_APPROVAL_ASSIGNMENT	APAPAS
applied_classification_assignment	APCLAS
APPLIED_DATE_AND_TIME_ASSIGNMENT	ADATA
APPLIED_DOCUMENT_REFERENCE	APDCRF
APPLIED_EXTERNAL_IDENTIFICATION_ASSIGNMENT	AEIA
APPLIED_GROUP_ASSIGNMENT	APGRAS
APPLIED_IDENTIFICATION_ASSIGNMENT	APIDAS
APPLIED_ORGANIZATION_ASSIGNMENT	APORAS
APPLIED_PERSON_AND_ORGANIZATION_ASSIGNMENT	APAOA
APPLIED_PERSON_ASSIGNMENT	APPRAS

AIM short names (continued)

Entity name	Short name
APPROVAL	APPRVL
APPROVAL_ASSIGNMENT	APPASS
APPROVAL_DATE_TIME	APDTTM
APPROVAL_PERSON_ORGANIZATION	APPROR
APPROVAL_ROLE	APPRL
APPROVAL_STATUS	APPSTT
AXIS1_PLACEMENT	AX1PLC
AXIS2_PLACEMENT_2D	A2PL2D
AXIS2_PLACEMENT_3D	A2PL3D
B_SPLINE_CURVE	BSPCR
B_SPLINE_CURVE_WITH_KNOTS	BSCWK
B_SPLINE_SURFACE	BSPSR
B_SPLINE_SURFACE_WITH_KNOTS	BSSWK
BEZIER_CURVE	BZRCRV
BEZIER_SURFACE	BZRSRF
BOUNDED_CURVE	BNDCRV
BOUNDED_PCURVE	BNDPCR
BOUNDED_SURFACE	BNDSRF
BOUNDED_SURFACE_CURVE	BNSRCR
BREP_WITH_VOIDS	BRWTVD
CALENDAR_DATE	CLNDT
CARTESIAN_POINT	CRTPNT
CARTESIAN_TRANSFORMATION_OPERATOR	CRTROP
CARTESIAN_TRANSFORMATION_OPERATOR_3D	CTO3
CHARACTERIZED_OBJECT	CHROBJ
CIRCLE	CIRCLE
CLASS	CLASS

AIM short names (continued)

Entity name	Short name
CLASSIFICATION_ASSIGNMENT	CLSASS
CLASSIFICATION_ROLE	CLSRL
CLOSED_SHELL	CLSSHL
COMPOSITE_CURVE	CMPCRV
COMPOSITE_CURVE_ON_SURFACE	CCOS
COMPOSITE_CURVE_SEGMENT	CMCRSG
COMPOUND_REPRESENTATION_ITEM	CMRPIT
CONIC	CONIC
CONICAL_SURFACE	CNCSRF
CONNECTED_EDGE_SET	CNEDST
CONNECTED_FACE_SET	CNFCST
CONTEXT_DEPENDENT_UNIT	CNDPUN
CONVERSION_BASED_UNIT	CNBSUN
COORDINATED_UNIVERSAL_TIME_OFFSET	CUTO
CURVE	CURVE
CURVE_REPLICA	CRVRPL
CYLINDRICAL_SURFACE	CYLSRF
DATE	DATE
DATE_AND_TIME	DTANTM
DATE_AND_TIME_ASSIGNMENT	DATA
DATE_TIME_ROLE	DTTMRL
DEFINITIONAL_REPRESENTATION	DFNRPR
DEGENERATE_PCURVE	DGNPCR
DEGENERATE_TOROIDAL_SURFACE	DGTRSR
DERIVED_UNIT	DRVUNT
DERIVED_UNIT_ELEMENT	DRUNEL
DESCRIPTION_ATTRIBUTE	DSCATT

AIM short names (continued)

Entity name	Short name
DESCRIPTIVE_REPRESENTATION_ITEM	DSRPIT
DIMENSIONAL_EXPONENTS	DMNEXP
DIRECTION	DRCTN
DOCUMENT	DCMNT
DOCUMENT_REFERENCE	DCMRFR
DOCUMENT_REPRESENTATION_TYPE	DCRPTY
DOCUMENT_TYPE	DCMTYP
DOCUMENT_USAGE_CONSTRAINT	DCUSCN
EDGE	EDGE
EDGE_BASED_WIREFRAME_MODEL	EBWM
EDGE_BASED_WIREFRAME_SHAPE_REPRESENTATION	EBWSR
EDGE_CURVE	EDGCRV
EDGE_LOOP	EDGLP
ELEMENTARY_SURFACE	ELMSRF
ELLIPSE	ELLPS
EVALUATED_DEGENERATE_PCURVE	EVDGPC
EXECUTED_ACTION	EXCACT
EXTERNAL_IDENTIFICATION_ASSIGNMENT	EXIDAS
EXTERNAL_SOURCE	EXTSRC
EXTERNAL_SOURCE_RELATIONSHIP	EXSRRL
EXTERNALLY_DEFINED_CLASS	EXD0
EXTERNALLY_DEFINED_GENERAL_PROPERTY	EDGP
EXTERNALLY_DEFINED_ITEM	EXDFIT
FACE	FACE
FACE_BASED_SURFACE_MODEL	FBSM
FACE_BOUND	FCBND
FACE_OUTER_BOUND	FCOTBN

AIM short names (continued)

Entity name	Short name
FACE_SURFACE	FCSRF
FACETED_BREP	FCTBR
FOUNDED_ITEM	FNDITM
FUNCTIONALLY_DEFINED_TRANSFORMATION	FNDFTR
GENERAL_PROPERTY	GNRPRP
GEOMETRIC_CURVE_SET	GMCRST
GEOMETRIC_REPRESENTATION_CONTEXT	GMRPCN
GEOMETRIC_REPRESENTATION_ITEM	GMRPIT
GEOMETRIC_SET	GMTST
GEOMETRICALLY_BOUNDED_WIREFRAME_SHAPE_REPRESENTATION	GBWSR
GLOBAL_UNCERTAINTY_ASSIGNED_CONTEXT	GC
GLOBAL_UNIT_ASSIGNED_CONTEXT	GUAC
GROUP	GROUP
GROUP_ASSIGNMENT	GRPASS
GROUP_RELATIONSHIP	GRPRLT
HYPERBOLA	HYPRBL
ID_ATTRIBUTE	IDATT
IDENTIFICATION_ASSIGNMENT	IDNASS
IDENTIFICATION_ASSIGNMENT_RELATIONSHIP	IDASRL
IDENTIFICATION_ROLE	IDNRL
INTERSECTION_CURVE	INTCRV
ITEM_DEFINED_TRANSFORMATION	ITDFTR
KNOWN_SOURCE	KNWSRC
LENGTH_UNIT	LNGUNT
LIBRARY_APPLIED_VERSION_ASSIGNMENT	LAVA
LIBRARY_CLASS_VERSION_ASSIGNMENT	LCVA
LINE	LINE

AIM short names (continued)

Entity name	Short name
LOCAL_TIME	LCLTM
LOOP	LOOP
MANIFOLD_SOLID_BREP	MNSLBR
MAPPED_ITEM	MPPITM
MEASURE_WITH_UNIT	MSWTUN
NAME_ATTRIBUTE	NMATT
NAMED_UNIT	NMDUNT
NON_MANIFOLD_SURFACE_SHAPE_REPRESENTATION	NMSSR
OBJECT_ROLE	OBJRL
OFFSET_CURVE_3D	OFCR3D
OFFSET_SURFACE	OFFSRF
OPEN_SHELL	OPNSHL
ORDINAL_DATE	ORDDT
ORGANIZATION	ORGNZT
ORGANIZATION_ASSIGNMENT	ORGASS
ORGANIZATION_ROLE	ORGRL
ORGANIZATIONAL_ADDRESS	ORGADD
ORGANIZATIONAL_PROJECT	ORGPRJ
ORIENTED_CLOSED_SHELL	ORCLSH
ORIENTED_EDGE	ORNEDG
ORIENTED_FACE	ORNFC
ORIENTED_OPEN_SHELL	OROPSH
ORIENTED_PATH	ORNPTH
ORIENTED_SURFACE	ORNSRF
PARABOLA	PRBL
PARAMETRIC_REPRESENTATION_CONTEXT	PRRPCN
PATH	PATH

AIM short names (continued)

Entity name	Short name
PCURVE	PCURVE
PERSON	PERSON
PERSON_AND_ORGANIZATION	PRANOR
PERSON_AND_ORGANIZATION_ASSIGNMENT	PAOA
PERSON_AND_ORGANIZATION_ROLE	PAOR
PERSON_ASSIGNMENT	PRSAAAAA ASS
PERSON_ROLE	PRSRL
PERSONAL_ADDRESS	PRSADD
PLACEMENT	PLCMNT
PLANE	PLANE
PLANE_ANGLE_UNIT	PLANUN
POINT	POINT
POINT_ON_CURVE	PNONCR
POINT_ON_SURFACE	PNONSR
POINT_REPLICA	PNTRPL
POLY_LOOP	PLYLP
POLYLINE	PLYLN
PRE_DEFINED_ITEM	PRDFIT
PRODUCT	PRDCT
PRODUCT_CATEGORY	PRDCTG
PRODUCT_CATEGORY_RELATIONSHIP	PRCTRL
PRODUCT_CONTEXT	PRDCNT
PRODUCT_DEFINITION	PRDDFN
PRODUCT_DEFINITION_CONTEXT	PRDFCN
PRODUCT_DEFINITION_FORMATION	PRDFFR
PRODUCT_DEFINITION_RELATIONSHIP	PRDFRL
PRODUCT_DEFINITION_SHAPE	PRDFSH

AIM short names (continued)

Entity name	Short name
PRODUCT_DEFINITION_WITH_ASSOCIATED_DOCUMENTS	PDWAD
PRODUCT_RELATED_PRODUCT_CATEGORY	PRPC
PROPERTY_DEFINITION	PRPDFN
PROPERTY_DEFINITION_RELATIONSHIP	PRDFR
PROPERTY_DEFINITION_REPRESENTATION	PRDFRP
QUASI_UNIFORM_CURVE	QSUNCR
QUASI_UNIFORM_SURFACE	QSUNSR
RATIONAL_B_SPLINE_CURVE	RBSC
RATIONAL_B_SPLINE_SURFACE	RBSS
REPARAMETRISED_COMPOSITE_CURVE_SEGMENT	RCCS
REPRESENTATION	RPRSNT
REPRESENTATION_CONTEXT	RPRCNT
REPRESENTATION_ITEM	RPRITM
REPRESENTATION_MAP	RPRMP
REPRESENTATION_RELATIONSHIP	RPRRLT
ROLE_ASSOCIATION	RLASS
SEAM_CURVE	SMCRV
SHAPE_ASPECT	SHPASP
SHAPE_ASPECT_RELATIONSHIP	SHASRL
SHAPE_DEFINITION_REPRESENTATION	SHDFRP
SHAPE_REPRESENTATION	SHPRPR
SI_UNIT	SUNT
SOLID_MODEL	SLDMDL
SPHERICAL_SURFACE	SPHSRF
SURFACE	SRFC
SURFACE_CURVE	SRFCRV
SURFACE_OF_LINEAR_EXTRUSION	SL

AIM short names (concluded)

Entity name	Short name
SURFACE_OF_REVOLUTION	SROFRV
SURFACE_REPLICA	SRFRPL
SWEPT_SURFACE	SWPSRF
THERMODYNAMIC_TEMPERATURE_UNIT	THTMUN
TIME_UNIT	TMUNT
TOPOLOGICAL_REPRESENTATION_ITEM	TPRPIT
TOROIDAL_SURFACE	TRDSRF
TRIMMED_CURVE	TRMCRV
UNCERTAINTY_MEASURE_WITH_UNIT	UMWU
UNIFORM_CURVE	UNFCRV
UNIFORM_SURFACE	UNFSRF
VALUE_REPRESENTATION_ITEM	VLRPIT
VECTOR	VECTOR
VERSIONED_ACTION_REQUEST	VRACRQ
VERTEX	VERTEX
VERTEX_LOOP	VRTLP
VERTEX_POINT	VRTPNT
WEEK_OF_YEAR_AND_DAY_DATE	WOYADD

Annex C
(normative)

Implementation method specific requirements

The implementation method defines what types of exchange behaviour are required with respect to this part of ISO 10303. Conformance to this part of ISO 10303 is realized in an exchange structure. The file format is encoded according to the syntax and EXPRESS language mapping defined in ISO 10303-21 and in the AIM defined in annex A of this part of ISO 10303. The header of the exchange structure identify use of this part of ISO 10303 by the schema name 'ship structures schema'.

Annex D
(normative)

Protocol Implementation Conformance Statement (PICS) proforma

This clause lists the optional elements of this part of ISO 10303. An implementation may choose to support any combination of these optional elements. However, certain combinations of options are likely to be implemented together. These combinations are called conformance classes and are described in the subclauses of this annex.

This annex is in the form of a questionnaire. This questionnaire is intended to be filled out by the implementer and may be used in preparation for conformance testing by a testing laboratory. The completed PICS proforma is referred to as a PICS.

A number of options are identified in this standard for possible use by conforming implementations. Some of these options may be dynamically (run-time) selected for use/non-use, for instance, OPTIONAL attributes of an entity. Others is statically (configuration-time) selected for use/non-use, such as a particular style of geometry as defined in a conformance class.

Questions:

1. For simplicity of reference, an identifier for the product or system with which the tested STEP implementation is packaged in or procured by is required.

Product/system identifier (or name): _____

2. There are nine classes defined in this international standard. Each class specifies a subset of ISO 10303-218 AIM constructs. These classes are detailed in 6 of this document. Conformance to this part of ISO 10303 requires conformance to at least one of the conformance classes 1 through 9.

Claimed classes of conformance (functionality) - circle choices:

Class 1: Support for exchange of ship structures definition and approval data that is created at the preliminary design stage of a ship, has structural definitions, edge based shape representations, and no hull applicability of this stage;

Class 2: Support for exchange of ship structures definition and approval data that is created at the preliminary design stage of a ship, has structural definitions, non-manifold shape representations, and hull applicability of this stage;

Class 3: Support for exchange of ship structures definition and approval data that is elaborated at the detailed design stage of a ship, under consideration of the production design of ship structures;

Class 4: Support for exchange of ship structures definition, approval data, non manifold shape representation and hull applicability that is elaborated at the detailed design stage of a ship, under consideration of the production design of ship structures;

Class 5: Support for exchange of ship structures definition, approval data, edge based wireframe shape representation, and hull applicability that is completed at the product manufacturing stage of a ship, has manufacturing and welding definitions;

Class 6: Support for exchange of ship structures definition, approval data, non manifold shape representation, and hull applicability that is completed at the product manufacturing stage of a ship, has manufacturing and welding definitions;

Class 7: Support for exchange of ship structures definition, approval data, geometrically bounded wireframe shape representation, and no hull applicability that is completed at the product manufacturing stage of a ship, has manufacturing and welding definitions;

Class 8: Support for exchange of ship structures definition, approval data, advanced B-rep shape representation, and hull applicability that is completed at the product manufacturing stage of a ship, has manufacturing and welding definitions;

Class 9: Support for exchange of class approval data, non manifold shape representation, and hull applicability for the structural parts (plates and profiles) of the ship, has class approvals with regard to the manufacturing definitions.

3. Conformance to this international standard may be realized in one or more of several different implementation methods. The implementation methods define what types of exchange behavior are required with respect to this international standard.

Claimed implementation forms - circle choices:

exchange structure (ISO 10303-21);

exchange structure (ISO 10303-22);

exchange structure (ISO 10303-28).

4. If the exchange structure used is ISO 10303-28, which one?

ISO 10303-28 exchange structure? : _____

5. If the implementation receives data, which does not comply with the requirements in this international standard for the selected conformance class(es), or with the requirements of the 20's series of Parts for the selected implementation method, it executes a default response. A default response is statically set.

Default Response: _____

6. A conforming implementation maintains the static options selected throughout subsequent dynamic assessment (testing) without requiring modification. In a user environment, a conforming implementation permanently maintains the provision of selected static options, or it provides users discretionary control over the changing and setting of the static options, or both (depending on the option).

Does the IUT provide some user discretion over the changing and setting of static options?
Yes or No

7. If yes, which ones?

(a)Conformance class(es): _____

(b) Default Response: _____

8. A statement of conformance includes identification of at least one party deeming conformance for the implementation.

Evaluator(s) (tester/certifier/accrediter):_____

Annex E
(normative)

Information object registration

E.1 Document identification

To provide for unambiguous identification of an information object in an open system, the object identifier

{iso standard 10303 part(218) version(1)}

is assigned to this part of ISO 10303. The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

E.2 Schema identification

To provide for unambiguous identification of the ship_structures_schema in an open system, the object identifier

{iso standard 10303 part(218) version(1) schema(1) ship_structures_schema(1)}

is assigned to ship_structures_schema expanded schema (see annex A).

The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Annex F
(informative)

Application activity model

The application activity model (AAM) is provided as an aid to understanding the scope and information requirements defined in this application protocol. The model is presented as a set of figures that contain the activity diagrams and a set of definitions of the activities and their data. Activities and data flows that are out of scope are marked with an asterisk.

The AAM covers activities which go beyond the subject of this application protocol. The diagrams use a modified IDEF0 notation [1]. Figure F.1 gives the basic notation. Each activity may be decomposed to provide more detail. If an activity has been decomposed, a separate figure is included.

As with any IDEF0 model, the AAM is dependent on a particular viewpoint and purpose. The viewpoint of this AAM is that of design information in the form it would be provided to a manufacturing engineer, and its purpose is to clarify the context and scope of this application protocol.

This is an AAM of life cycle activities across all shipbuilding. There are several activity diagrams whose activities are all out of scope but they are important in illustrating the process of developing and manufacturing a part, and how the in-scope requirements were derived.

NOTE The viewpoint of the application activity model is of the global ship development process. This activity model identifies the life cycle activities across all shipbuilding APs. Activities related to the shipbuilding lifecycle that are not expanded in this activity model are detailed in other shipbuilding application protocols.

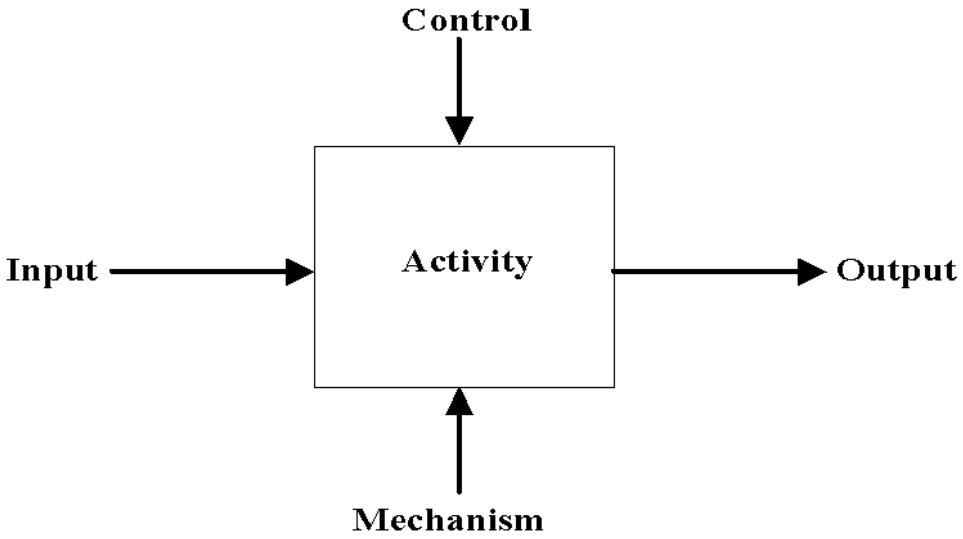


Figure F.1 — IDEF0 Basic notation

F.1 Application activity model definitions and abbreviations

The following terms are used in the application activity model. Terms marked with an asterisk are outside the scope of this application protocol.

The definitions given in this annex do not supersede the definitions given in the main body of the text.

F.1.1 alarm: the document or report resulting by operating a ship that warns the maintenance personnel which problem the ship has.

F.1.2 Approve design of ship structure (A242): the top level activity for the approbation of ship design.

NOTE It is used as the entry activity for both design approval preview, approval of primary design (primary structure) and approval of detail design (secondary structure). This activity is needed not just for certifying the ship; in addition this activity is needed to perform construction surveys and testing before certificates are issued.

F.1.3 approved design: the release state of the design for production and inspection.

F.1.4 approved primary design: the approved fundamental and principal structural design.

NOTE It is a result from the approve design of ship structure activity by a classification society.

F.1.5 arrangements: the ship compartments and spaces.

NOTE Any description of arrangements will include associated definitions of purpose for the compartment or space.

F.1.6 Assemble ship (A33): the production stage of the ship that assembles the modular units and additional parts.

NOTE The result is an assembled ship that still has to be tested.

F.1.7 assembly definition: the rough division into assemblies.

F.1.8 authorities: the governmental agency or corporation to administer a statutory and class survey.

F.1.9 availability, reliability, and maintainability information: the information required for installation and planned maintenance for all components on the ship.

F.1.10 basic hull parameters*: principal dimensions and proportions based on estimations from historical data or from preliminary design development.

F.1.11 budget*: the monetary constraint on the design, construction, and maintenance of the ship.

F.1.12 building capacities of yard: the production ability available to the yard, usually in tones per year.

F.1.13 building sequences: the sequences of producing a ship, as input for preparing the section plans.

F.1.14 Calculate cost of ship* (A124): the description of creating the negotiating documents based on technical product data and the calculation of their estimated manufacturing cost.

NOTE The results of this activity may contain sale price documents, financing support plan, and documents describing funding and possible loans.

F.1.15 capacities for lifting, drydock spaces and production lines: the information that results from estimating drydock resources, which includes scheduling and assignment of a dry dock.

F.1.16 certificates*: documents, licenses, and permits issued for the completed ship.

F.1.17 Check bottom structures (A24241): checking the hull structure within the area of the lowest part of the hull, including tanks, ballast area, and keel to confirm conformance with the regulations.

F.1.18 Check bulkheads (A24244): checking the dimensioning, the correct type, and the correct position of the bulkheads.

F.1.19 Check deck plans (A24243): controlling whether the necessary spaces and volumes of the vessel are correct in volume, size, and position.

F.1.20 Check design of general arrangement* (A2422): the arrangement checking.

NOTE It is basically a checking of watertight integrity arrangements and stability conditions (intact and damage stability). The requirements are considered to meet the relevant regulations of the international Convention on load lines. 1969. Attention should, however, be given to possible additional requirements of the Maritime Authorities in the country in which the ship is to be registered. For passenger ships the "International Convention for Safety Of Life At Sea (SOLAS)" will normally apply.

F.1.21 Check design of girder strength* (A2423): checking the compliance with rule requirements with respect to global strength requirements.

NOTE This approval is necessary before the yard can start ordering steel.

F.1.22 Check global strength and secondary structure (A2424): checking the strength of the whole ship for the approval of the secondary design as part of the approval and certification process .

NOTE The content of this activity is the same for all ships when it comes to conformance with main class rules, but varies when it comes to additional class rules (type of vessel) and register notations.

F.1.23 Check side structures (A24242): checking the sufficient material thickness and the fulfillment of the regulation requirements.

F.1.24 Check WCoG (A2432): checking the weight and center of gravity for the design.

F.1.25 classification society: an organization that enhances the safety of life and property at sea by providing rules, regulations, and personnel for assessing and classifying ships during their lifecycles.

F.1.26 Collect and analyze design transfer into drawings* (A2441): the necessary activities to ensure the material supply for the production of the ship.

F.1.27 Collect part order information (A24344): collecting part ordering information to be used in management information systems (MIS) and estimations.

F.1.28 collision control results: the results from managing the control for clashes.

NOTE The clashes are between different design parts, such as pipes, equipment items, and cable trays.

F.1.29 Complete and approve design of machinery* (A25): the selection, arrangement, and approval of all machinery for the ship.

F.1.30 Complete and approve design of outfitting and distribution systems* (A26): the selection and approval of the necessary outfitting equipment .

NOTE The selection is based mainly on former designs and in accordance with the ship specified requirements. It also contains the layout of the different types of distribution systems such as piping, electrical, and HVAC.

F.1.31 Complete and approve design of ship structure (A24): certify the final structural design of the ship before construction begins.

F.1.32 Complete and approve ship design (A2): the production, validation, and certification of ship design product data, required rules and regulations, documents, and the classification drawings using the preliminary design..

NOTE The result of this activity is the approved design and the production and delivery schedule.

F.1.33 Conduct acceptance trials* (A344): the final trials for delivery of the ship.

F.1.34 Conduct contractor sea trials* (A343): the sea trials to test if the built ship meets the contract requirements.

F.1.35 configuration plan for outfitting: one part of the production documents for steel structures.

F.1.36 consultants: organizations and agencies that provide specific services to shipyards, ship owners, and classification societies during the ship lifecycle.

F.1.37 contract*: the legal agreement that documents and authorizes the requisition for the ship.

NOTE The contract is used as a constraint in subsequent activities such as final design and approval and production.

F.1.38 Coordinate subcontractors tasks* (A24434): the distribution and coordination of the tasks which are done by the subcontractors.

F.1.39 cost*: the calculated monetary outlay for the ship based on the expenditure for material, labour, and overhead.

F.1.40 Create design of classification items (A2412): creating the design of those items required by the classification society.

EXAMPLE Midship sections, bulkheads, shell profile arrangement, and foundations.

F.1.41 Create preliminary design (A122): the early design stage for a ship taking into consideration the classification rules, national and international demands, shipyard constraints, and owner requirements.

F.1.42 Create preliminary general arrangements* (A1222): produce the early compartmentation plans from the preliminary hull form definition.

F.1.43 Create preliminary hull form* (A1221): the early hull form design stage for a ship.

NOTE The first step of designing a ship by using parent ships main dimensions and form parameters. One or more preliminary hull forms will be generated.

F.1.44 Create preliminary machinery design* (A1225): the early selection and arrangement of main machinery for the ship.

NOTE It includes the prime propulsion system, shaft system, power systems, and cargo handling equipment

F.1.45 Create preliminary outfitting design* (A1226): produce the early design for the ship outfitting equipment.

NOTE It includes distributed systems, such as piping and electrical systems.

F.1.46 Create preliminary structure design (A1224): produce the early design of the steel structure and the arrangement of the primary structural members.

F.1.47 Create production documents (A2442): produce the production documents which are needed to describe the features of technical parts and their relationship.

F.1.48 crew: the persons who have duties on a ship.

F.1.49 critical design areas: the design areas at which a change may be expected when performing a design approval preview.

F.1.50 cut-out definition: the definition of the cutouts when defining the notches, lag, and welding of the plates.

F.1.51 Decide post-sales and maintenance support* (A123): develop a proposal for the maintenance of the ship.

NOTE This is part of the bid package documentation and includes the post sales support.

F.1.52 Decommission and disassemble* (A5): the phase of the ship life cycle during which the ship is taken out of service or dismantled.

F.1.53 Define brackets (A243413): add brackets to early and secondary designs.

F.1.54 Define endcuts, beveling, and welding (A243433): defining the design features required for preparing structural parts for the welding of profiles.

F.1.55 Define holes and notches (A243432): fixing the dependent holes and notches in their position when defining the profiles in their length and position.

F.1.56 Define holes and penetrations (A243422): defining the holes and penetrations according to the other functions of the system that has the connected throughputs.

F.1.57 Define insulation and surface treatment (A24345): defining the insulation and surface treatment definitions to be used dependent on the location of structure and functional requirements.

F.1.58 Define notches, lag, and welding (A243423): all design activities which are meant to support the welding process.

F.1.59 Define plates (A243411): setting the layout of individual plates according to the structural hull design.

F.1.60 Define profiles (A243412): setting the position of the profiles according to the structural hull and related design.

F.1.61 Define seams, beveling, and welding (A243421): defining the designed features of structured parts for the welding of plates.

F.1.62 Define Secondary Profiles (A243434): selecting the detailed length of the individual profiles and their size and cross-section.

F.1.63 design change request: the feedback from approval design of ship structure activity.

NOTE It is the requested change of the primary structural design.

F.1.64 Design collision bulkheads (A241212): designing the collision bulkheads, specifically concerning the water resistance and strength in the bow area.

F.1.65 Design longitudinal, horizontal structures (A24122): supporting the design of decks, bottoms, and horizontal shell profiles.

F.1.66 Design longitudinal, vertical structures (A24123): supporting the design of vertical structural separations such as bulkheads, cofferdams, and vertical shell profiles.

F.1.67 Design of transverse structures (A24121): dimensioning the midship section, including plates, shell profiles, and shell thickness.

F.1.68 design schedule*: data that controls the time span from the design phase to production.

F.1.69 Design secondary structure (A24341): designing structure details not defined in early design.

F.1.70 Design ship structure (A241): creating a synthesis of the product specification and transforming it into the physical representation of the product.

NOTE In this case the function is understood to contain mainly the activities to satisfy the regulations set up by the classification society.

F.1.71 Design special areas (A24125): design areas of the ship not covered by the longitudinal and vertical structure .

EXAMPLE The engine foundations as well as aft and fore body

F.1.72 Design transverse bulkheads (A241211): finishing the design by completing the dimensioning of the transverse structure.

F.1.73 Design transverse frames (A241213): design frames between transverse bulkheads.

NOTE The frames are the transverse members that make up the riblike part of the skeleton of a ship.

F.1.74 detail design: the detailed structural design with extended treatment of the primary structural design and attention to particular items, such as the structural parts.

NOTE It is a result from the activity “elaborate ship structure design (see F.1.77)”.

F.1.75 drawing: the technical representations outlining figures or plans of the ship structures.

NOTE The drawing is the result of detail design and input for creating the production documents.

F.1.76 early design: the design of the ship structures including the primary design and approved primary design.

F.1.77 elaborate ship structure design (A243): the function giving detailed information to order parts and to manufacture, and giving the physical outcome of the design.

NOTE Among these activities are the subdividing of physical structures and the determination of tasks for manufacturing.

F.1.78 engineering consultants: organizations that provide specific engineering services to shipyards, ship owners, and classification societies during the ship lifecycle.

F.1.79 equipment certificates*: the certificates issued by the Classification Society on completing the equipment items which will be assembled to create the final product.

F.1.80 Estimate drydock resources and its divisions* (A24431): the function that engages with the time scheduling and assignment of dry dock capacities.

F.1.81 Estimate hydrodynamics and powering* (A1223): calculations approximating hydrodynamic properties data.

EXAMPLE Resistance, propulsion, seakeeping, and maneuverability for the preliminary hull form.

F.1.82 Evaluate request and schedule bid* (A121): shipyard function to analyze and consider the inquiry from the ship owner for a new ship.

F.1.83 feedback: the return of information from an activity.

F.1.84 Finalize and approve general arrangements* (A21): the final design of space allocation and arrangement of compartments and accesses.

F.1.85 Finalize and approve hull form* (A22): achieve an authorized complete hull form with the aid of preliminary design.

NOTE The result is an approved hull form design.

F.1.86 Finalize and approve hydrodynamics and powering* (A23): complete the hydrodynamic analysis for the final design.

NOTE Includes all relevant hydrodynamic calculations such as resistance, propulsion, seakeeping, and manoeuvrability.

F.1.87 Handle plates (A24342): splitting plate parts and add production information.

F.1.88 Handle profiles (A24343): partition profiles, add production information, and design additional and secondary profiles.

F.1.89 historical data from previous designs: data held by the shipyard or model basin on previous ship designs.

NOTE This data is used to estimate the hydrodynamics, powering requirements, and seakeeping.

F.1.90 hole definition: the geometrical and topological information of a hole when defining holes and penetrations on a plate.

F.1.91 hull design: the design of hull form, results of the hull detail design.

F.1.92 hull form sections: the design of the hull moulded form at planar sections taken along the longitudinal axis of the ship.

F.1.93 hull moulded form: the surface that defines the shape of the ship hull.

NOTE Includes the aft-, mid-, and fore-body hull shape definitions, but does not take into account the thickness of the material from which the hull is made.

F.1.94 hydrodynamical loads: the loads caused by hydrodynamical forces.

F.1.95 hydrodynamics and powering results*: engineering design calculations and model basin test data.

NOTE Containing resistance, propulsion, propeller performance, brake power, service speed, seakeeping and manoeuvrability data.

F.1.96 inspection information: the information resulting from visual, non-destructive examinations or tests carried out to assess the condition of one or more components of a ship system.

F.1.97 knowledge and experience: a company's previous history of performance, comprehension of ship information, understanding problem areas, and analysis ability related to a ship throughout its life cycle.

F.1.98 laws, rules, and regulations: national laws, statutory regulations, and classification society rules that are used to control the design, manufacture, operation, maintenance, and scrapping of the ship.

F.1.99 Layout primary structure (A24124): the entry activity for design approval preview and the approval of the primary design.

F.1.100 list of classification items: the list of those items required by the classification society.

EXAMPLE Midship sections, bulkheads, shell profile arrangement, and foundations.

F.1.101 list of required certificates*: certificates for the ship issued by the Classification Society, national, and international regulatory bodies.

F.1.102 loading and stability manual: a booklet placed on board the ship specifying the prescribed limits for loading and unloading the ship as related to strength and stability of the ship.

F.1.103 loads: a mass, force or weight supported by a ship.

F.1.104 longitudinal bulkheads walls: the bulkhead walls between transverse frames.

F.1.105 machinery design: design drawings, prototypes, and electronic models of the ship mechanical systems.

NOTE An output from the final design process.

F.1.106 main cross section, longitudinal profile dimension and plate dimensions: the dimensions of the main hull cross section and longitudinal profiles and plates resulting from the transverse structure design.

F.1.107 Maintain a ship (A42): the maintenance activities during the operation phase of a ship

NOTE It is based on the existence of a planned maintenance system (PMS) for preventive maintenance on board, at the harbor base, or at a yard. Preventative maintenance includes planned regular inspections, lubricating, cleaning, and replacement of parts and may be scheduled by calendar date and/or usage condition. A PMS contains a schedule, procedures, and a listing of required materials, skill, tools, and test equipment. Corrective maintenance (repair) will be applied in case of a failure of a system or component. It may be predictable but is obviously not scheduled. Generally after discovering the failure, a failure analysis and a specific corrective maintenance plan which includes a repair procedure will be implemented. Execution of the maintenance is followed by inspection, testing, reinspection, and approval by an organization authorized for the specific maintenance procedures.

F.1.108 maintenance history: the diary of the maintenance performed on the ship.

F.1.109 maintenance personnel: the persons who have maintenance duties on a ship.

F.1.110 maintenance report: the report on the maintenance results.

F.1.111 maintenance test results: the results of functional tests carried out after the execution of maintenance actions.

F.1.112 Make hull detail design (A2434): develop detail design of the hull steel structure by the use of plates and stiffening elements.

F.1.113 Make initial assembly definition (A2431): the first rough division into assemblies .

NOTE This is further developed during production design.

F.1.114 Make outfitting detail design* (A2435): develop further the outfitting design made at early stages, and design all systems and functionalities not done in early design.

F.1.115 Make production design of ship structure (A244): the function that contains all activities which perform the material supply, the creation of production information and its documentation, and the refined working information.

F.1.116 Manage collision control* (A2433): checking the design for clashes.

NOTE The clashes are between different design parts, such as pipes, equipment items, and cable trays.

F.1.117 manufacturing restrictions: constraints of the ship construction, fabrication, and design processes governed by available technology and shipyard facilities.

F.1.118 material list: the list of raw materials needed to manufacture the ship.

NOTE A result of the final design process.

F.1.119 modification: a feedback from the modification for classification when designing the ship structure.

F.1.120 modifications from machinery*: revisions or adjustments to the hydrodynamics and powering due to feedback from the preliminary machinery design.

F.1.121 modifications to hull form*: revisions or adjustments to the hull shape due to feedback from hydrodynamics and powering results and the final design process.

F.1.122 Modify for classification* (A2413): the function concerned with the items of classification found to be not solved properly according to the rules.

NOTE The inputs are the design change requests which are modified to be satisfactory for the technical solution.

F.1.123 modular units*: sub-sections of the ship complete with machinery and outfitting which will be assembled to create the final product.

F.1.124 offer*: a proposal from the shipyard to produce the requested ship.

NOTE This is shipyard's data that results from the preliminary design process.

F.1.125 offer guidelines*: the data necessary to make an unconditional offer to the ship owner.

F.1.126 Operate a ship* (A41): provides feedback into other activities based on running the ship.

NOTE The main aspect of ship operation in this model is with respect to maintenance. This maintenance process is controlled by "planned maintenance system" which is performed during the "create ship design" activity, and further more controlled by classification rules.

F.1.127 Operate and maintain a ship (A4): procedures and documentation for running, servicing, and repairing the ship during its useful lifetime.

F.1.128 operational histories*: the diary made during the operation of a ship.

F.1.129 operational information*: the document resulting from operating, maintaining, and surveying a ship to give information about a ship condition.

NOTE Accumulated information recorded during the functional and serviceable phase of the ship used for maintenance and in the final scrapping stage.

F.1.130 operation manuals: the booklets which are placed on board for the information, which enables the crew to operate the ship.

F.1.131 owner: the organization that requests, orders, takes delivery of, and operates the ship.

F.1.132 owner request with requirements: the requirements document submitted to the shipyard by the owner with the invitation to submit a bid.

F.1.133 part ordering information: the information to order the parts.

NOTE It is obtained from the hull detail design. After that the plates and profiles have been handled and the parts are collected.

F.1.134 Partition profiles (A243431): defining the resulting length of plate related stiffeners

F.1.135 Perform design approval preview (A2421): This is the top level activity for the approval preview of ship design. This activity is a feasibility study conducted by a classification society, in which the design is checked very roughly to detect critical areas for thorough investigation and conformity checking, both as a design comment and to draw attention to specific areas during design approval. The content of this activity may vary with contract specifications and type of ship.

F.1.136 Perform ship life cycle (A0): the completed stages of a ship existence that include conception, design, construction, operation, and disposal.

F.1.137 Place order* (A13): to acknowledge and accept a bid proposal from a shipyard to produce a ship.

NOTE From this a contract is awarded to the shipyard.

F.1.138 planned maintenance system: data created during the final design process and used during the operation and maintenance of the ship.

F.1.139 plate info: results from defining the plates during the secondary structure design state.

NOTE It includes material type, thickness, and geometrical form of the plates.

F.1.140 pre layout: the first layout of the ship that is produced during the bid evaluation stage and is the basis for the preliminary design.

F.1.141 preliminary design: the design that leads to the submission of a bid proposal.

F.1.142 preliminary general arrangements*: the early definition of spaces and compartments for the ship.

NOTE General arrangements are approximated or calulated as a result of the preliminary design process.

F.1.143 preliminary hull form: the early definition of the hull moulded form, as a result of the preliminary design process.

NOTE This is used in the bid proposal documents, for preliminary compartment design, hydrodynamics, and powering calculations.

F.1.144 preliminary machinery design*: the early definition of the ship mechanical systems.

NOTE This is used early to estimate the noise, speed and vibration, and to estimate the machinery weights.

F.1.145 preliminary machinery, structure, and outfitting design: the early definition of the ship machinery, structure, and outfitting and furnishing.

NOTE This assists in the preliminary general arrangements development.

F.1.146 preliminary outfitting design*: the early definition of the ship outfitting and accommodation.

NOTE This contributes to part of the preliminary design process.

F.1.147 preliminary structure design: the early definition of the steel structure for the ship.

NOTE This contributes to part of the preliminary design process.

F.1.148 Prepare bid (A12): the preparation of all documents and data necessary for submitting a proposal to the owner for building the ship.

F.1.149 Prepare outfitting and equipping plans and schedules* (A24433): work out outfitting and equipping plans.

F.1.150 Prepare production documents for steel structure (A24421): the function that performs the segmentation of the steel structure, creating working and assembly information.

F.1.151 Prepare section plans (A24432): the function that coordinates the proper sequence of the steel structure and outfitting of the structure.

F.1.152 Present offer* (A125): submit all required bid proposal documents for producing the requested ship to the prospective owner.

F.1.153 primary design: the fundamental and principal structural design, a result from the design ship structure activity.

F.1.154 primary design in special areas: the fundamental and principal design of the ship not covered by the longitudinal and vertical structure.

F.1.155 primary design of similar ships: the fundamental structural data of ships in the same category.

F.1.156 primary structure: the first result during the design of the classification items with design approval preview.

F.1.157 Produce and approve reference documents* (A35): compose and assemble technical documentation authorized for the ship using production information.

NOTE The output includes the loading and stability manual that is approved by the Classification Society.

F.1.158 Produce and inspect a ship (A3): build and perform quality assurance at each stage of construction or fabrication of the ship.

NOTE Inspect means the controlling of all activities throughout the entire production life cycle of a ship.

F.1.159 Produce modular build units (A32): fabricate steel subsections and erect into large detached certified units that when assembled or attached will make up the completed ship.

NOTE Their production is controlled by the schedule, contract, the approved design, any manufacturing restrictions, and the classification society. The results are the modular units, which are assembled into the ship.

F.1.160 Produce steel sub-sections (A31): construct and join structural steel members that when assembled and certified make up a modular unit.

NOTE Their production is controlled by the schedule, contract, the approved design, any manufacturing restrictions, and the classification society. The results are the steel sub-sections, which are assembled into modular units for completing the ship.

F.1.161 product component information: the technical data for all parts and items that will be incorporated into the ship.

NOTE These are taken into consideration when the preliminary designs are being made.

F.1.162 production and delivery schedule*: the timetable to which the ship is to be manufactured and delivered.

F.1.163 production information: data describing the manufacturing and construction details of the product.

EXAMPLE Dimensions, mechanical properties, materials, workshop information.

F.1.164 production planner: the person making and carrying out production plans.

F.1.165 production schedule*: data that controls the time from the production phase to delivery.

F.1.166 propeller design*: the design of the propeller or propulsor as a result of the hydrodynamics and powering calculations.

NOTE The design controls some of the ship machinery design.

F.1.167 quality assurance*: the rules applied by an organization within the shipyard that has the task to audit the shipyard organization and applied processes in a manner such that the quality of the resulting product is assured.

F.1.168 Refine thickness (A243424): refining material strength of the plates.

NOTE They are found and fixed as working information for the hull design.

F.1.169 Refine working information (A2443): the most detailed and last stage of information to perform the production.

F.1.170 Request a ship* (A11): the first action or contact of a ship owner when intending to order a ship.

NOTE Having definite ideas regarding appearance and functionality of the ship, the owner expresses these ideas in an inquiry to the shipyard.

F.1.171 Request classification items* (A2411): collecting the approvable items from the classification society with information about the components and the request for changes.

NOTE It produces a list of items which have to be calculated and approved.

F.1.172 request for production changes: changes that are requested to the ship design as a result of production experience or difficulties with the realization of the ship design.

F.1.173 resistance and shaft power*: the opposition to motion that results in hydrodynamics and powering estimates.

NOTE Resistance and shaft power is a constraint on the creation of the preliminary hull form.

F.1.174 resources: the shipyard, classification society, and outside consultants.

F.1.175 schedule*: the plan for governing the timing of the production phases.

NOTE Formed as a part of the final design process.

F.1.176 scrapping plan*: the document used to schedule the time and resources required to dismantle the ship.

F.1.177 secondary structure design: the structure detail design not defined in early design, as input for handling the plates and profiles.

F.1.178 shell and longitudinal bulkhead walls: the watertight skin of the ship and the bulkhead walls between transverse frames.

F.1.179 ship: a large waterborne vessel whose design, manufacture, and lifecycle operation is governed by the principles of naval architecture and in accordance with international and classification society regulations.

F.1.180 ship product model data: the product data of the ship accumulated throughout its lifecycle.

NOTE Since scrapping is part of the lifecycle, the ship is not an output, only the documented information and knowledge about the ship survives.

F.1.181 ship weight modifications: revisions to ship weight due to the preliminary structure design.

NOTE This is fed back to modify the preliminary hull form and revise the preliminary general arrangements.

F.1.182 shipyard: an organization that designs, builds, maintains, and repairs ships.

F.1.183 shipyard and classification society: the organizations to give mechanism to all ship building activities. Shipyards design, build, maintain, and repair ships. Classification societies approve the design and manufacture of ships.

F.1.184 shipyard and consultants: the organizations to give mechanism to all ship building activities. Shipyards design, build, maintain, and repair ships. Consultants are organizations that provide specific engineering services to shipyards.

F.1.185 shipyard/owner: the organizations to give mechanism to all ship building activities. The owner is the resource that operates the ship.

F.1.186 Specify ship (A1): the production of precise specifications for a ship prior to a contract being placed.

F.1.187 Statutory and class survey (A43): preparative and executive tasks to perform a survey with focus on the statutory survey to meet the requirements of the class certificate.

F.1.188 steel sub-sections: the sub-sections of the steel structure which are outfitted with the machinery and distribution systems before assembly.

F.1.189 structural design: design of the ship foundation and framework.

EXAMPLE Includes the keel, hull, bulkheads, decks, superstructures, girders, stiffeners, and the like.

F.1.190 structural design in special areas: the design of the hull structure not including longitudinal and vertical structure.

F.1.191 survey report: the report on the results of inspections of ship machinery, structure, or other equipment.

F.1.192 survey status: the status of a survey at a particular point in time.

F.1.193 technical documentation: the detailed information about material parts needed for producing the ship and system.

NOTE In the case of maintenance, the technical documentation of a system means part of the product description required to perform preventative maintenance, repair, and failure analysis of that system.

F.1.194 technical requirements: the owner's specifications that must be realized by the completed ship.

F.1.195 test results: the documented evaluations and conclusions for each inspection, check, or analysis against design specifications, rules, and regulations.

NOTE As part of the ship maintenance throughout its lifecycle, test results are the results of functional tests carried out after the execution of maintenance actions.

F.1.196 Test ship (A34): inspect and evaluate the actual ship against the design specifications, rules and regulations, and contract requirements.

NOTE The structure is tested and sea trials are carried out. The test results are documented.

F.1.197 Test structures (A341): the steel structures are inspected and evaluated against the design specifications, rules and regulations, and contract requirements.

NOTE The output is the test result documentation.

F.1.198 Test systems* (A342): the ship systems including outfitting, machinery, and mission systems are inspected, put into service, and evaluated against the design specifications, rules and regulations, and contract requirements.

NOTE The output is the test result documentation.

F.1.199 Transform piping and electrical design into production design* (A24422): a function to transform the conceptual data from design (CA) into material demand and assembling information.

F.1.200 transportation need: a constraint which determines the specification for the ship construction.

F.1.201 updated survey status: the updated status of a survey after carrying a statutory and class survey.

F.1.202 WCoG information: the information about weights and centres of gravity.

F.1.203 weights and centres of gravity: the amount that things weigh and the centres of mass for those things.

NOTE Weights and centres of gravity necessary for further calculations.

F.1.204 Work out plans for outfitting and equipment* (A24423): the function that contains information for the various workforces in scheduling, controlling, and coordinating their activities in their specific fields.

F.1.205 working information: the information for production stage of the ship structures.

F.1.206 workload: the total effort required to build the chosen ship design as estimated by the shipyard and assisting consultants.

F.2 Application activity model diagrams

The application activity model diagrams are given in F.2 through F.24. The graphical form of the application activity model is presented in the IDEF0 activity modelling format (see SD, SC4 n1217, clause 9.9.2.1.2). Activities and data flows that are out of scope are marked with asterisks.

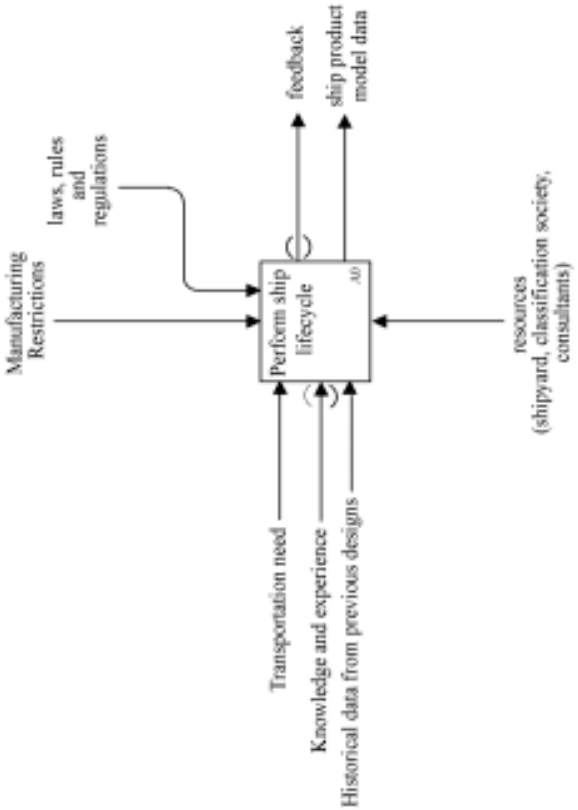


Figure F.2 — Node A-0: Describe ship life cycle

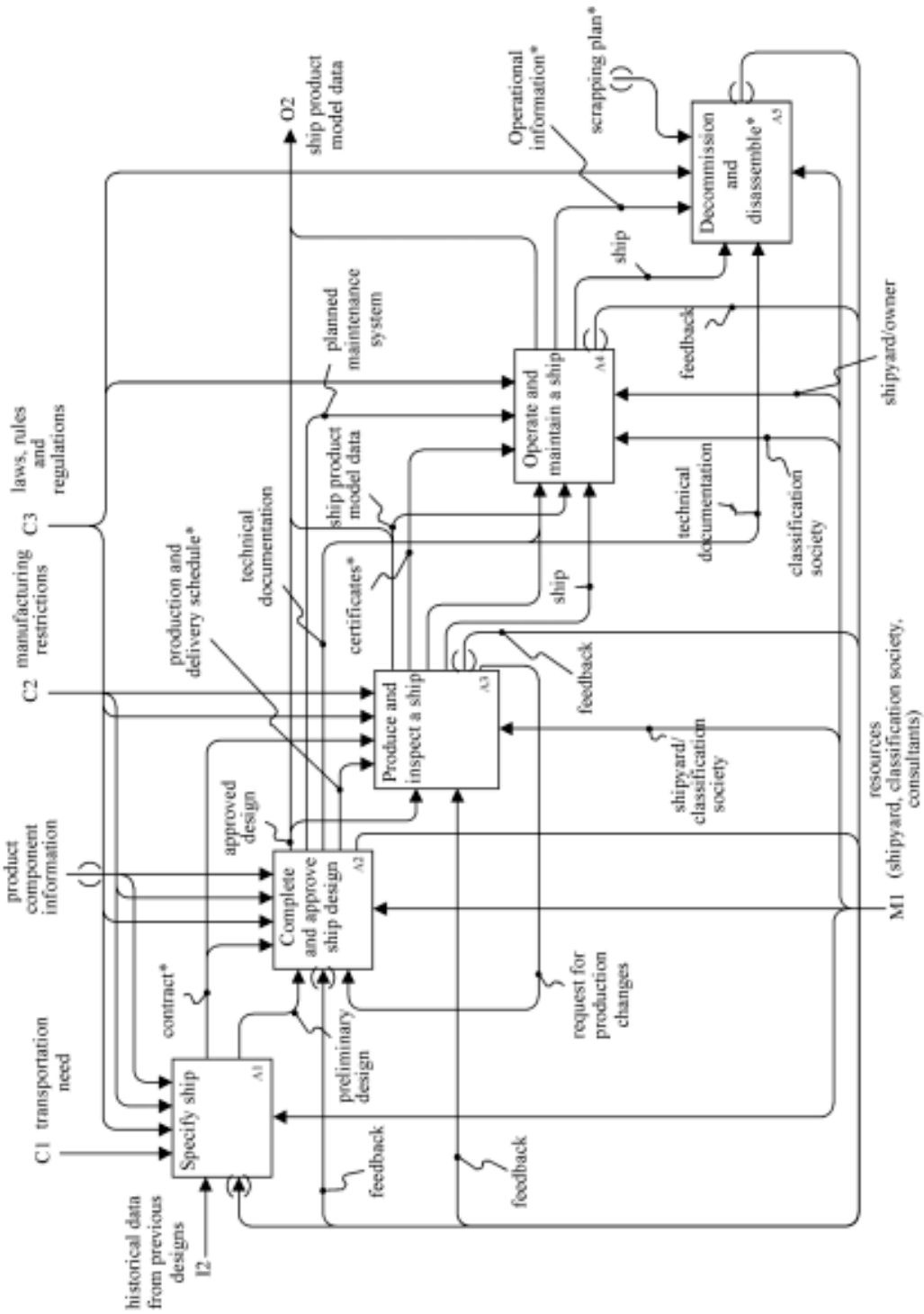


Figure F.3 — Node A0: Perform ship life cycle

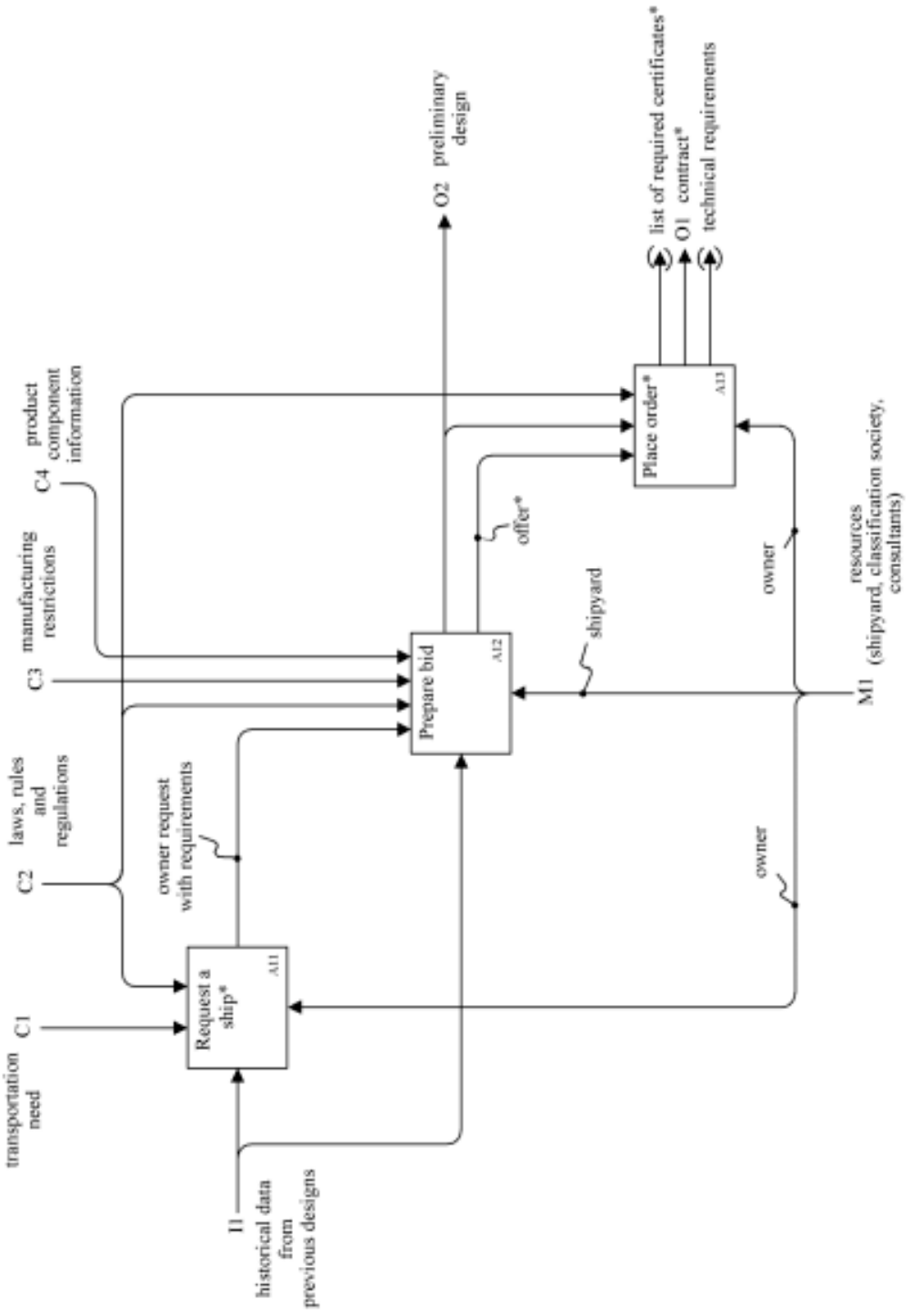


Figure F.4 — Node A1: Specify ship

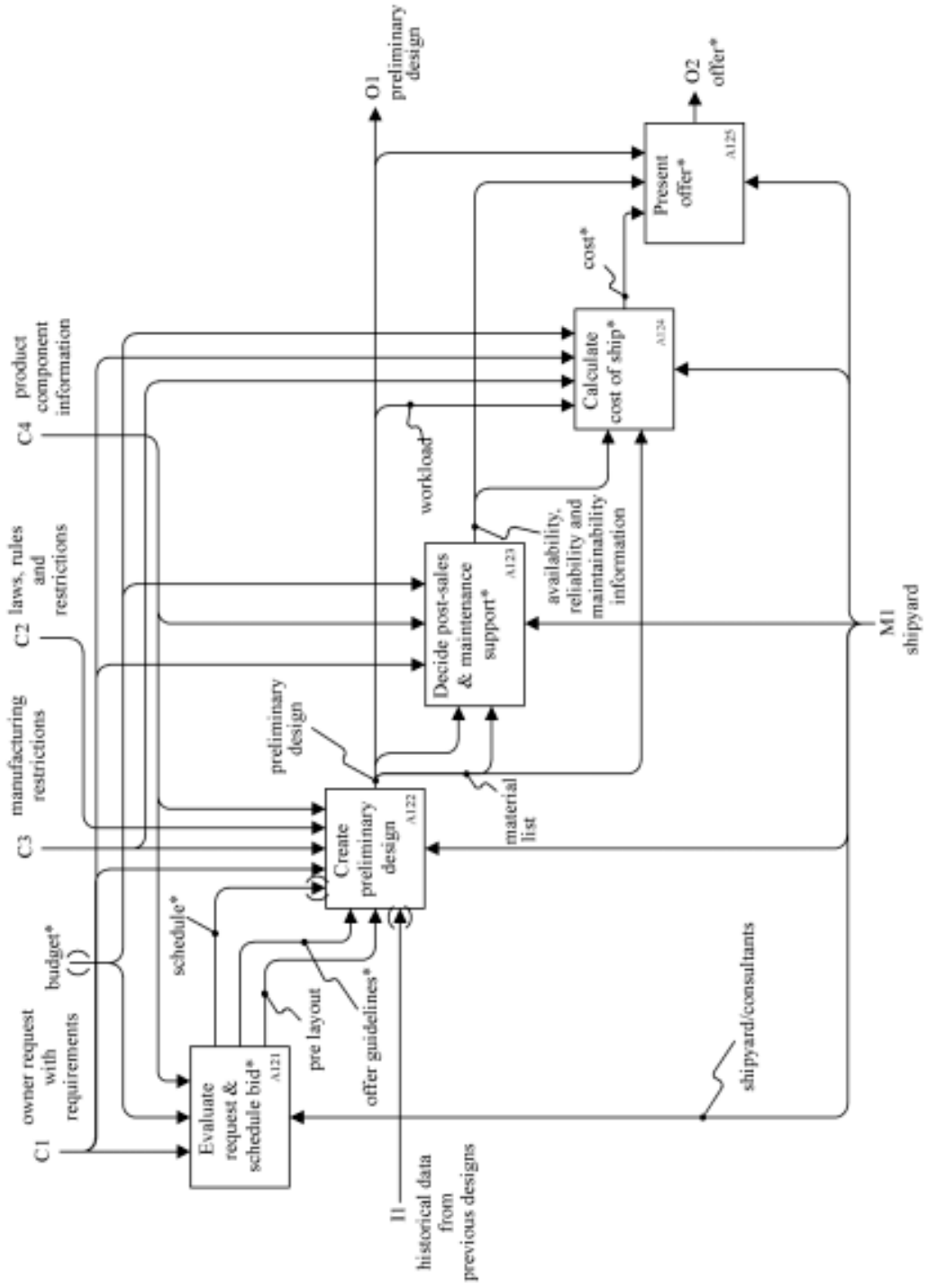


Figure F.5 — A12: Prepare bid

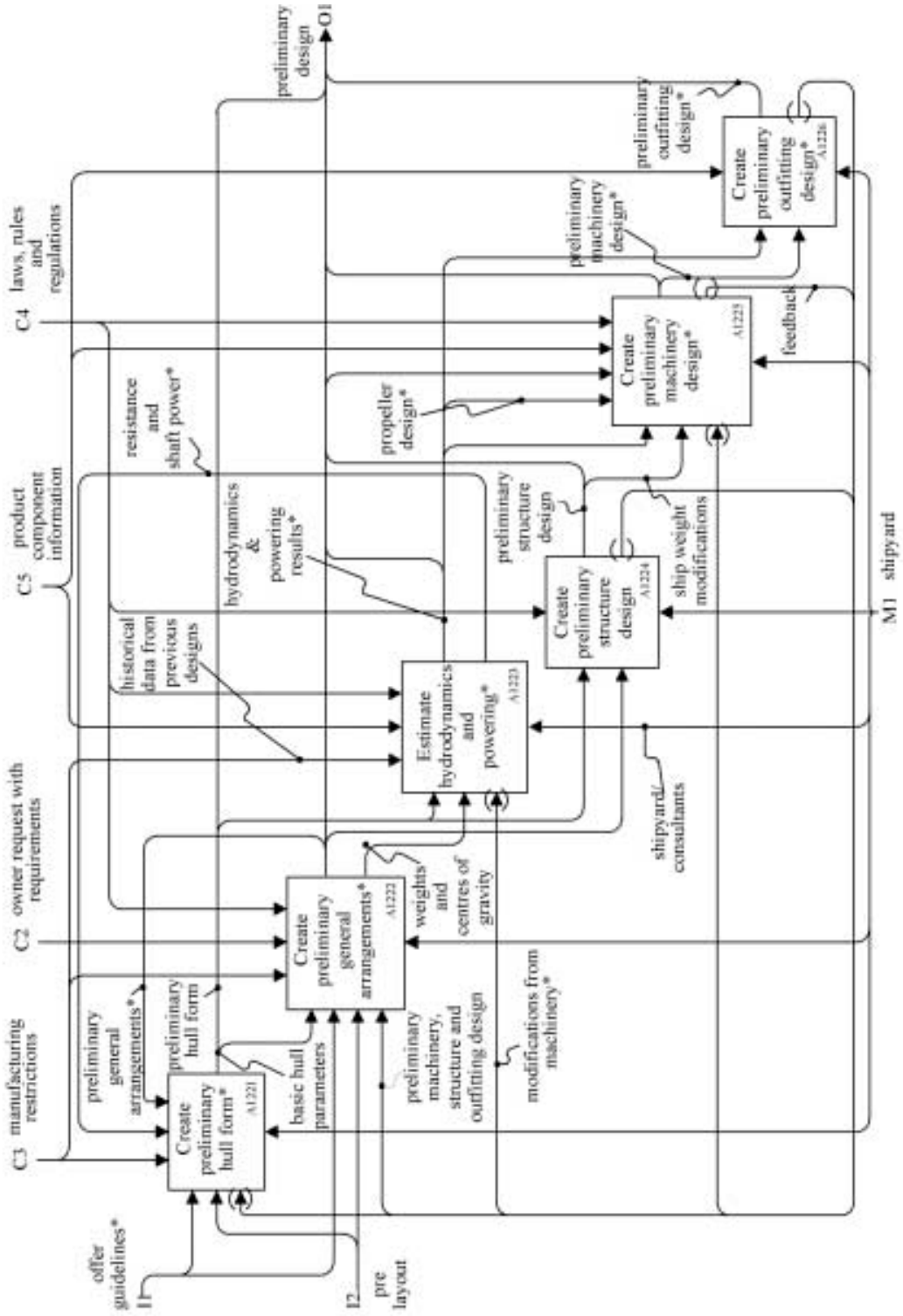


Figure F.6 — A122: Create preliminary design

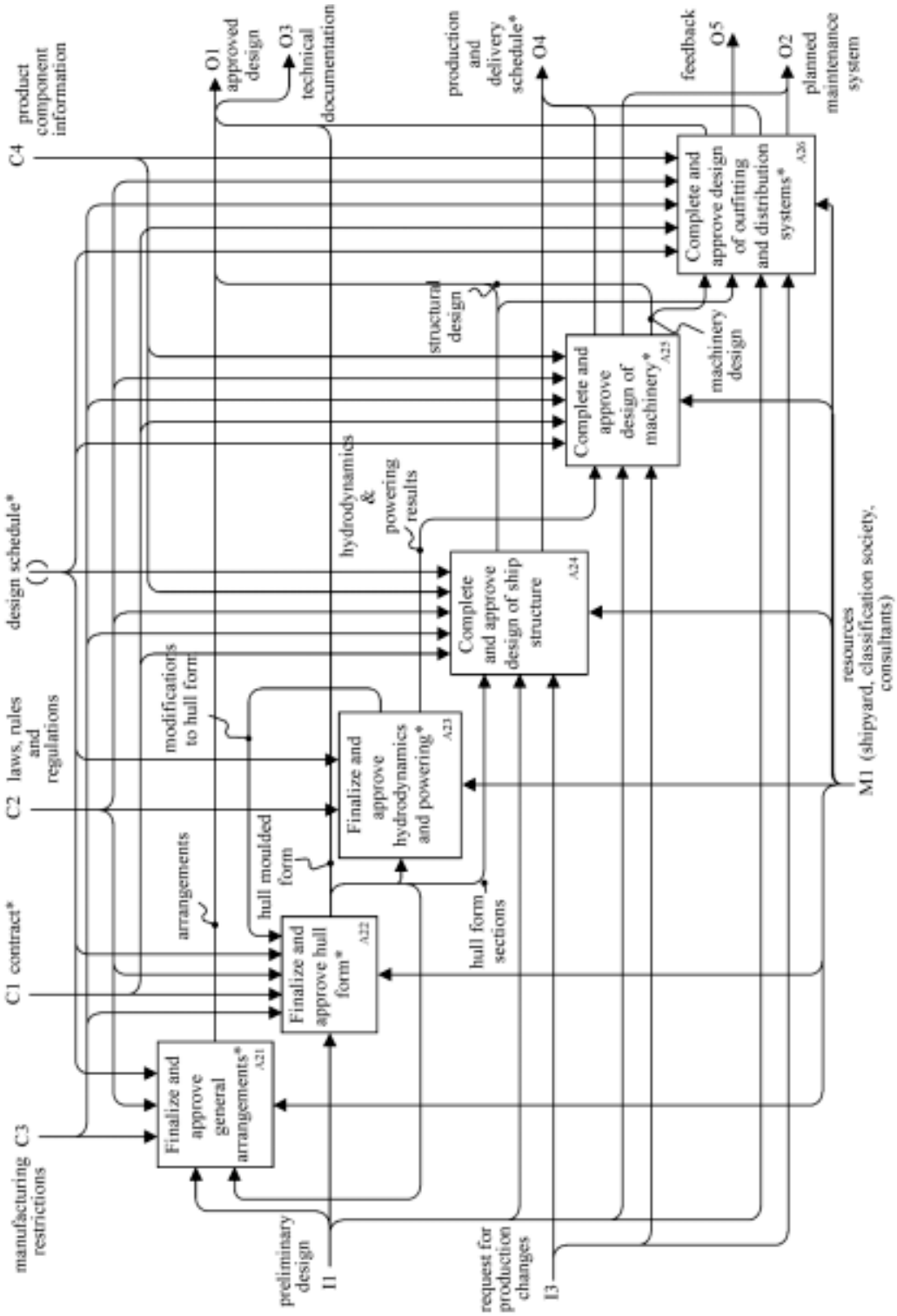


Figure F.7 — Node A2: Complete and approve ship design

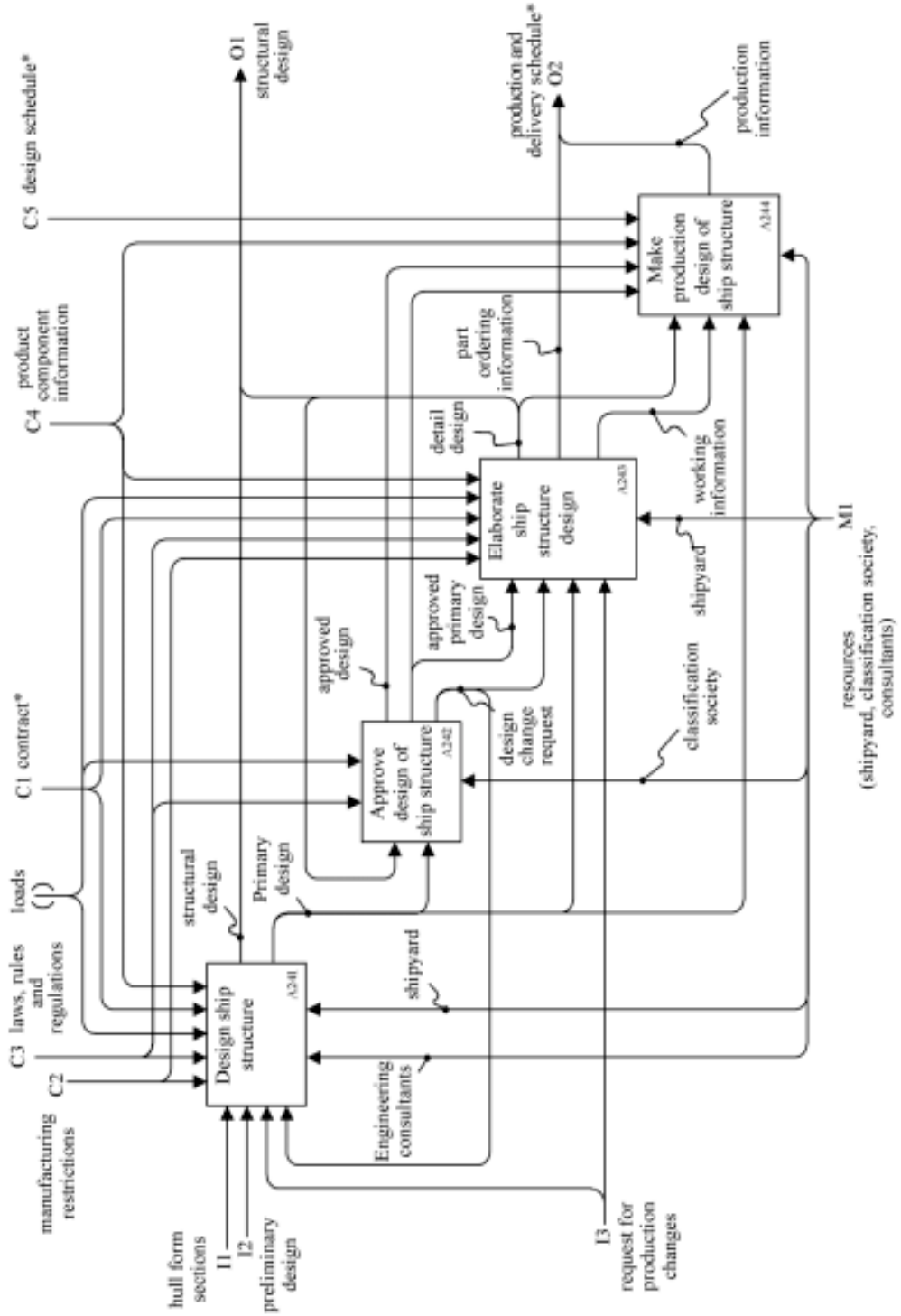


Figure F.8 — Node A24: Complete design of ship structure

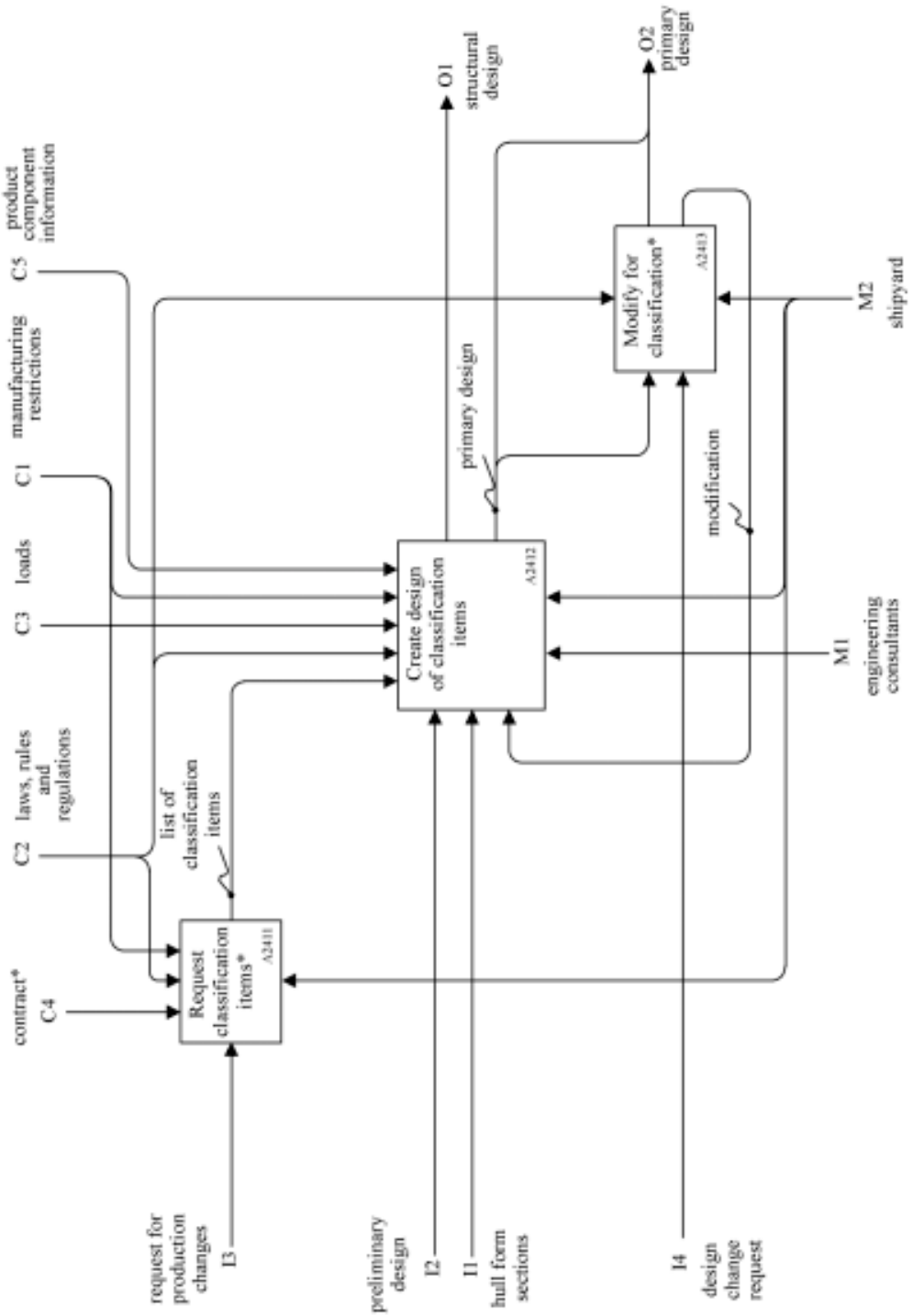


Figure F.9 — Node A241: Design ship structure

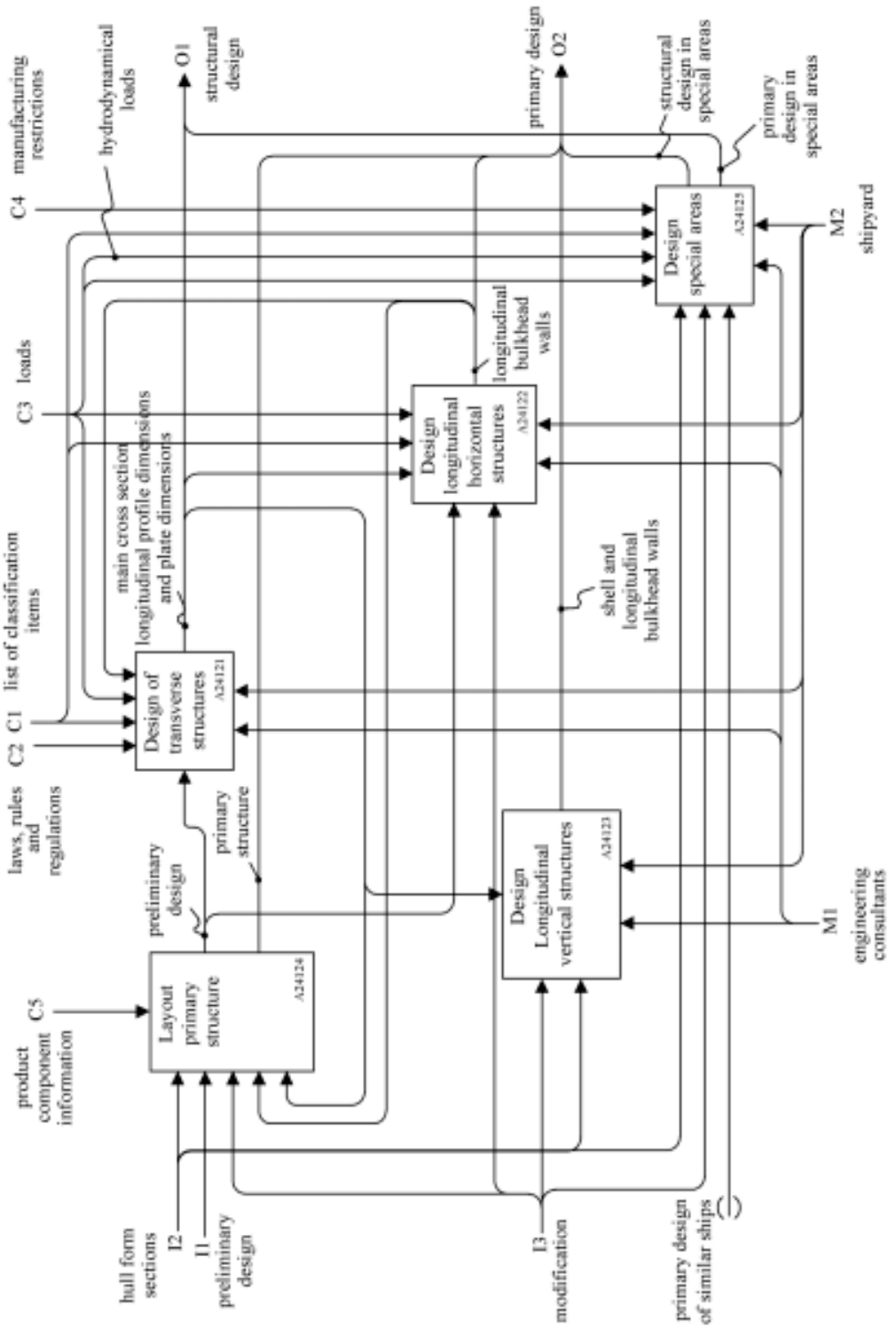


Figure F.10 — Node A2412: Create design of classification items

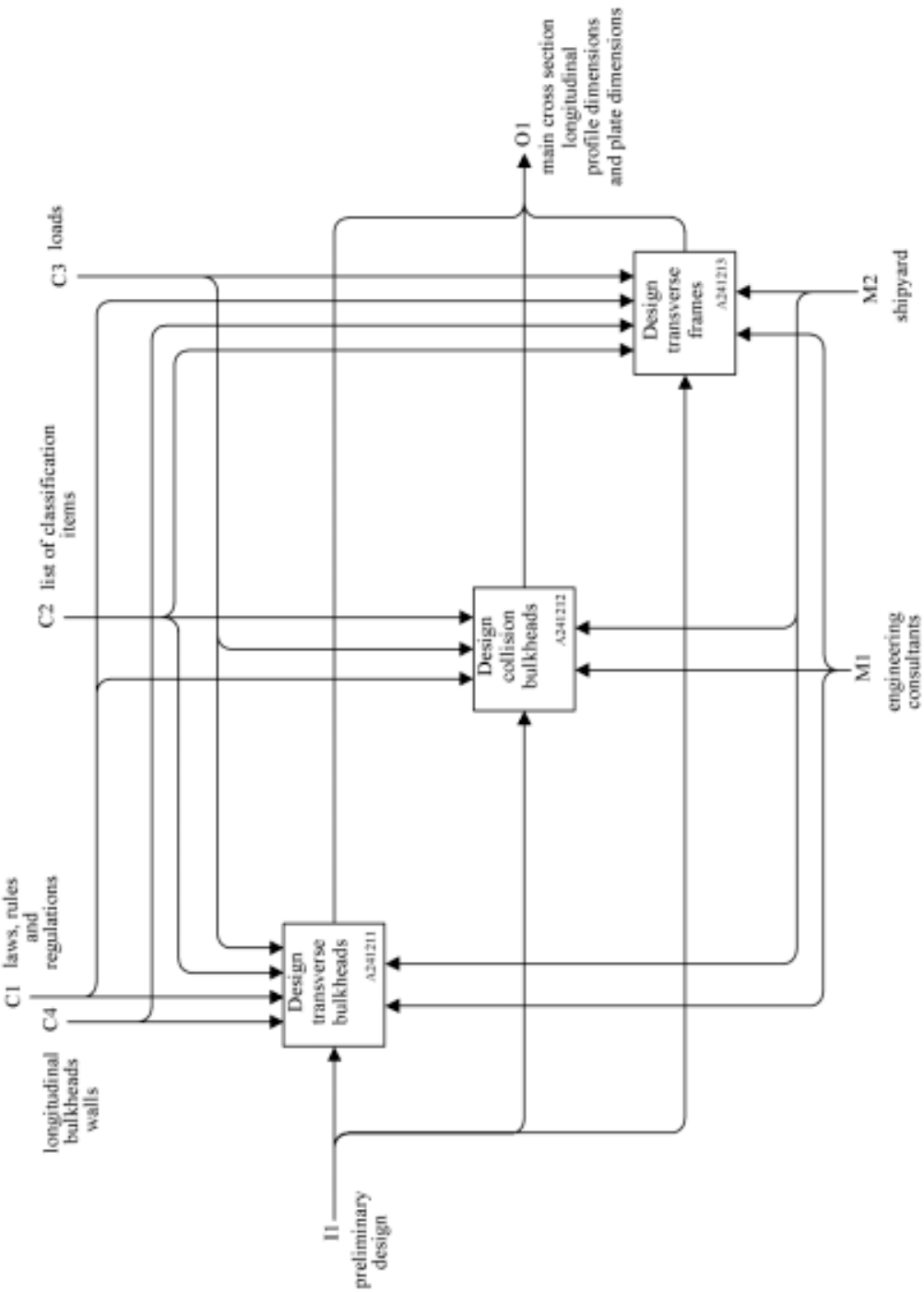


Figure F.11 — Node A24121: Design of transverse structure

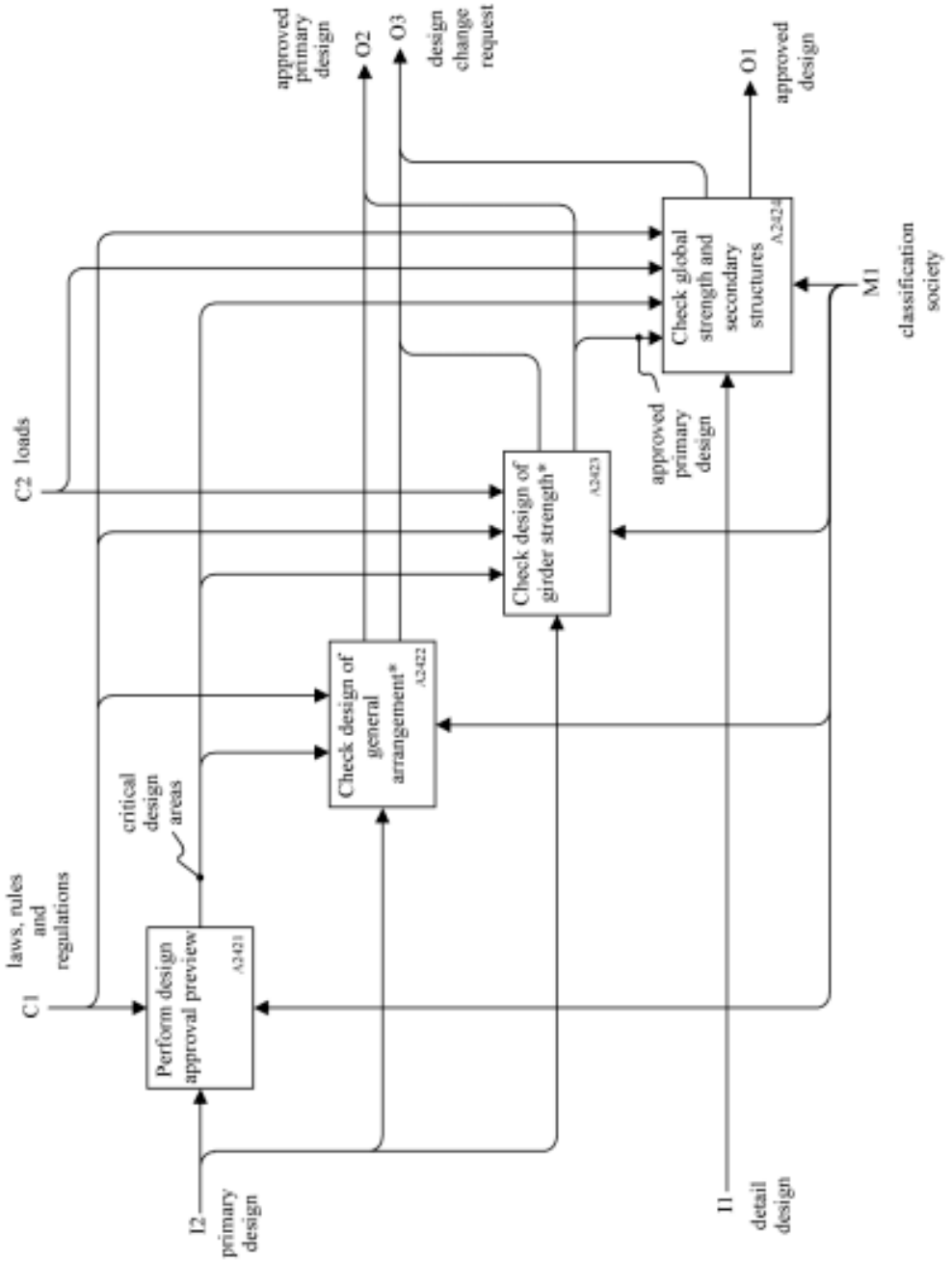


Figure F.12 — Node A242: Approve design of ship sructure

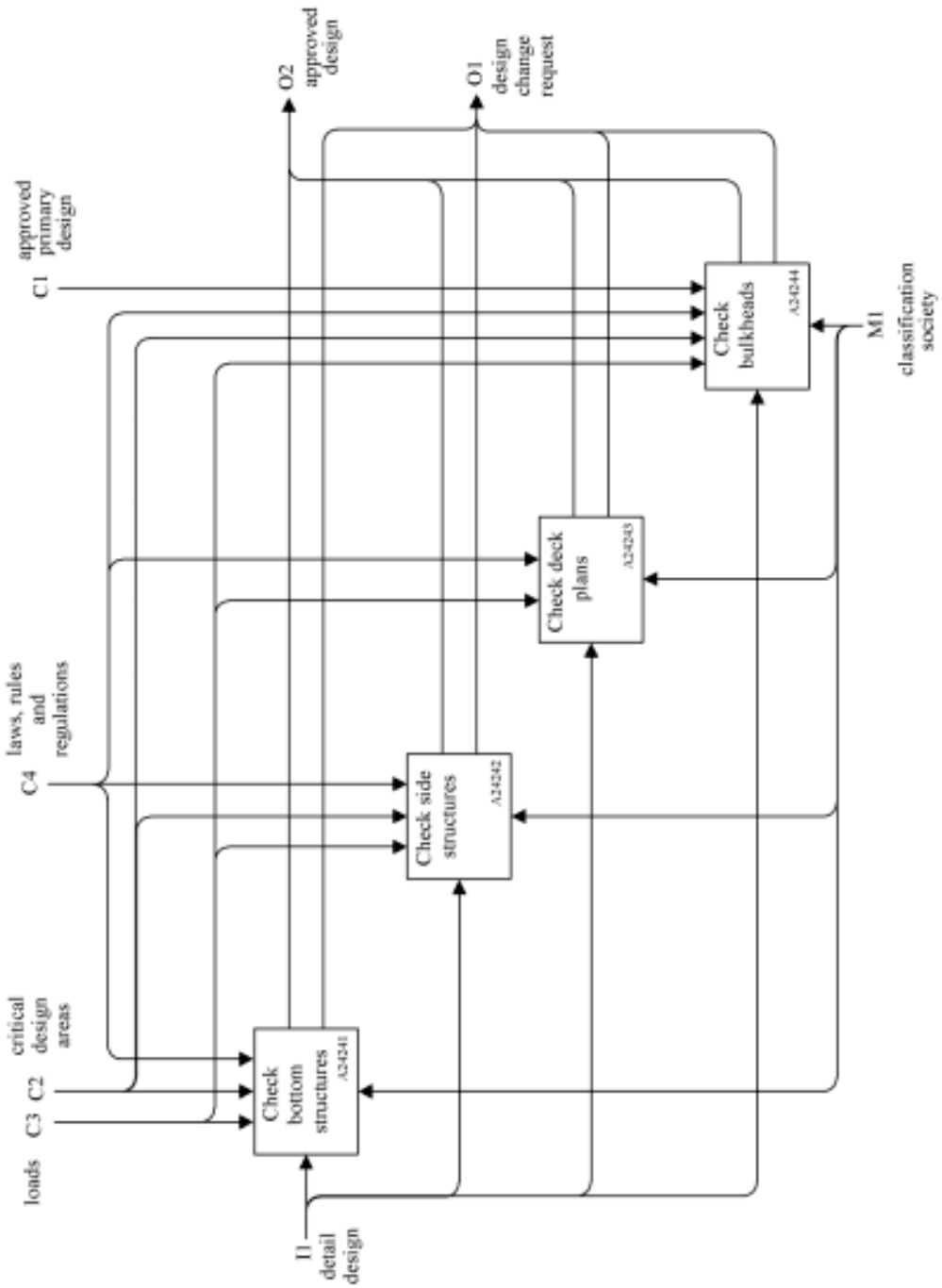


Figure F.13 — Node A2424: Check global strength and secondary structure

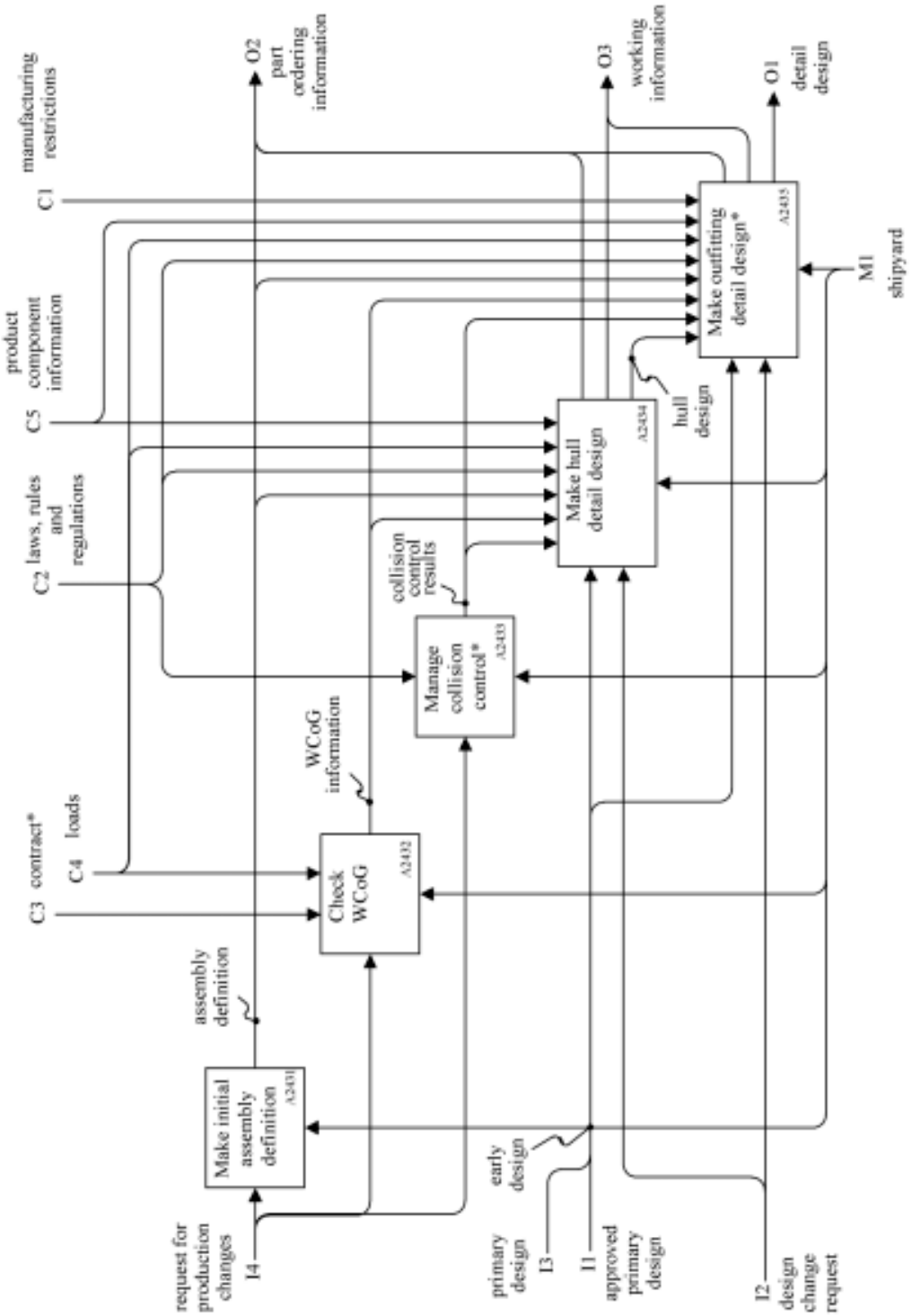


Figure F.14 — Node A243: Elaborate ship structure design

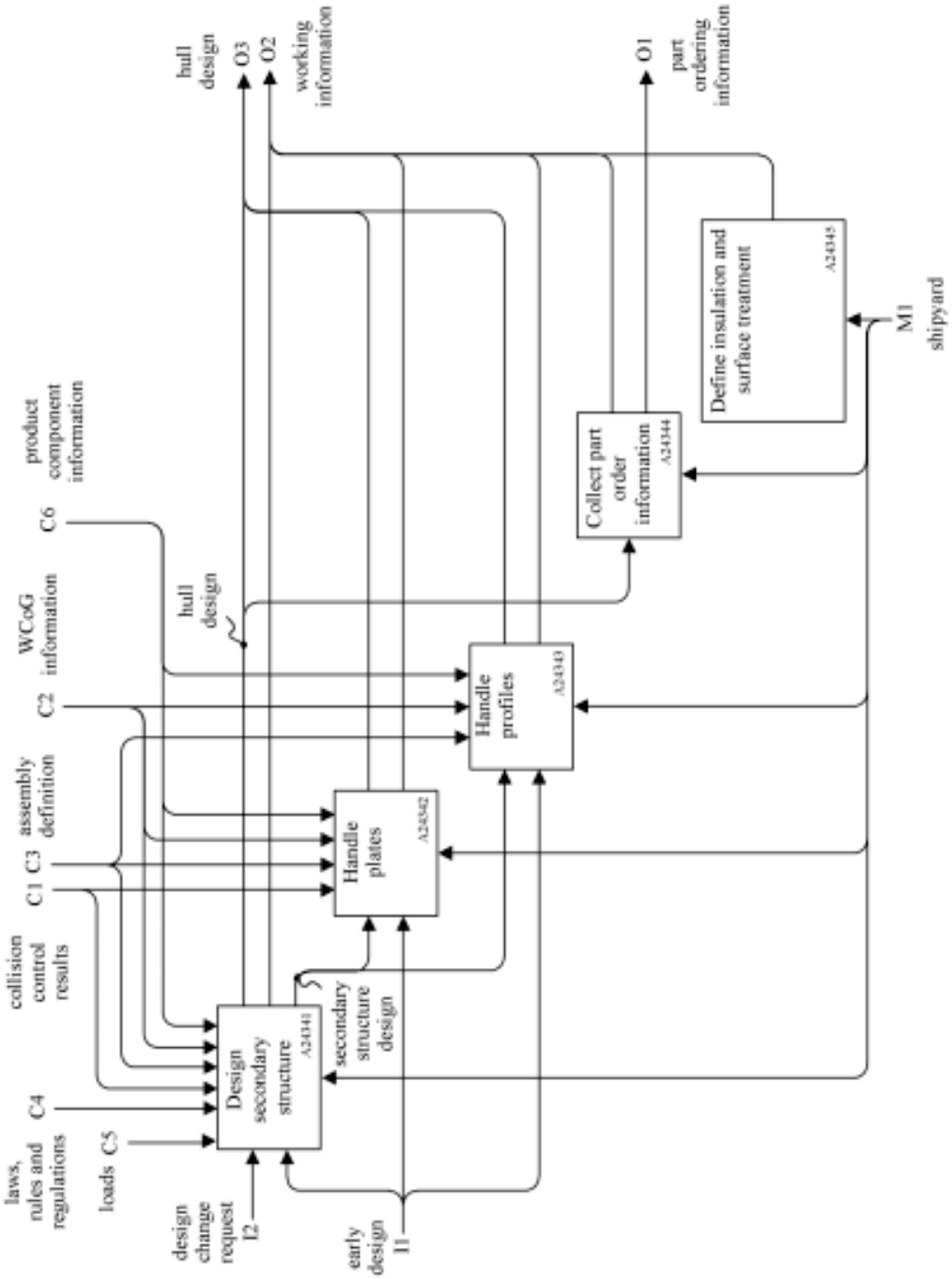


Figure F.15 — Node A2434: Make hull detail design

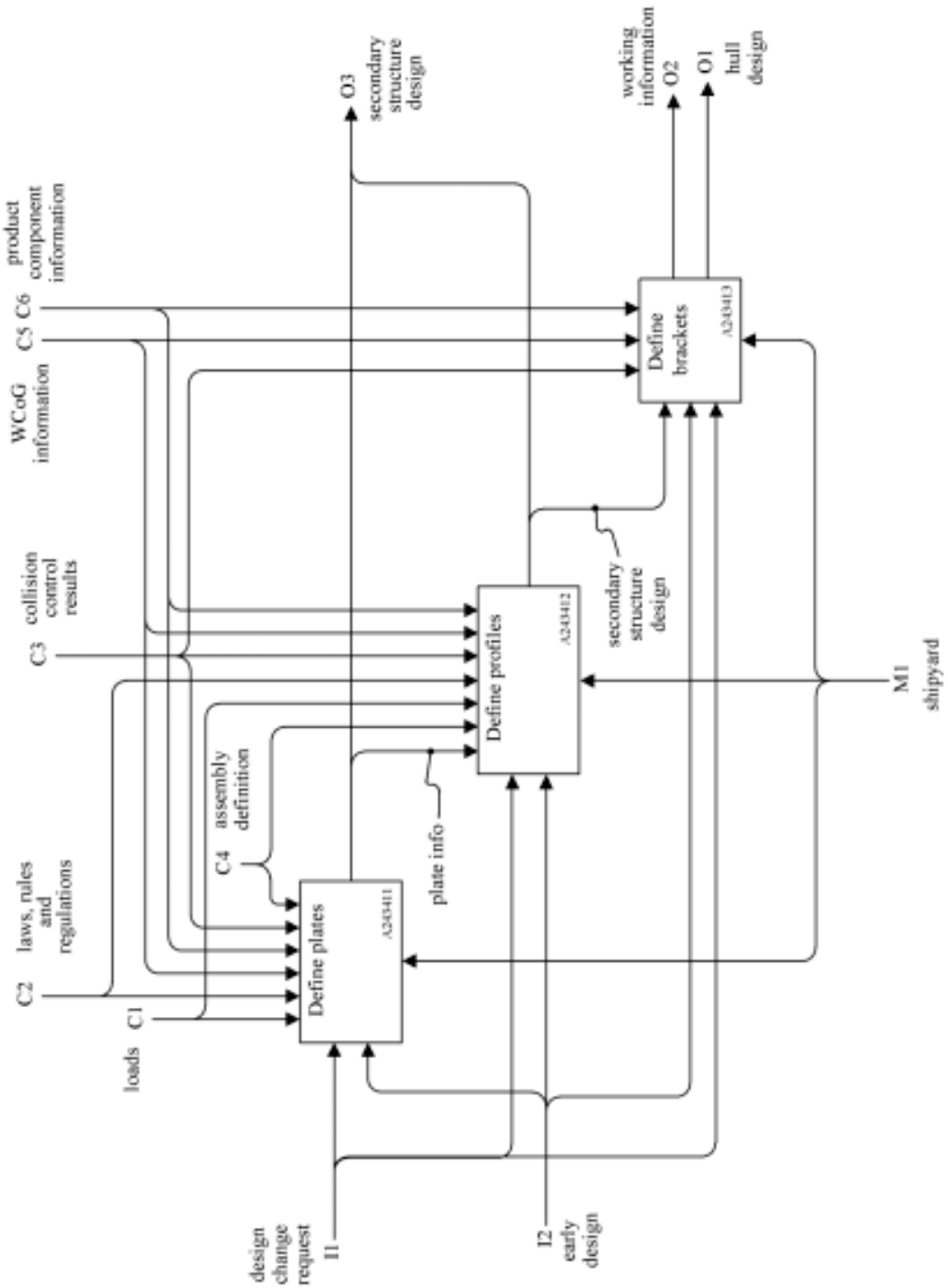


Figure F.16 — Node A24341: Design secondary structure

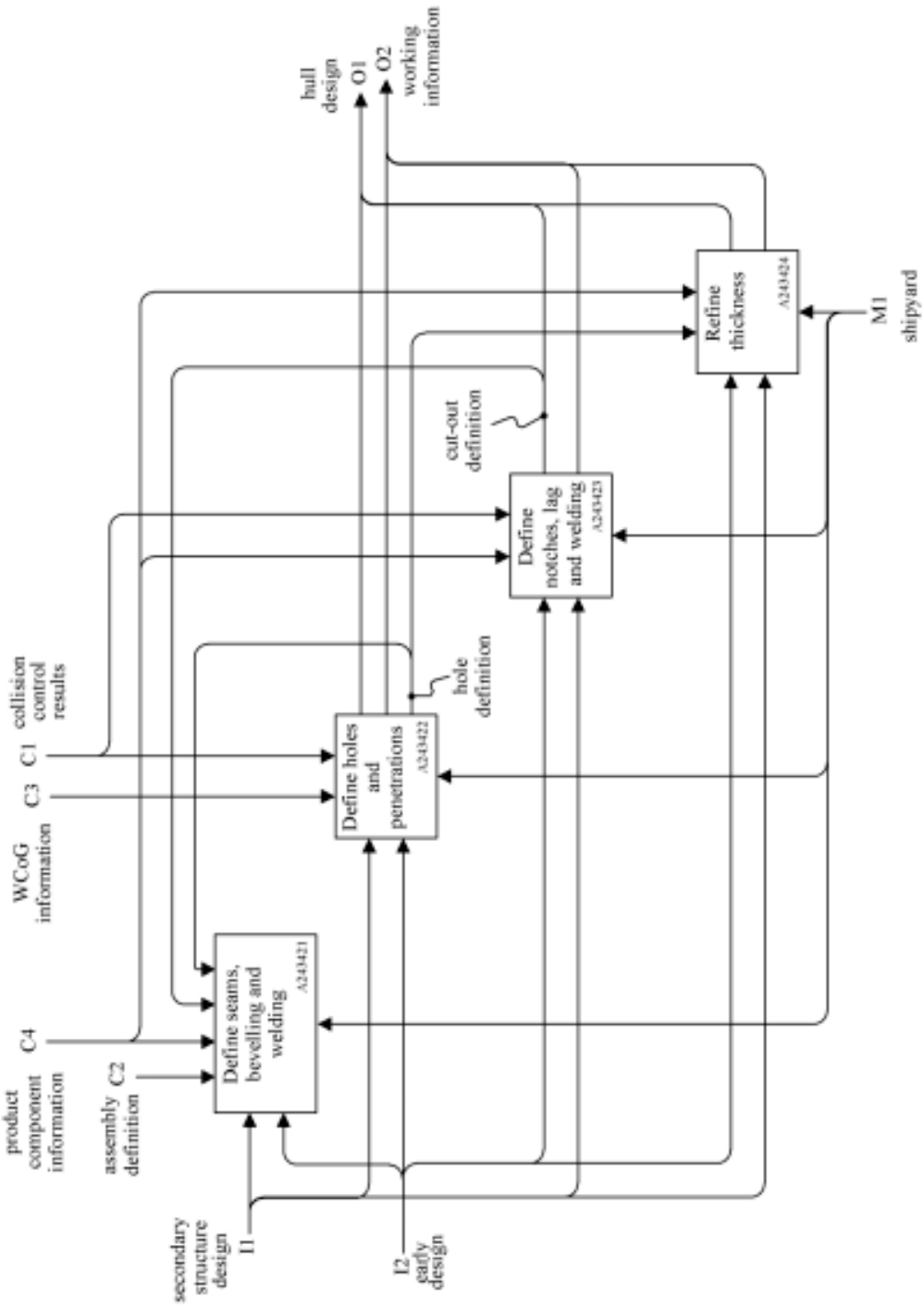


Figure F.17 — Node A24342: Handle plates

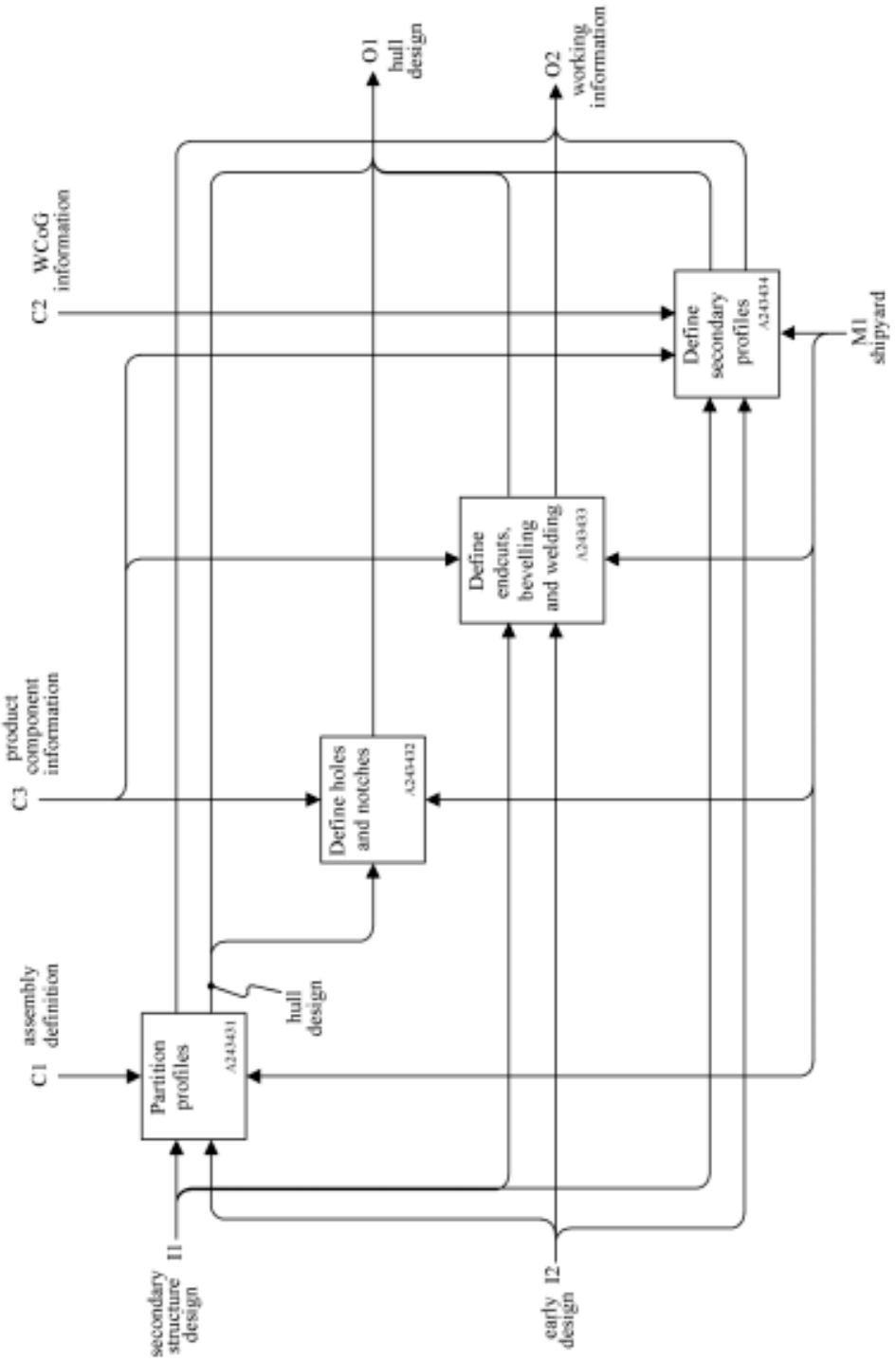


Figure F.18 — Node A24343: Handle profiles

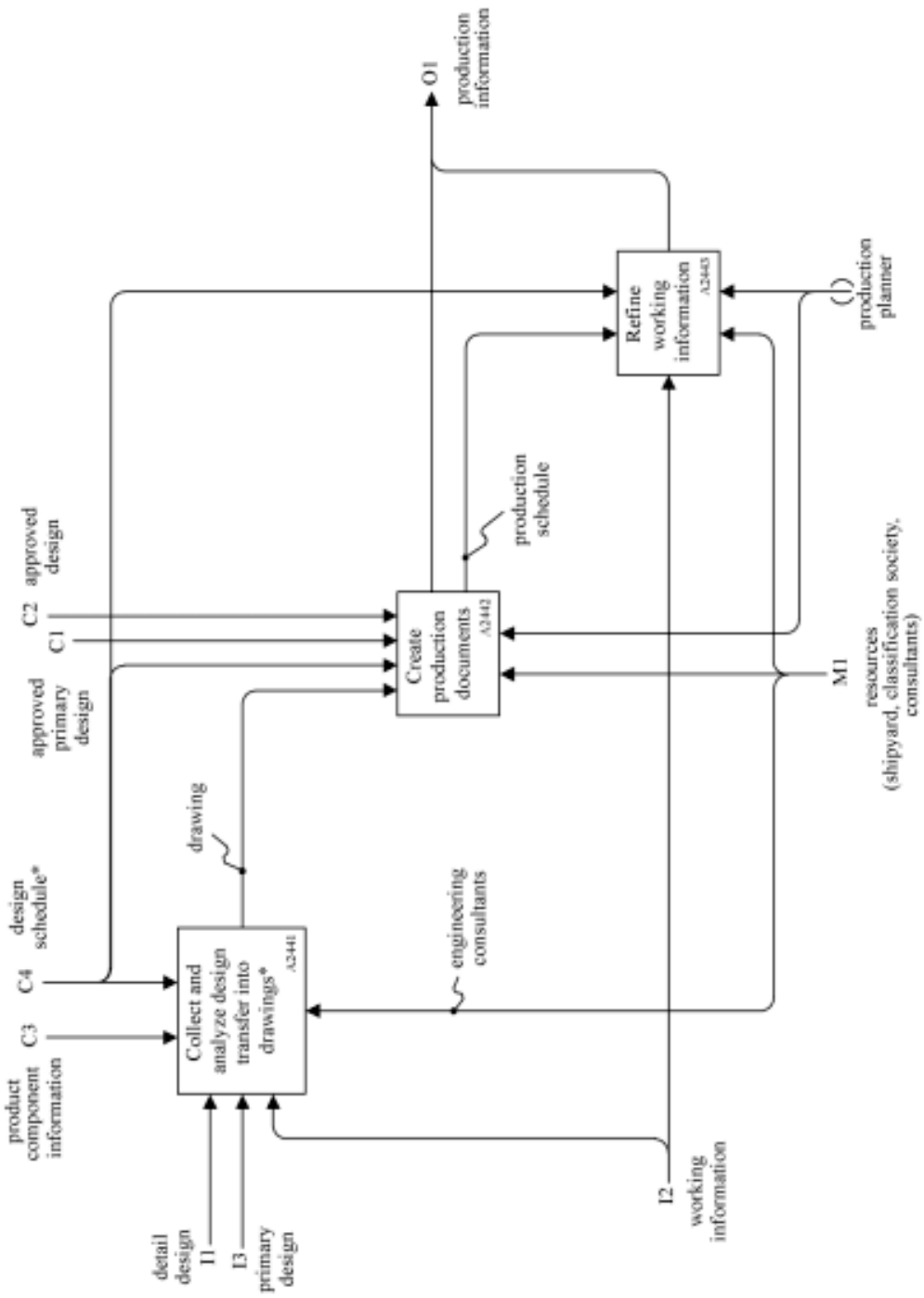


Figure F.19 — Node A244: Make production design of ship structure

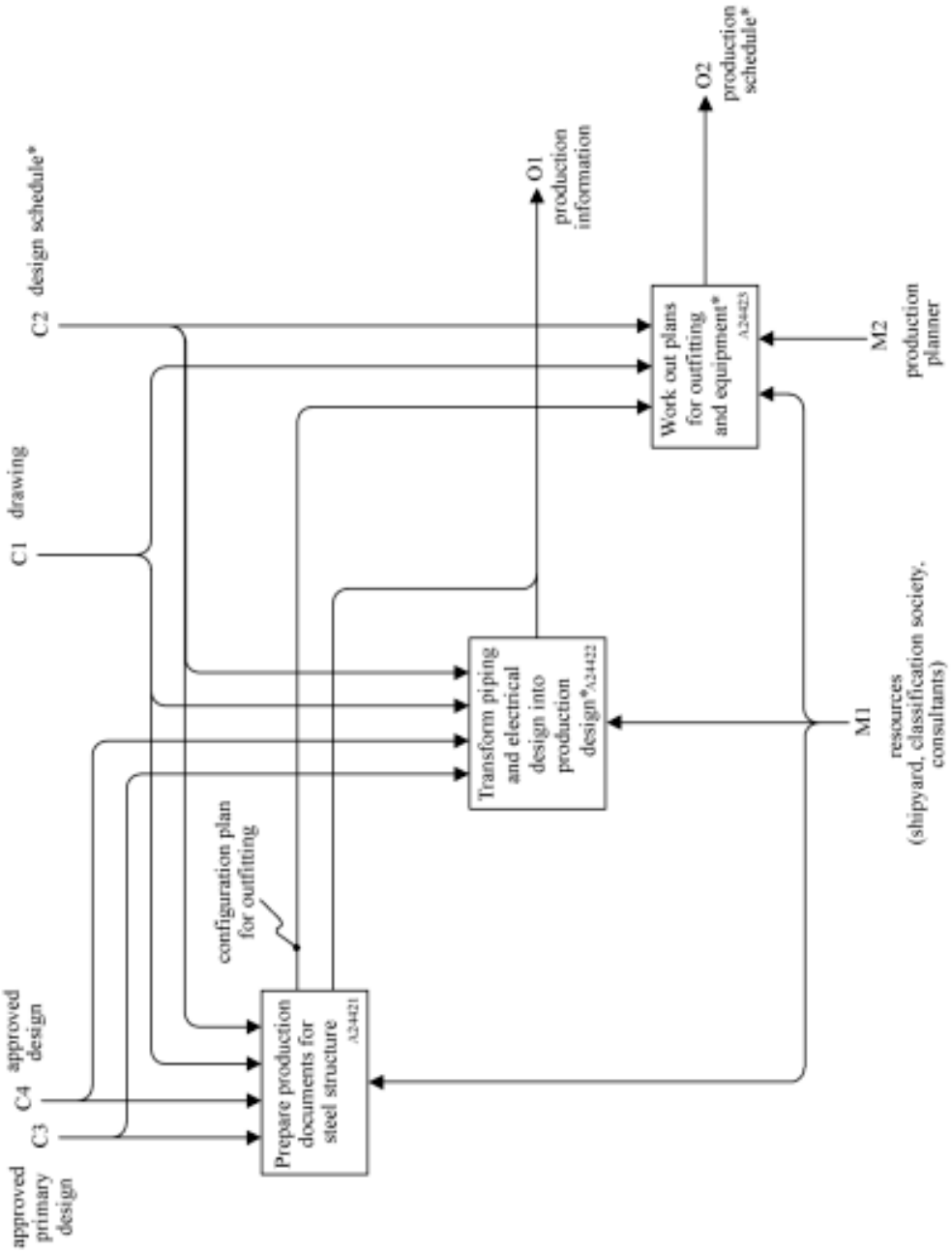


Figure F.20 — Node A2442: Create production documents

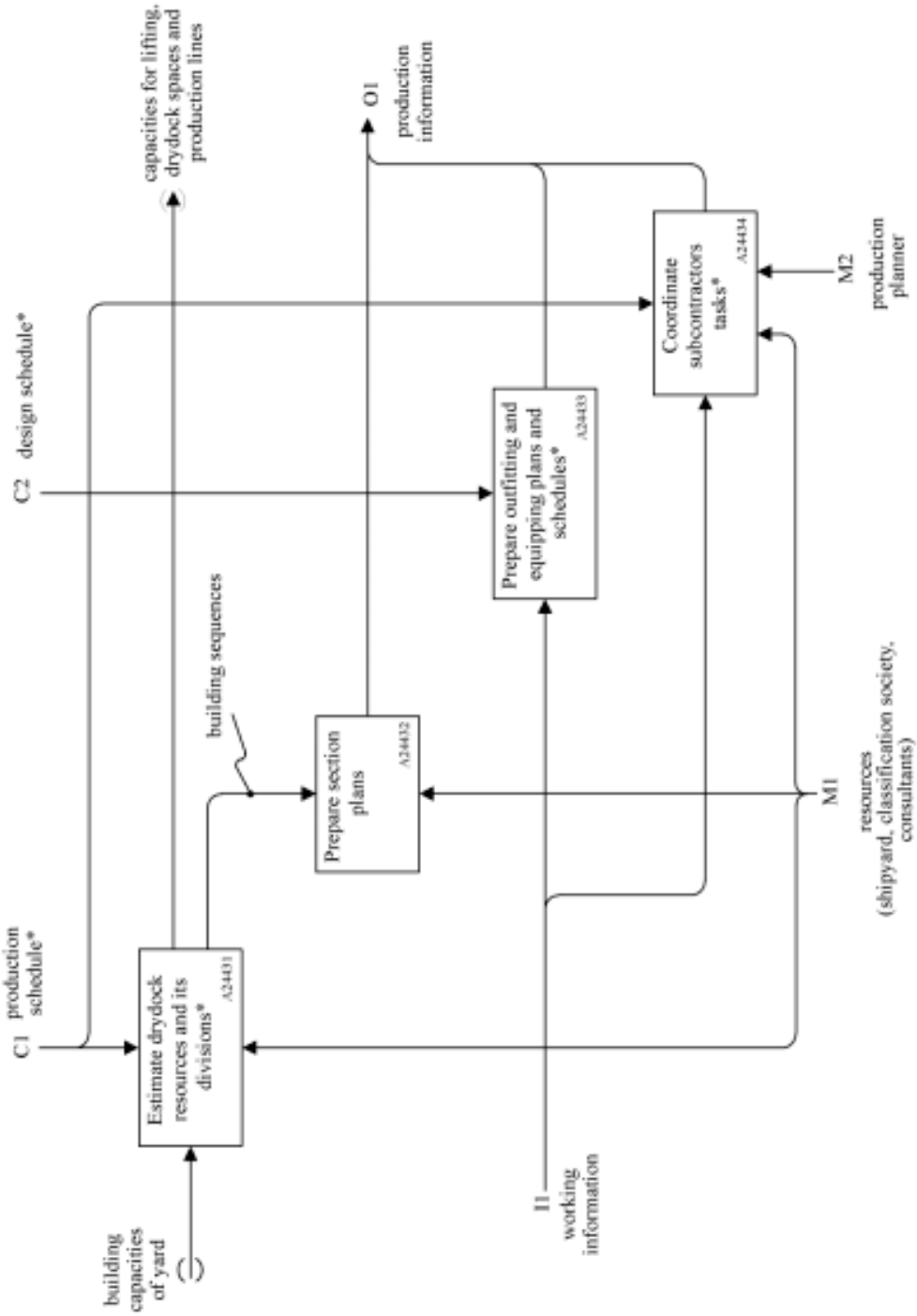


Figure F.21 — Node A2443: Refine working information

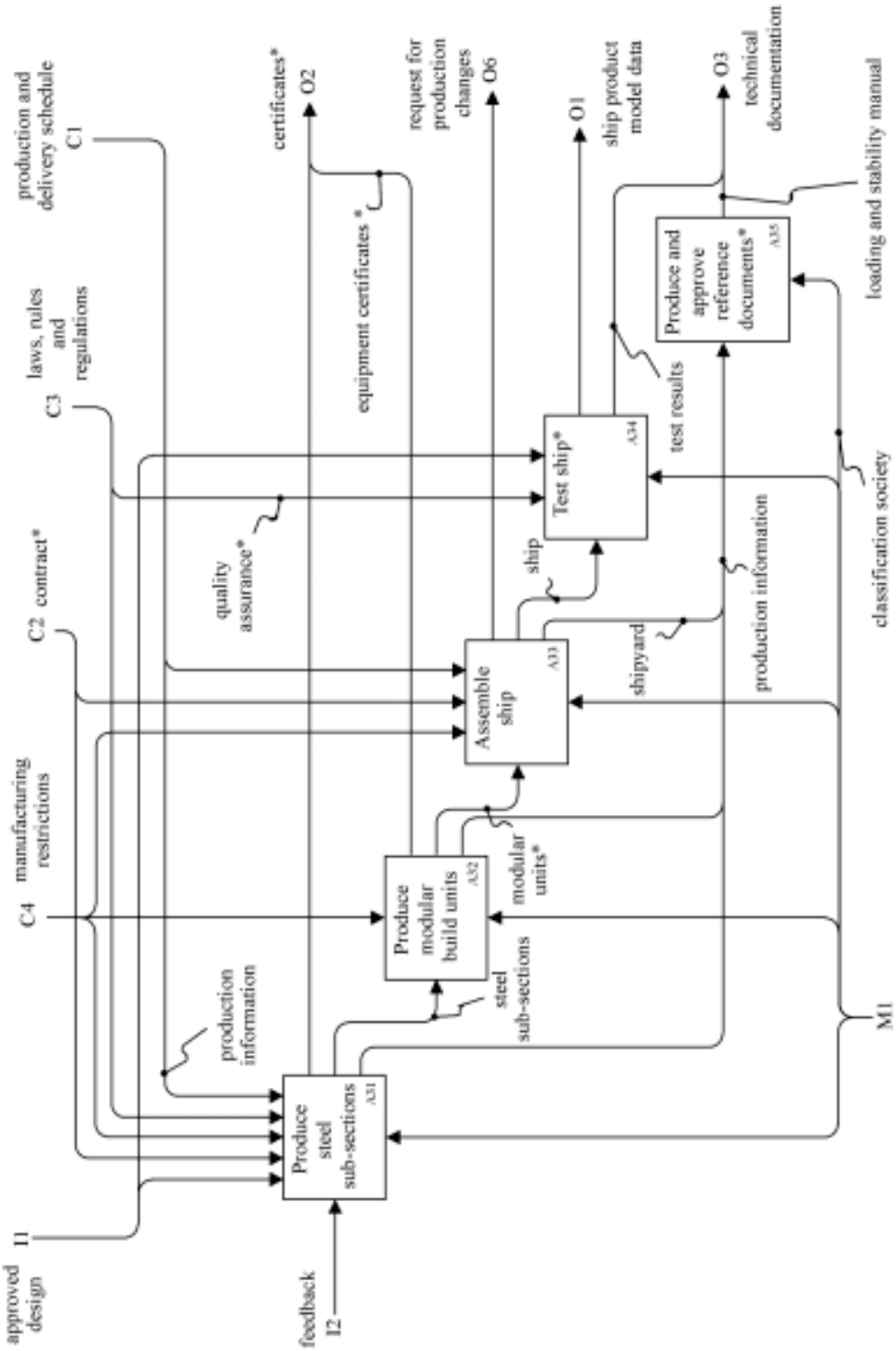


Figure F.22 — Node A3: Produce and inspect a ship

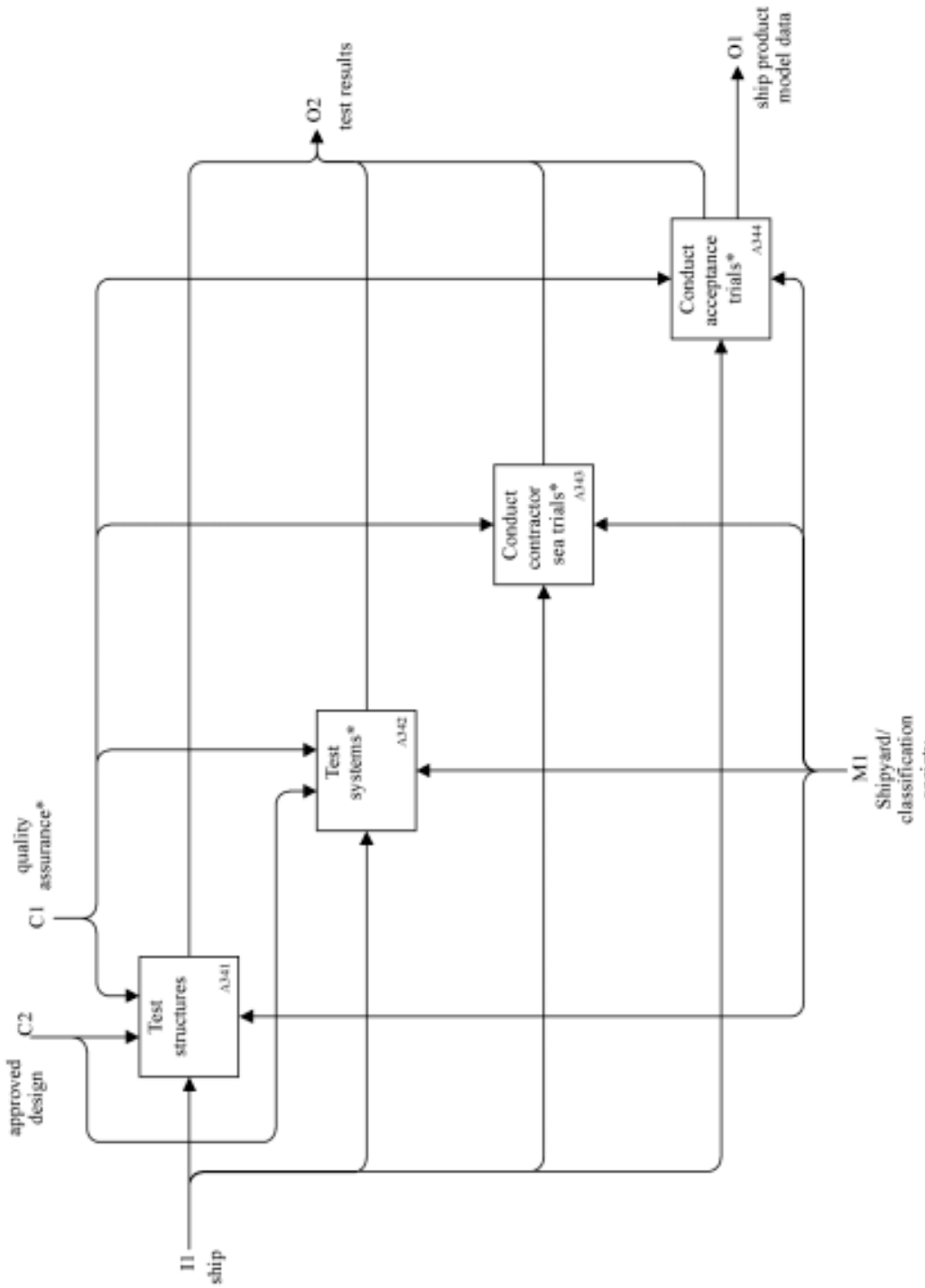


Figure F.23 — Node A34: Test a ship

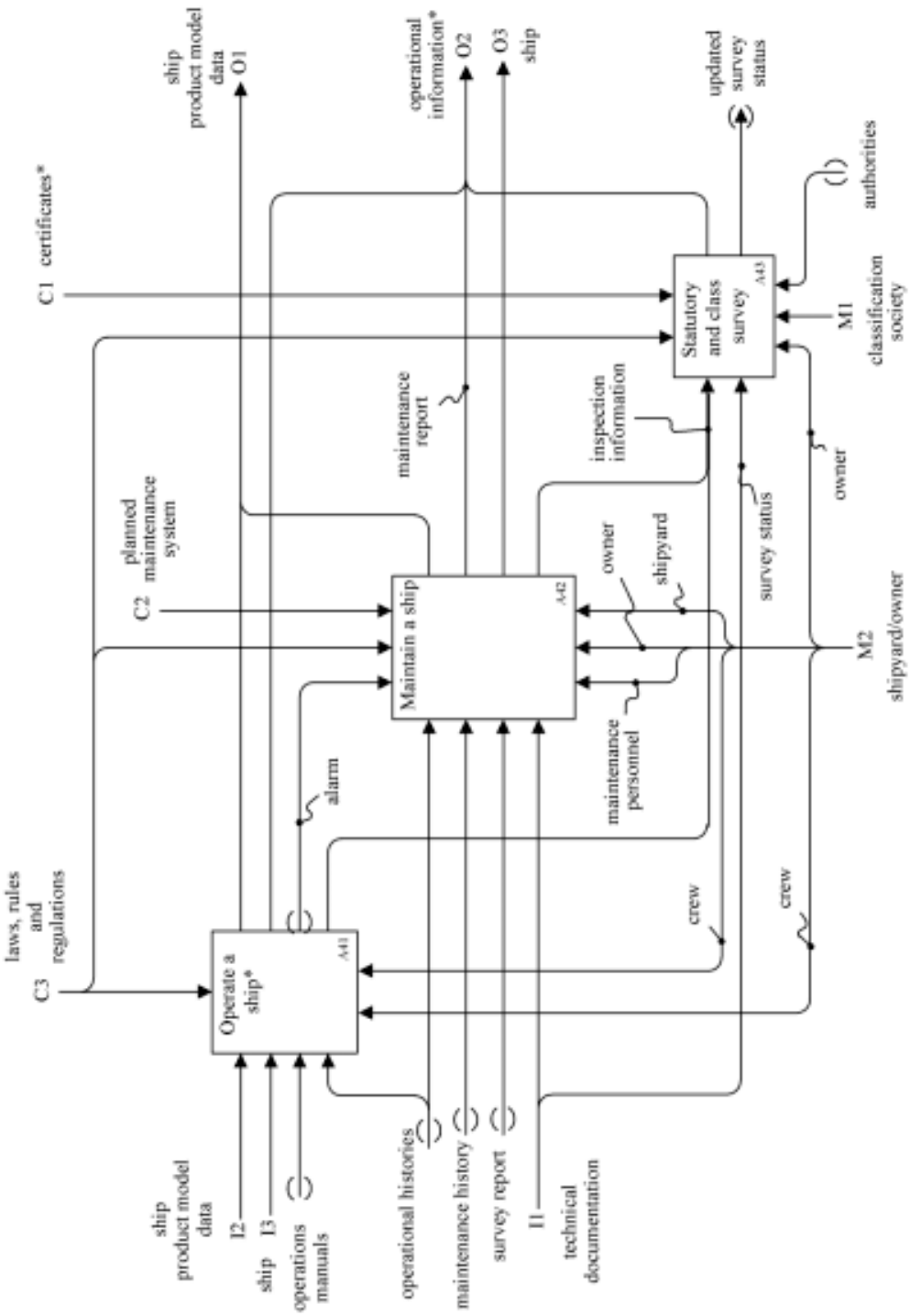


Figure F.24 — Node A4: Operate and maintain a ship

Annex G
(informative)

Application reference model

This annex provides the application reference model for this part of ISO 10303. The application reference model is a graphical representation of the structure and constraints of the application objects specified in clause 4. The graphical form of the application reference model is presented in EXPRESS-G. The application reference model is independent of any implementation method. EXPRESS-G is defined in Annex D of ISO 10303-11.

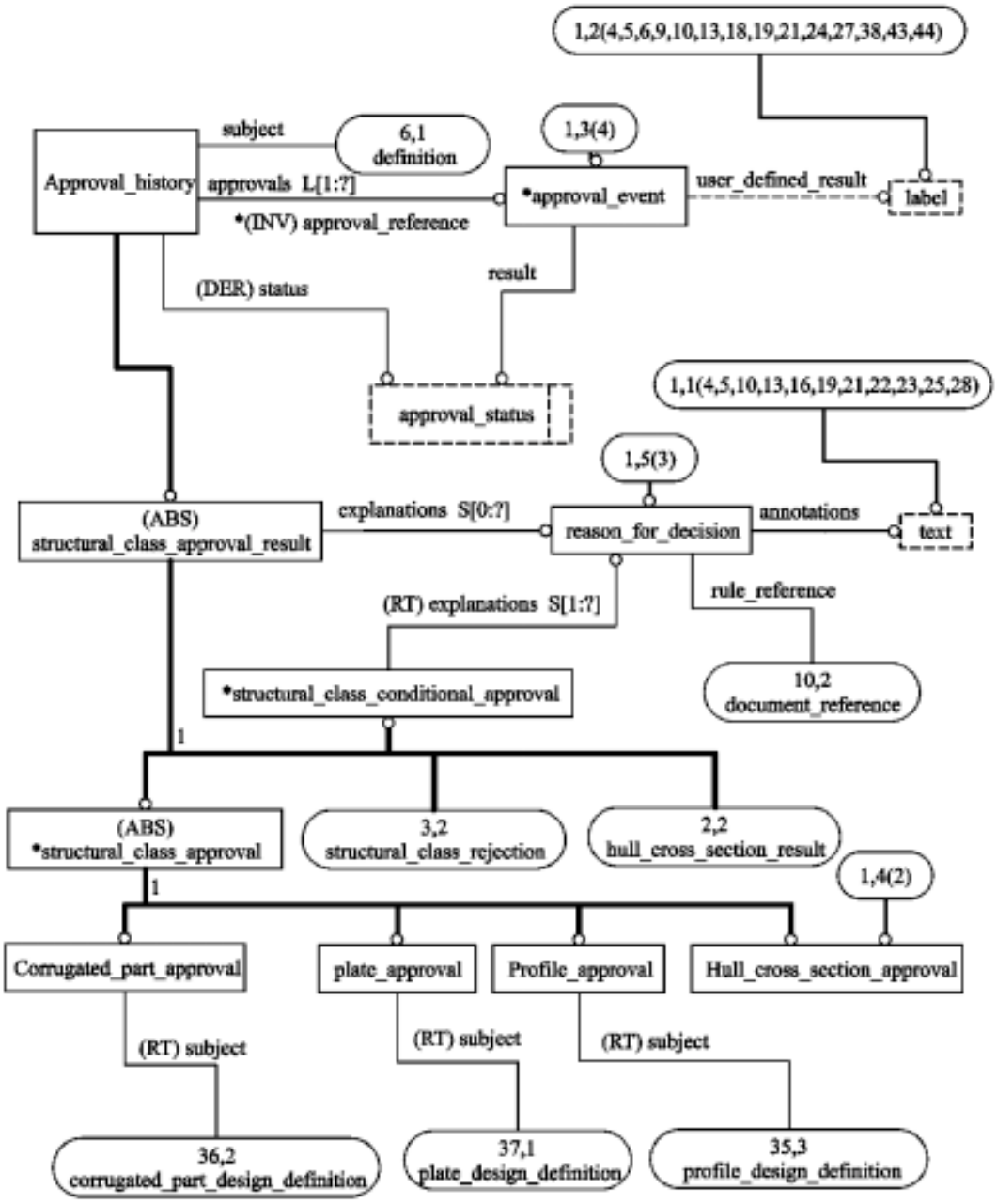


Figure G.1 — ARM EXPRESS-G diagram 1 of 49

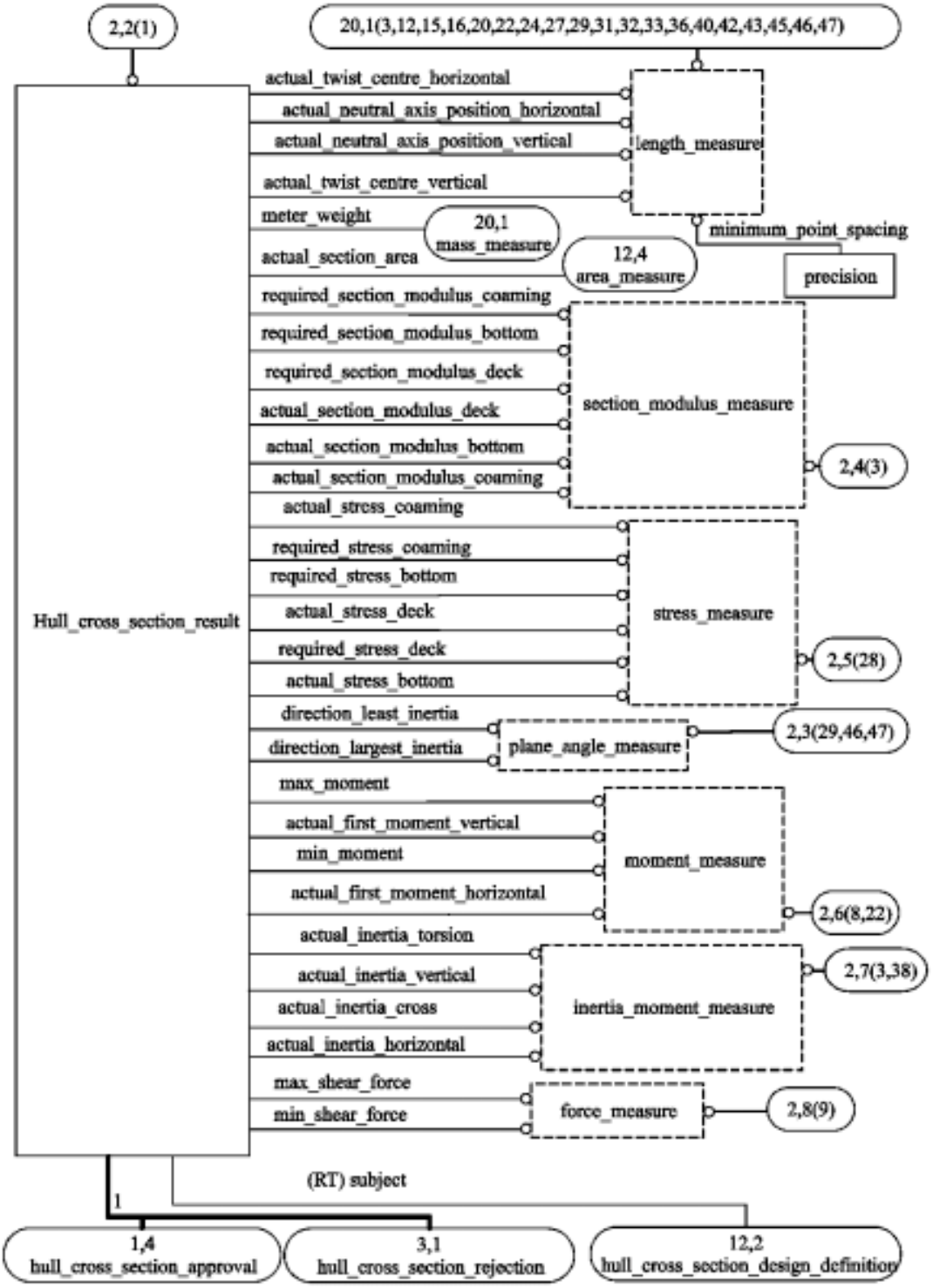


Figure G.2 — ARM EXPRESS-G diagram 2 of 49

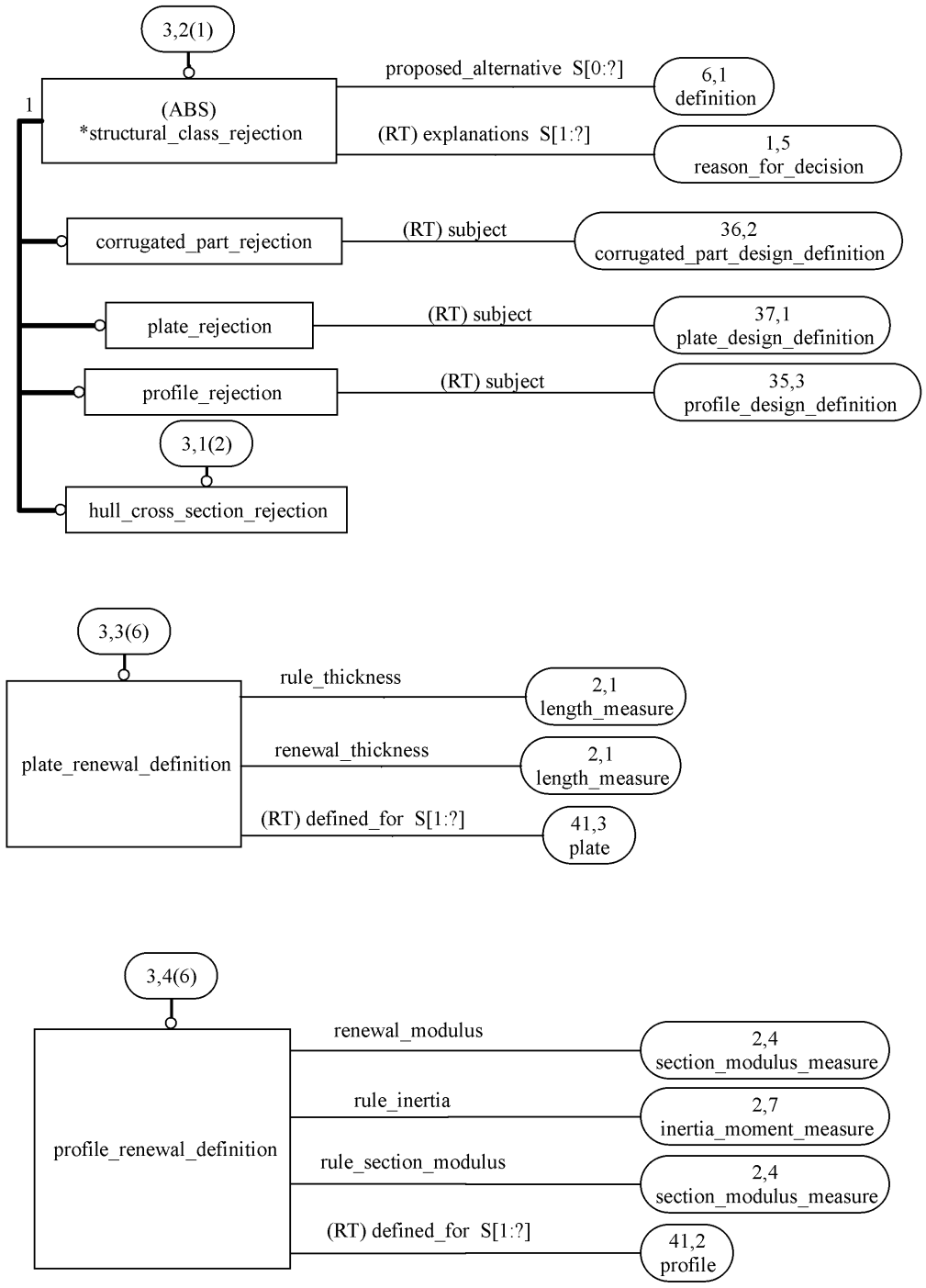


Figure G.3 — ARM EXPRESS-G diagram 3 of 49

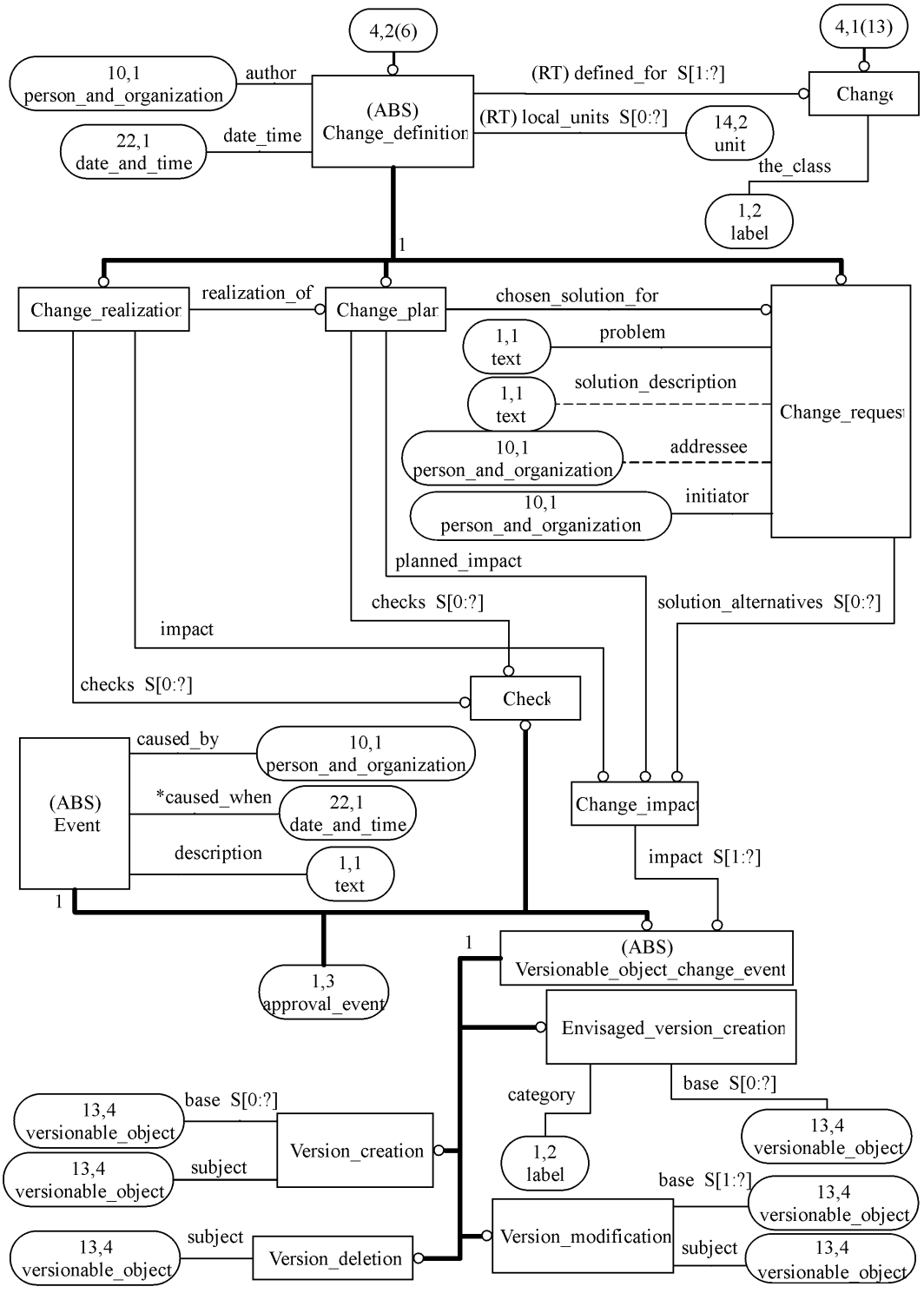


Figure G.4 — ARM EXPRESS-G diagram 4 of 49

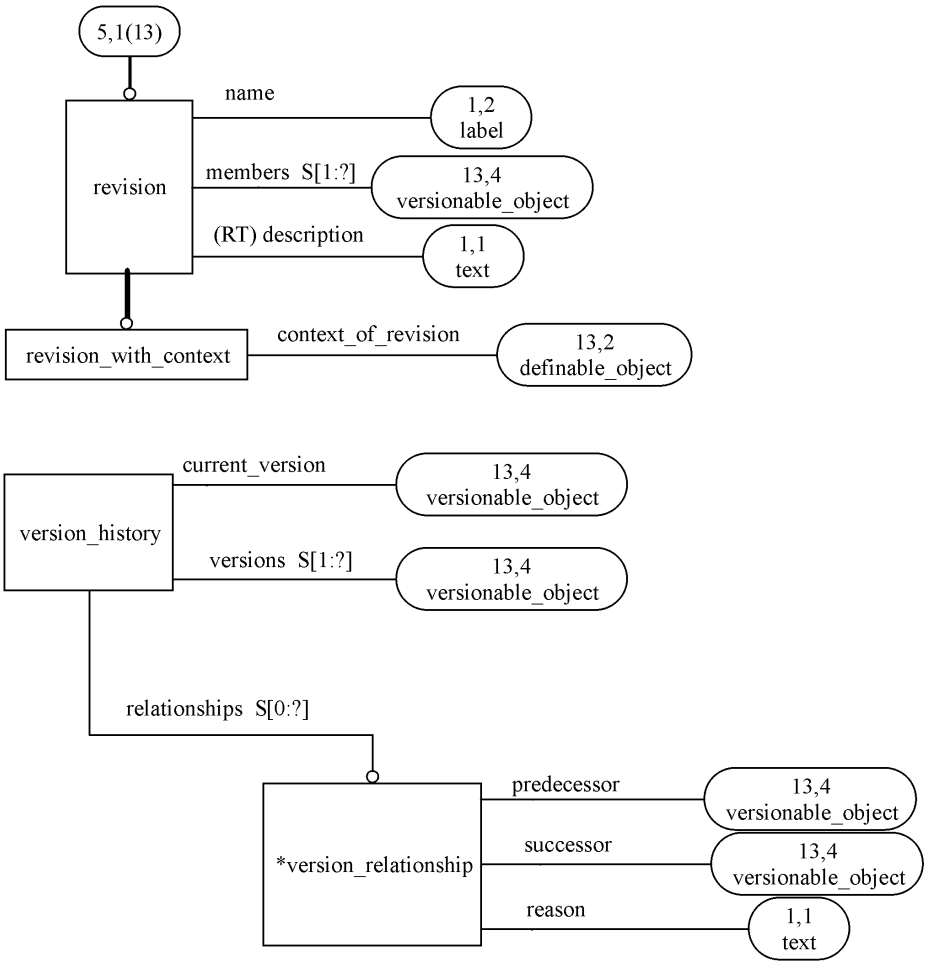


Figure G.5 — ARM EXPRESS-G diagram 5 of 49

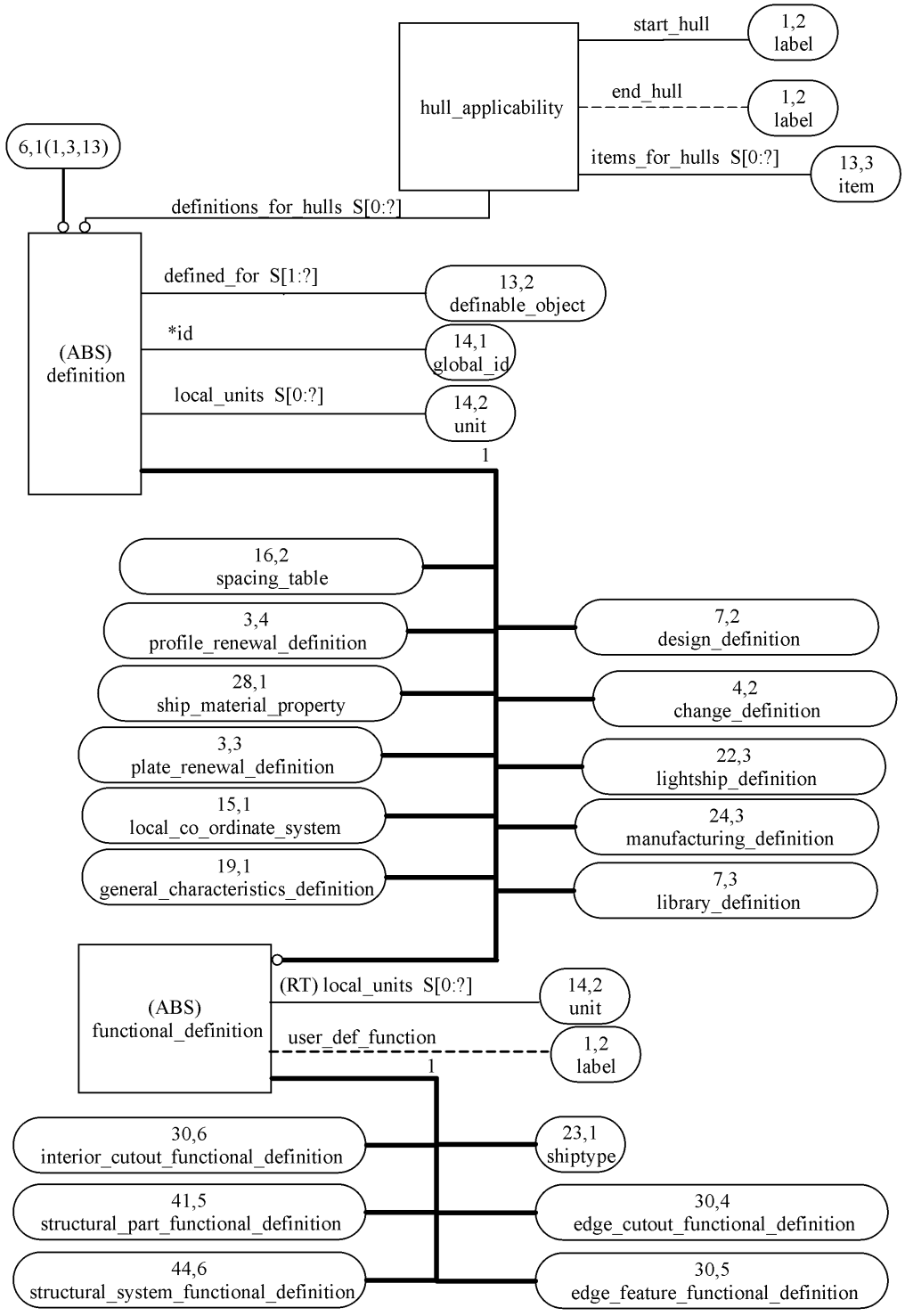


Figure G.6 — ARM EXPRESS-G diagram 6 of 49

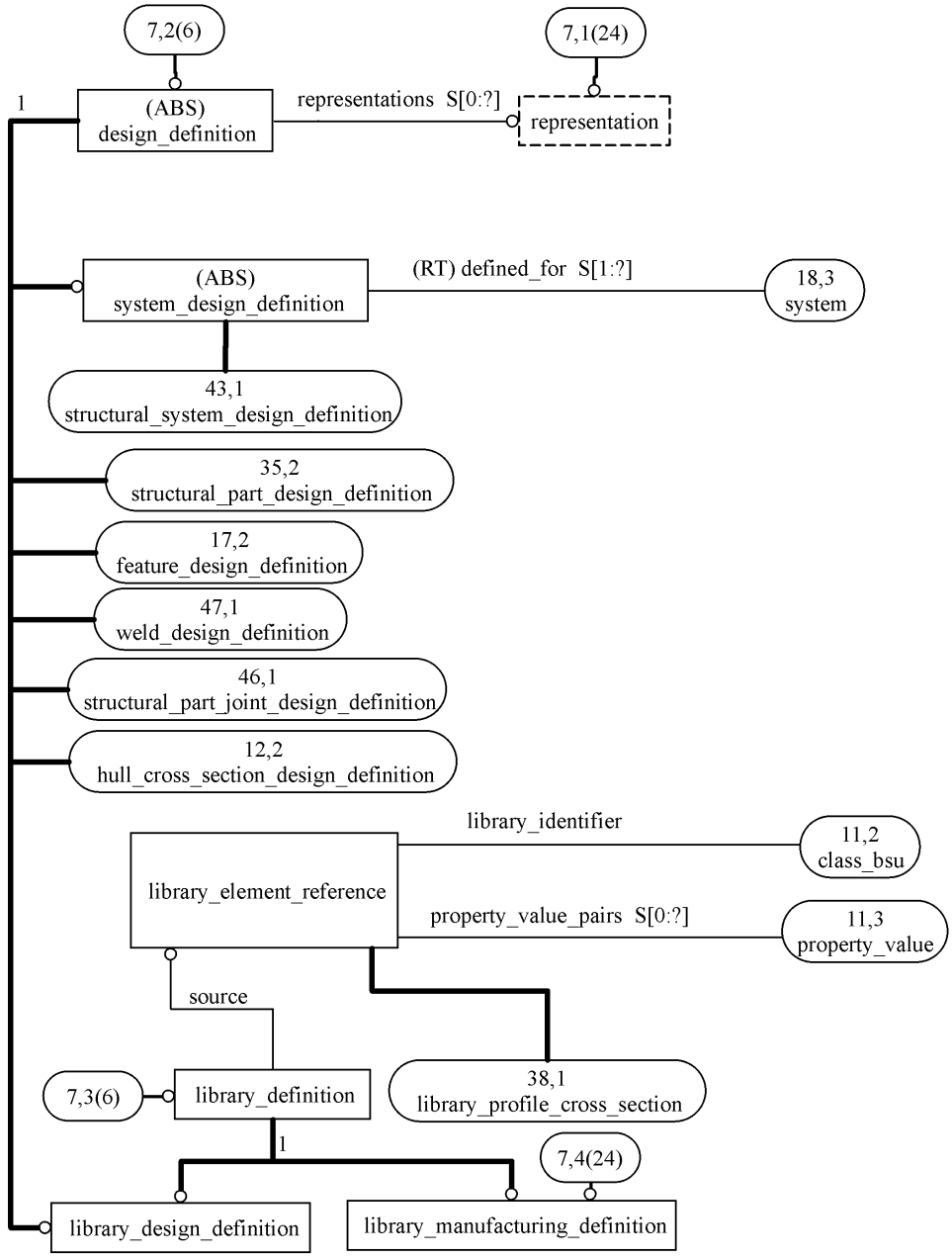


Figure G.7 — ARM EXPRESS-G diagram 7 of 49

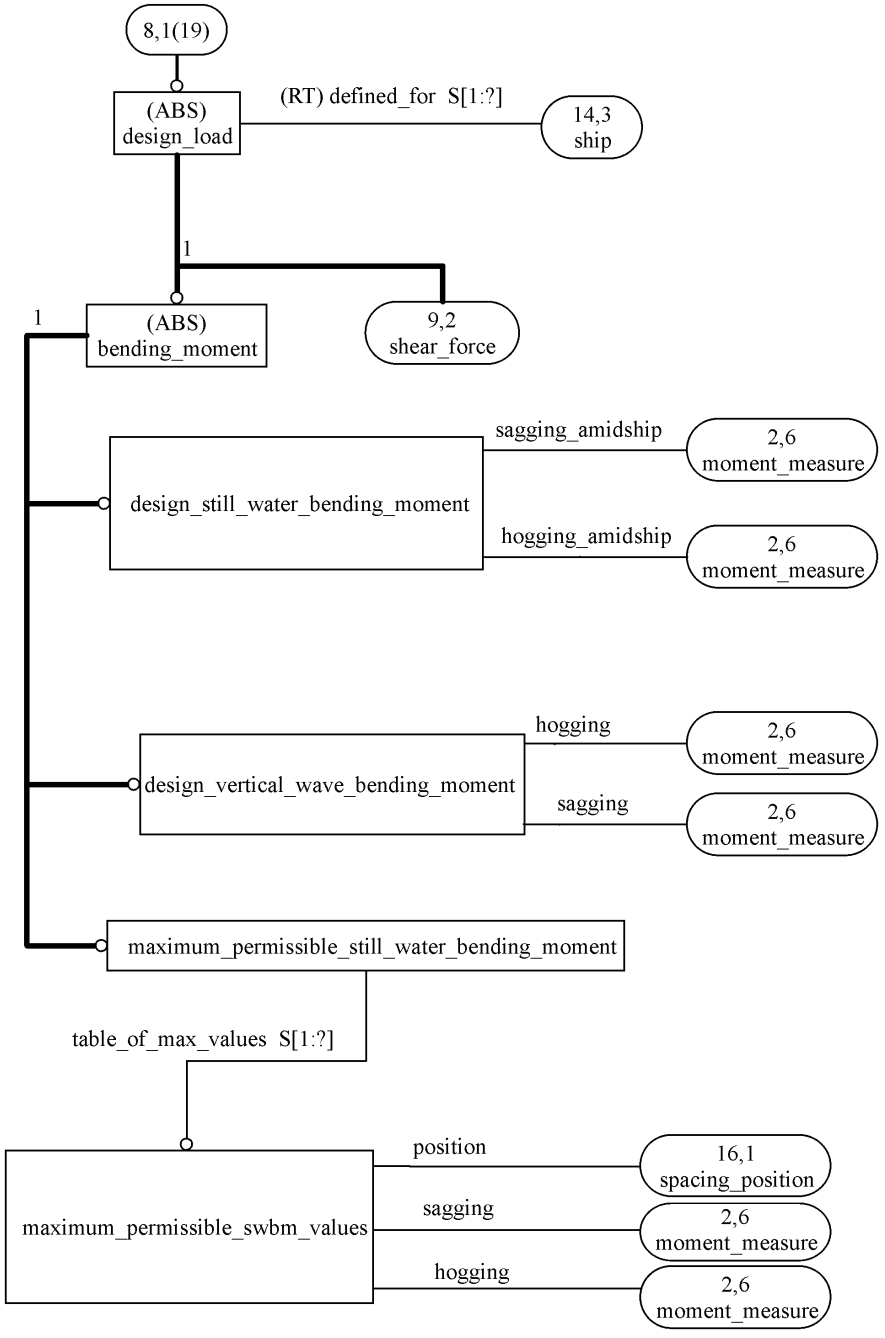


Figure G.8 — ARM EXPRESS-G diagram 8 of 49

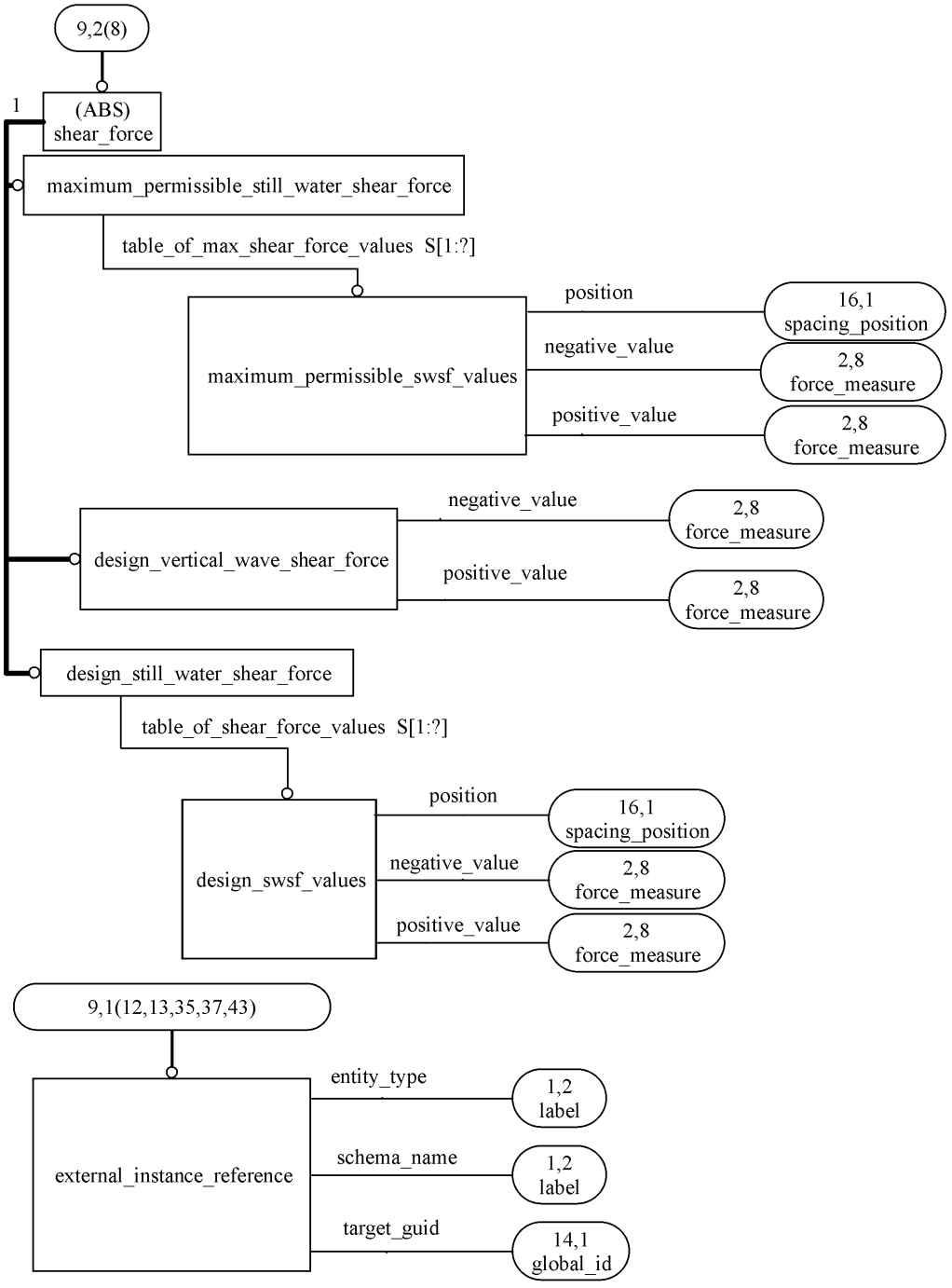


Figure G.9 — ARM EXPRESS-G diagram 9 of 49

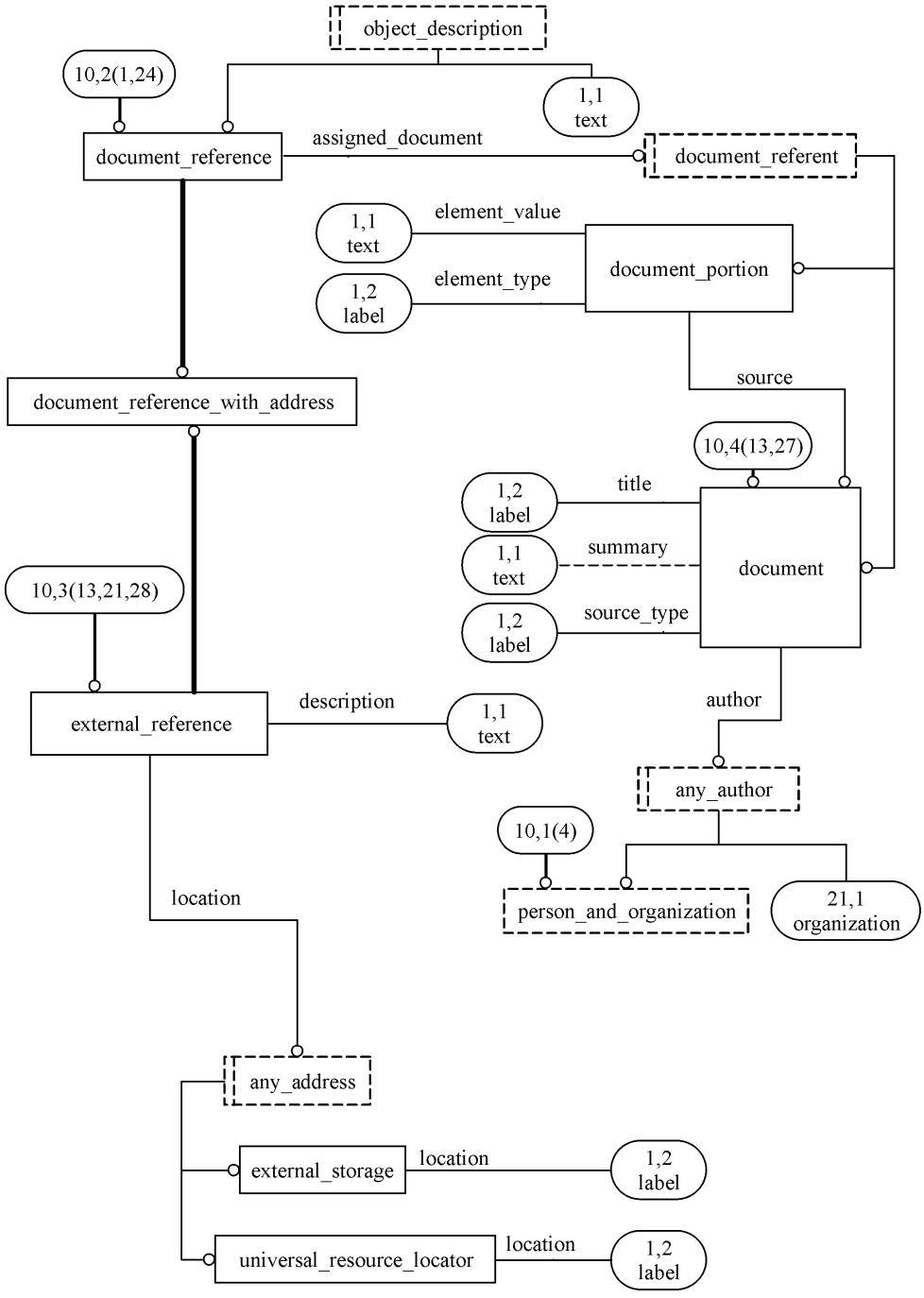


Figure G.10 — ARM EXPRESS-G diagram 10 of 49

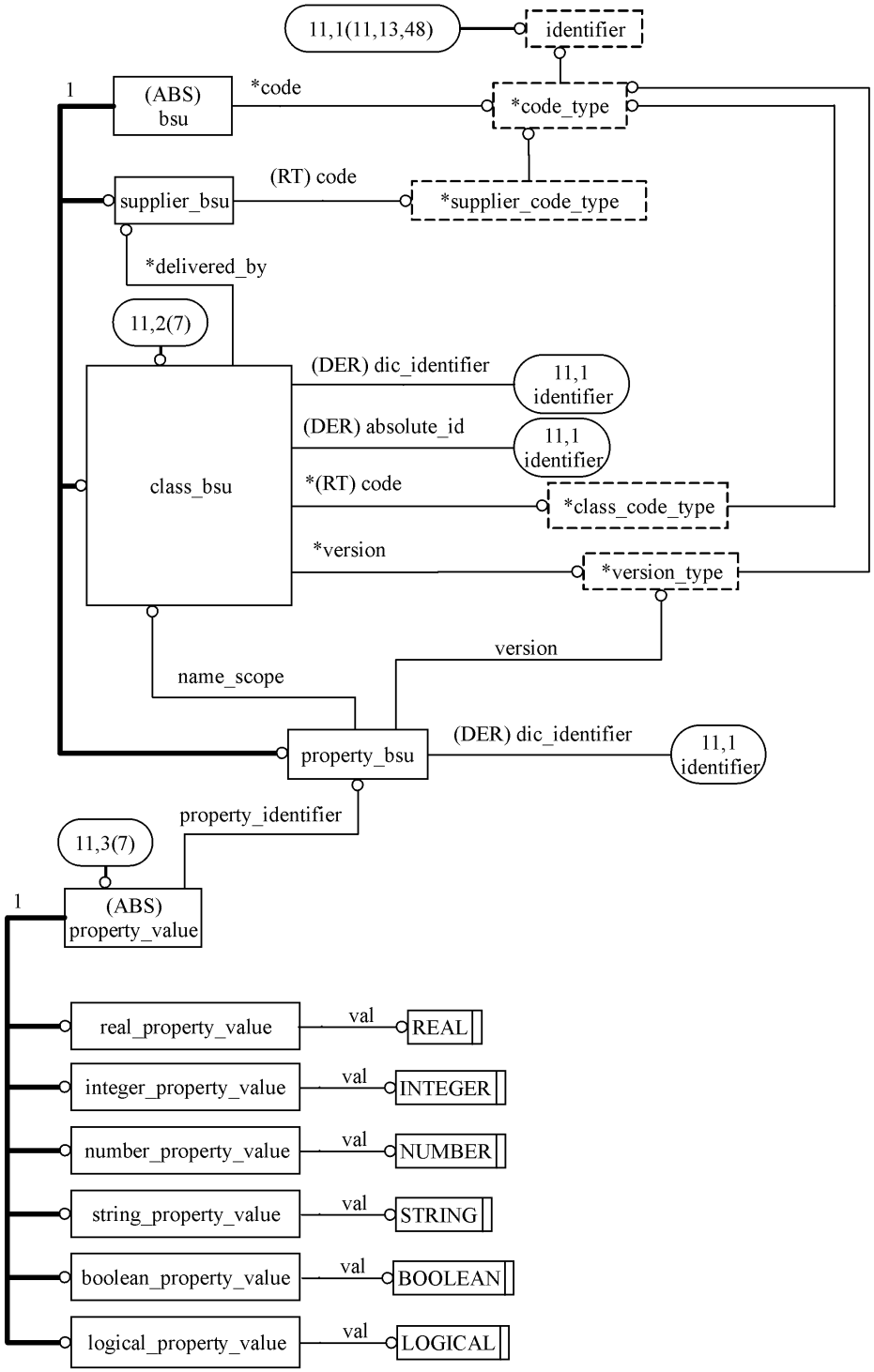


Figure G.11 — ARM EXPRESS-G diagram 11 of 49

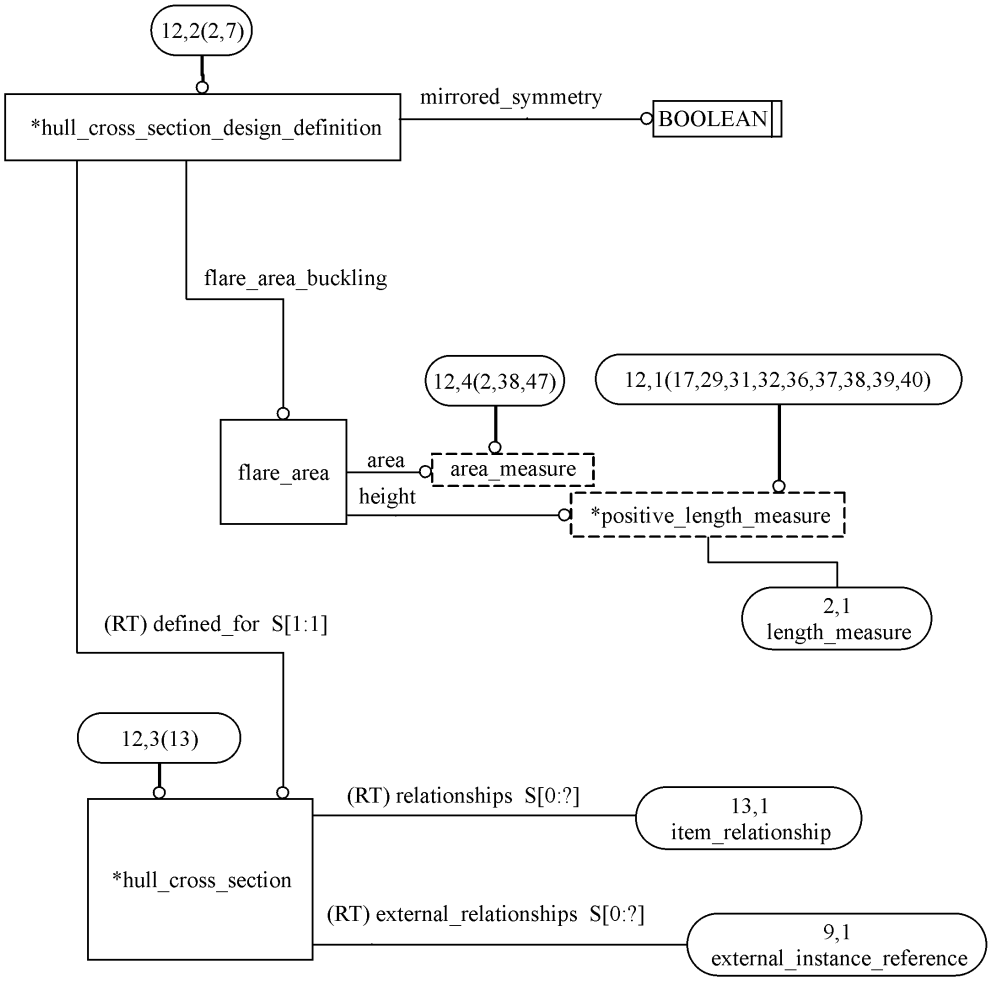


Figure G.12 — ARM EXPRESS-G diagram 12 of 49

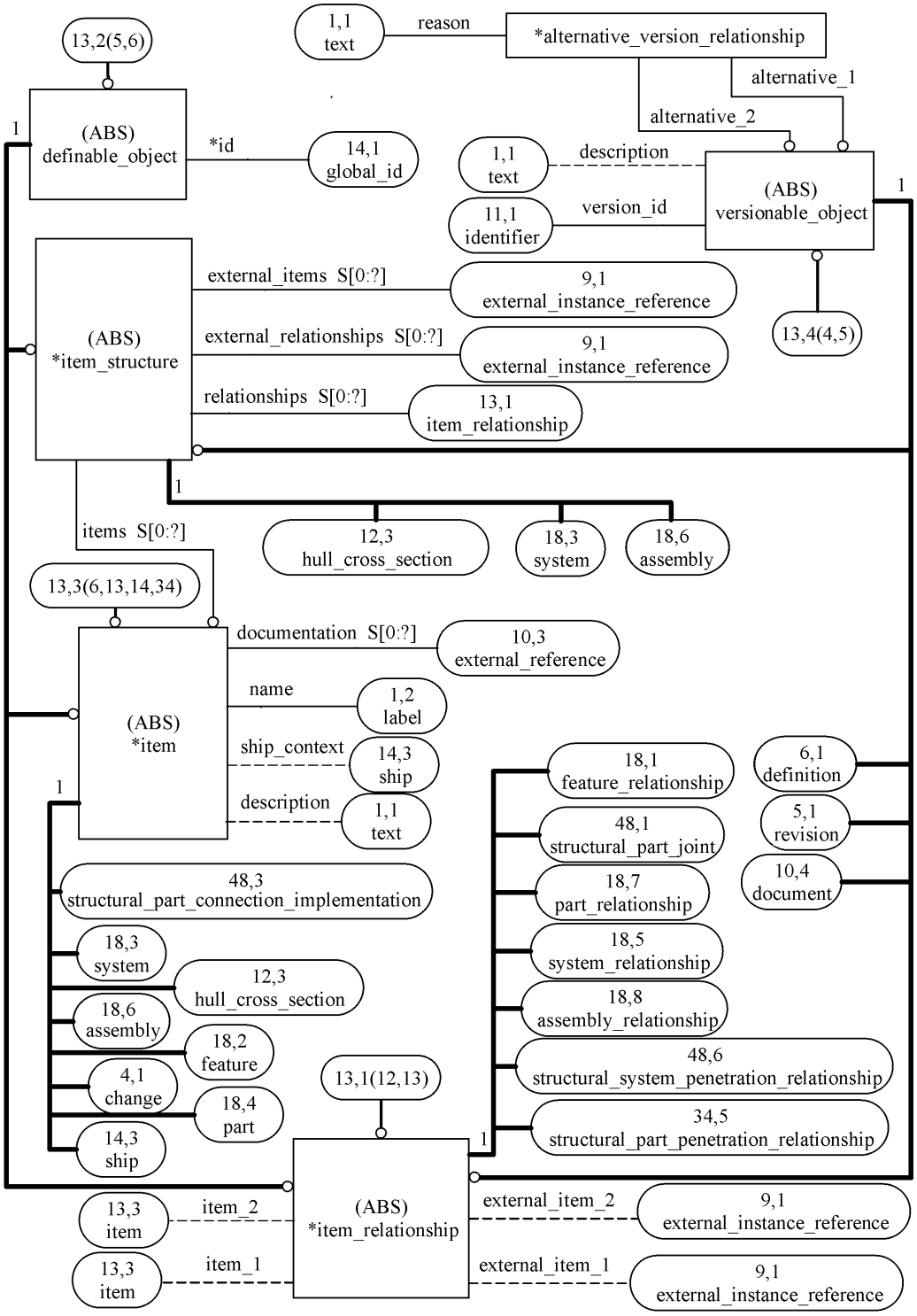


Figure G.13 — ARM EXPRESS-G diagram 13 of 49

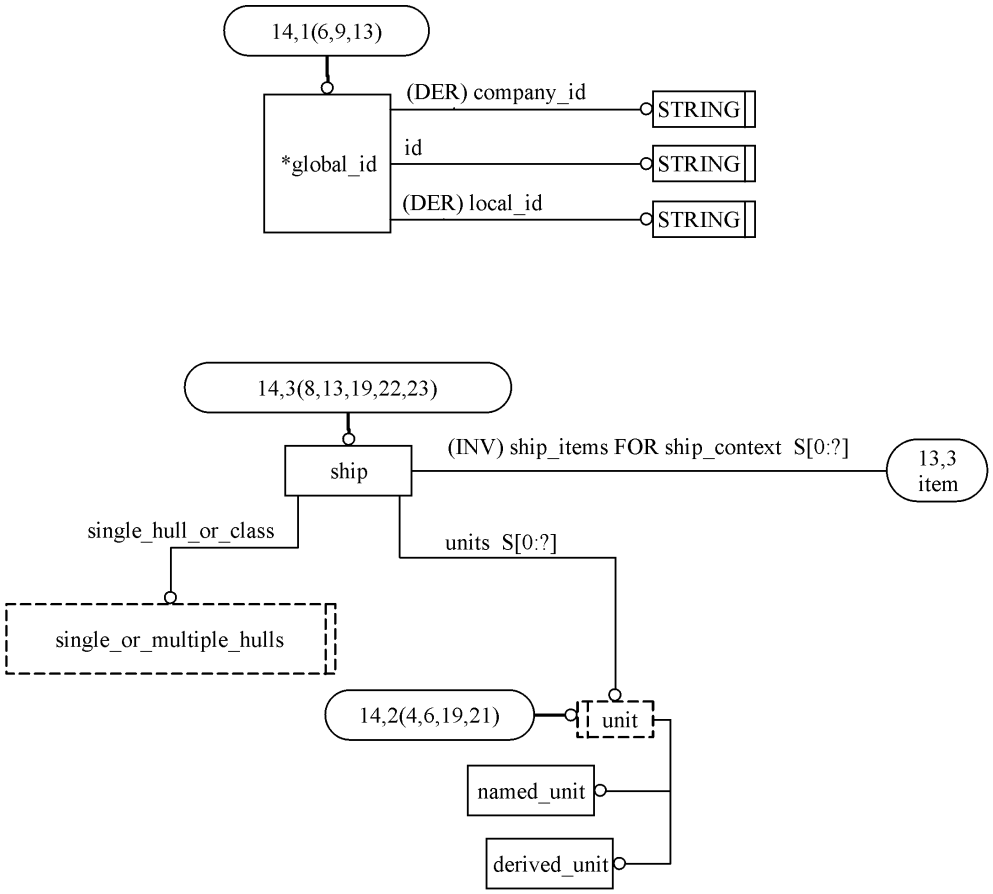


Figure G.14 — ARM EXPRESS-G diagram 14 of 49

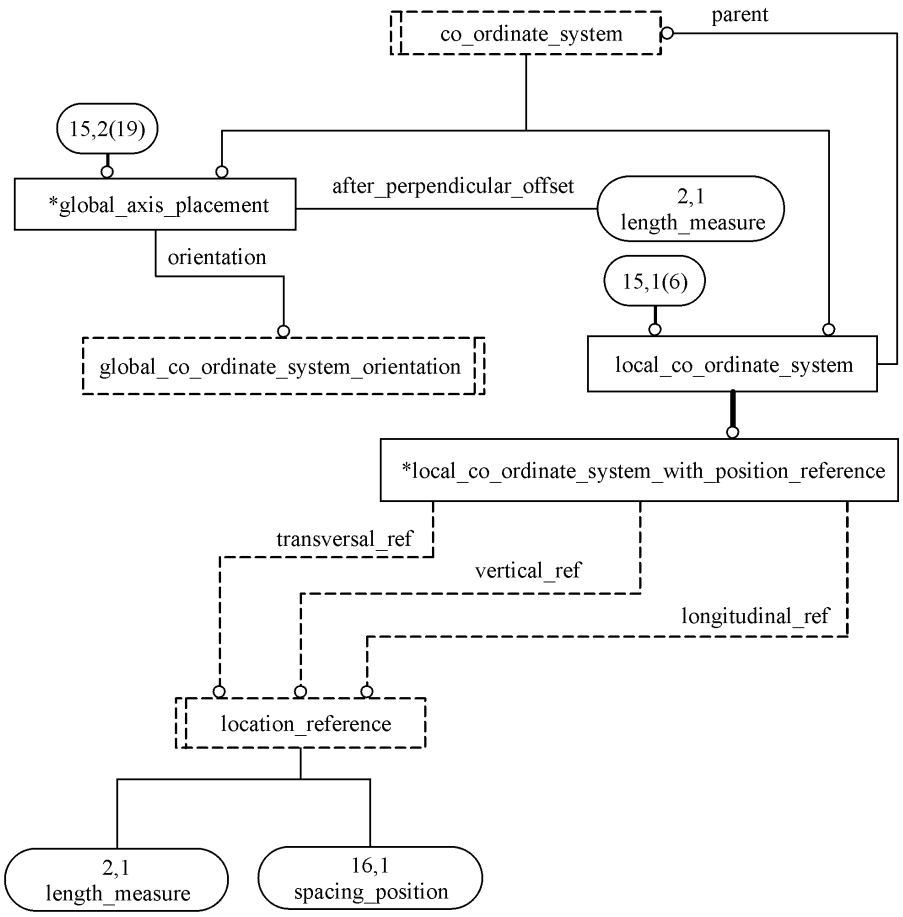


Figure G.15 — ARM EXPRESS-G diagram 15 of 49

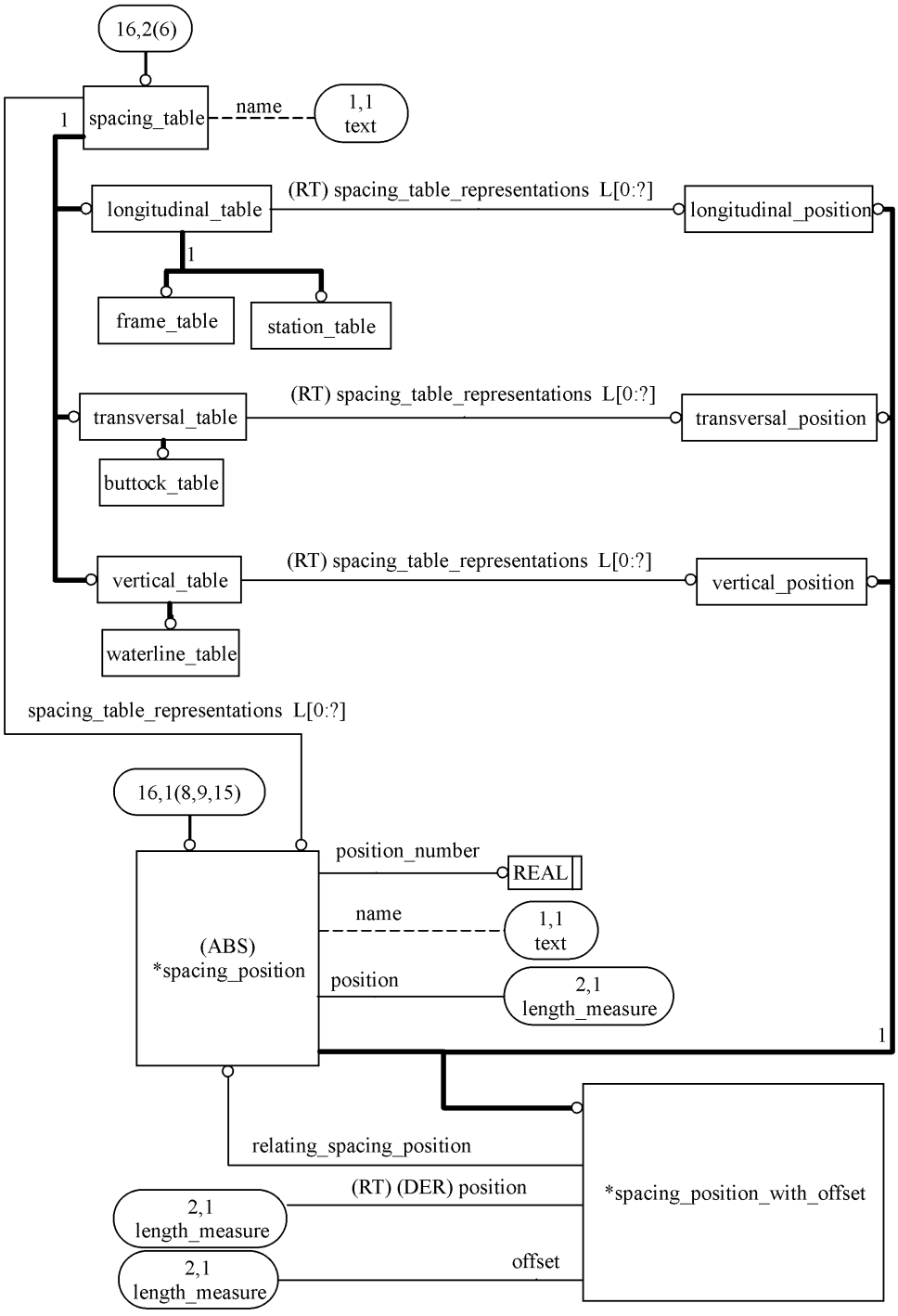


Figure G.16 — ARM EXPRESS-G diagram 16 of 49

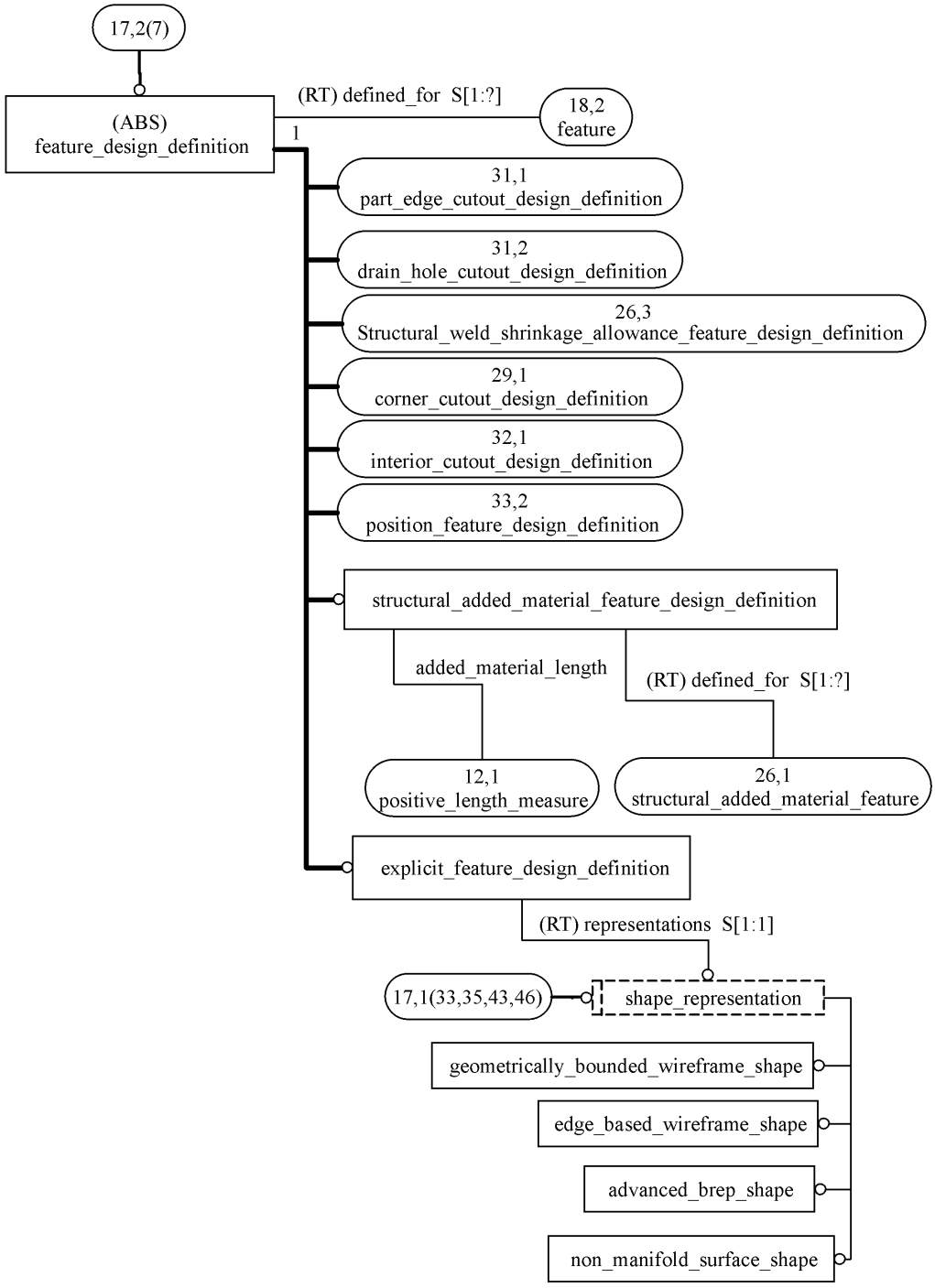


Figure G.17 — ARM EXPRESS-G diagram 17 of 49

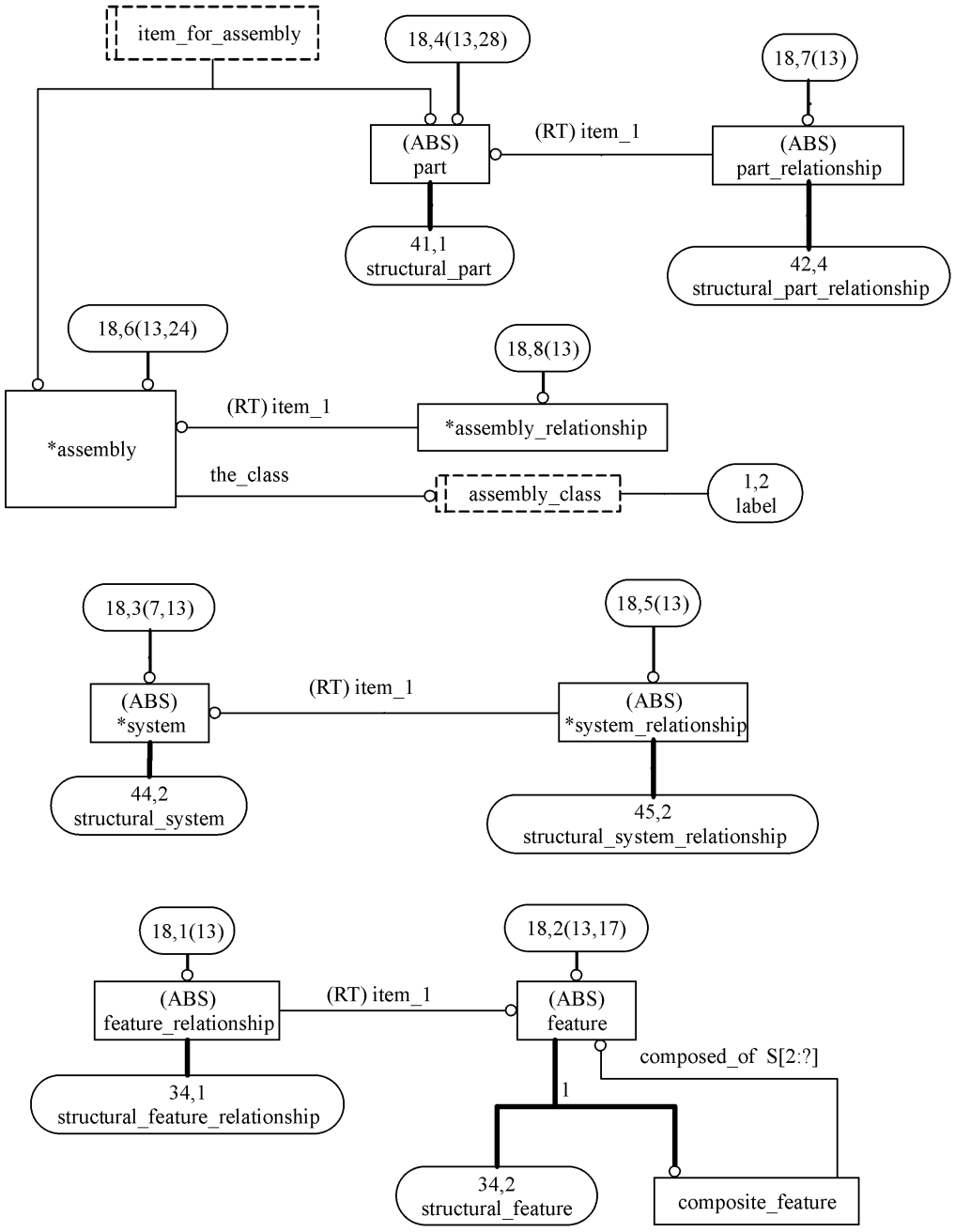


Figure G.18 — ARM EXPRESS-G diagram 18 of 49

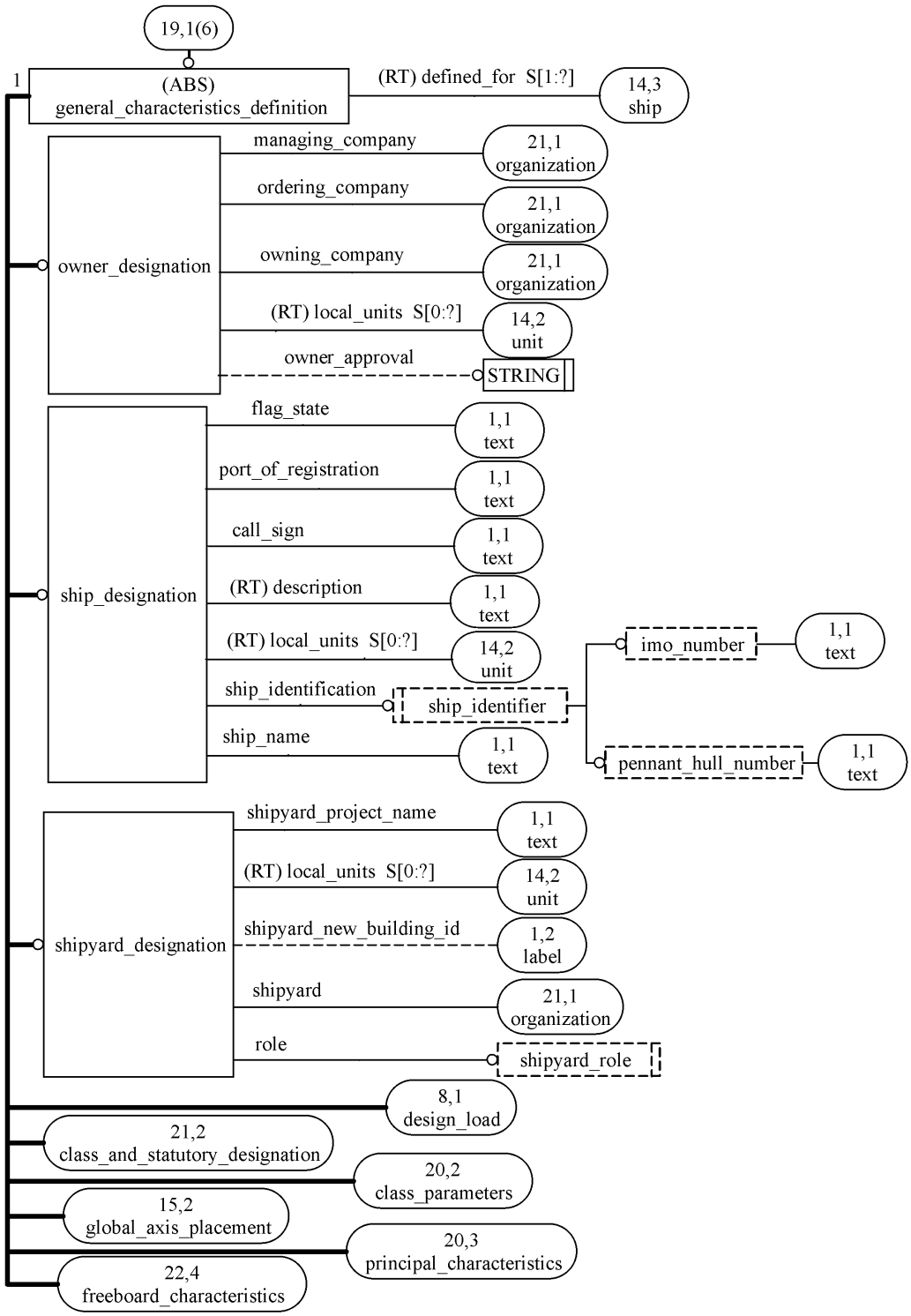


Figure G.19 — ARM EXPRESS-G diagram 19 of 49

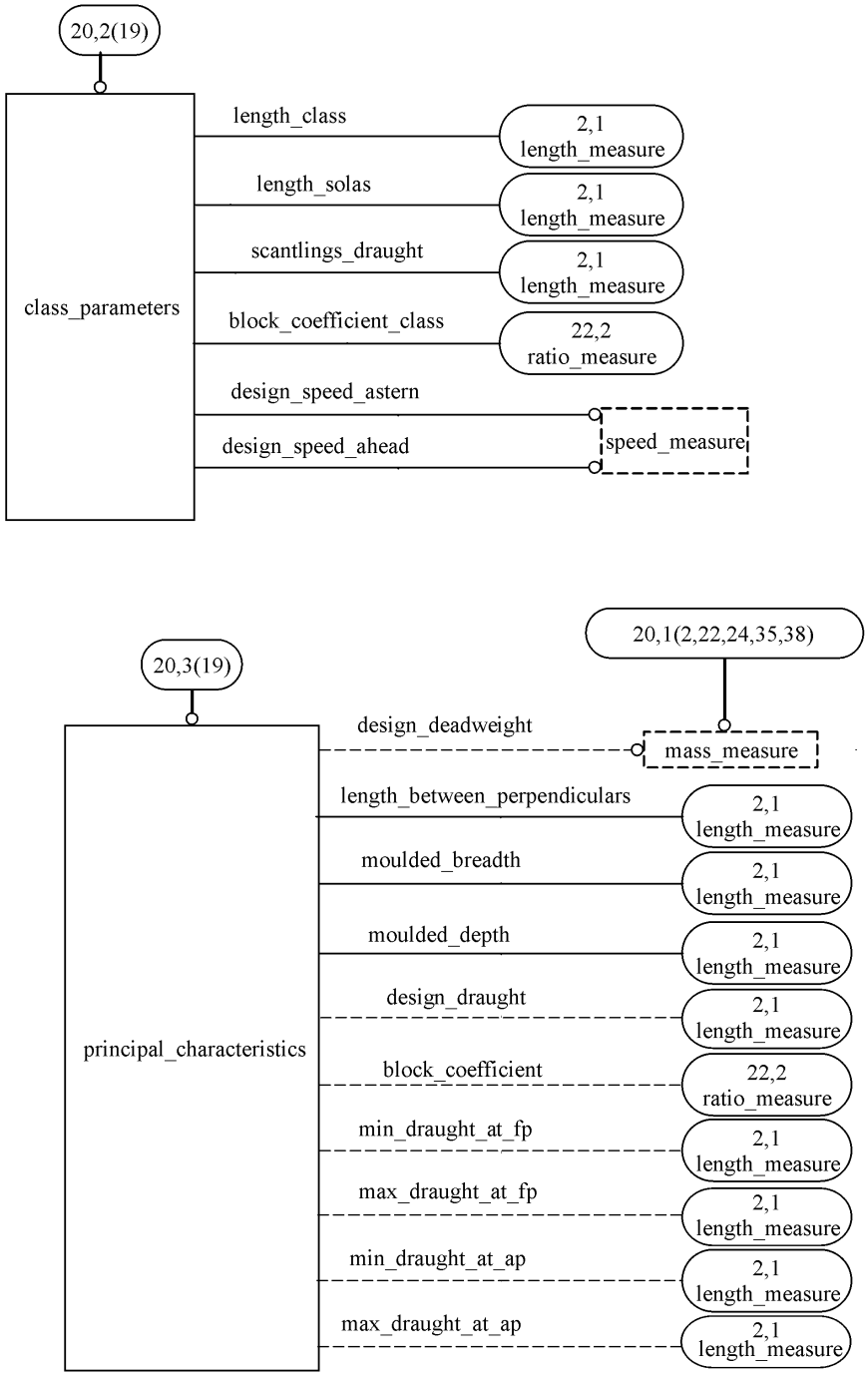


Figure G.20 — ARM EXPRESS-G diagram 20 of 49

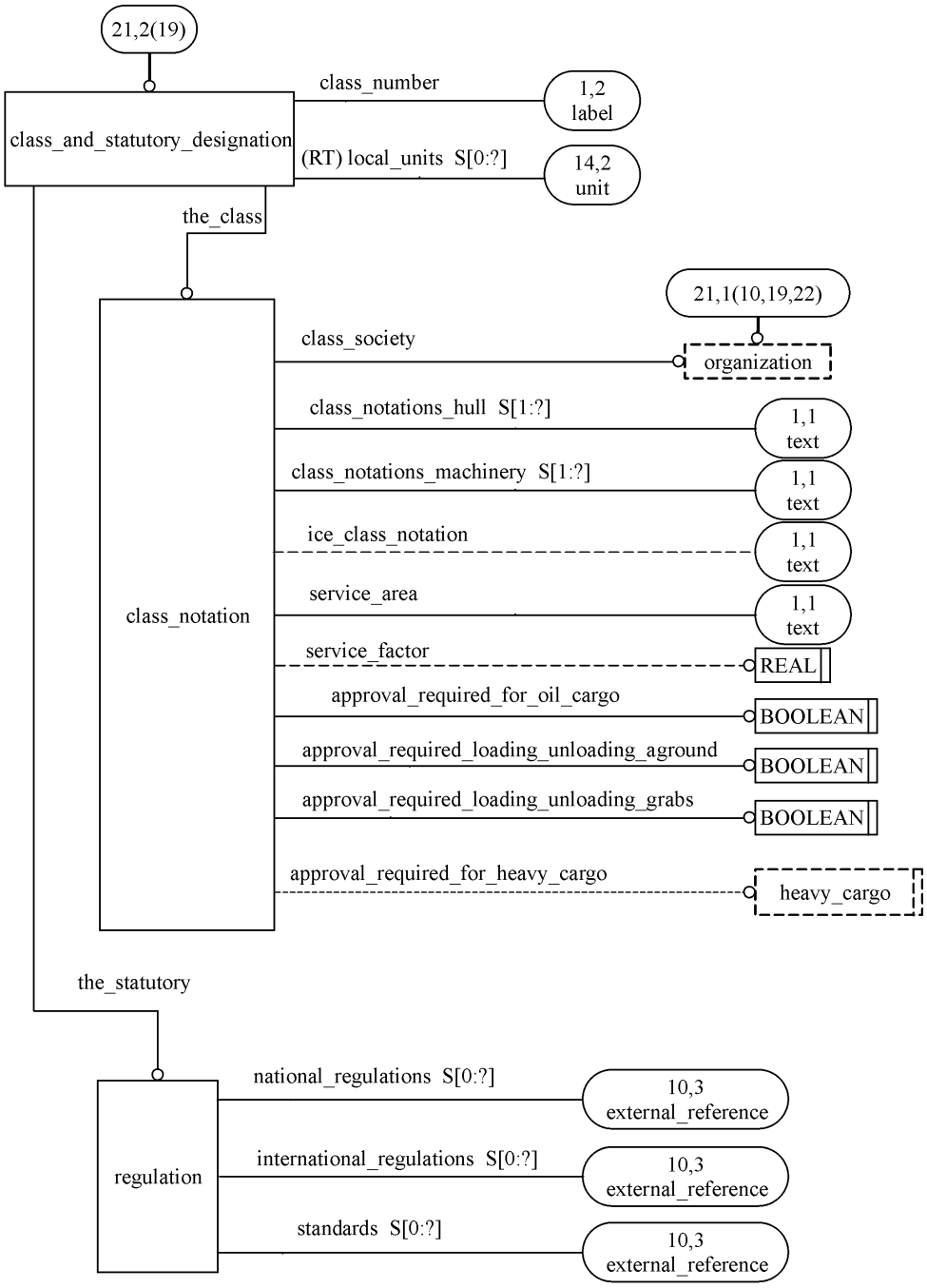


Figure G.21 — ARM EXPRESS-G diagram 21 of 49

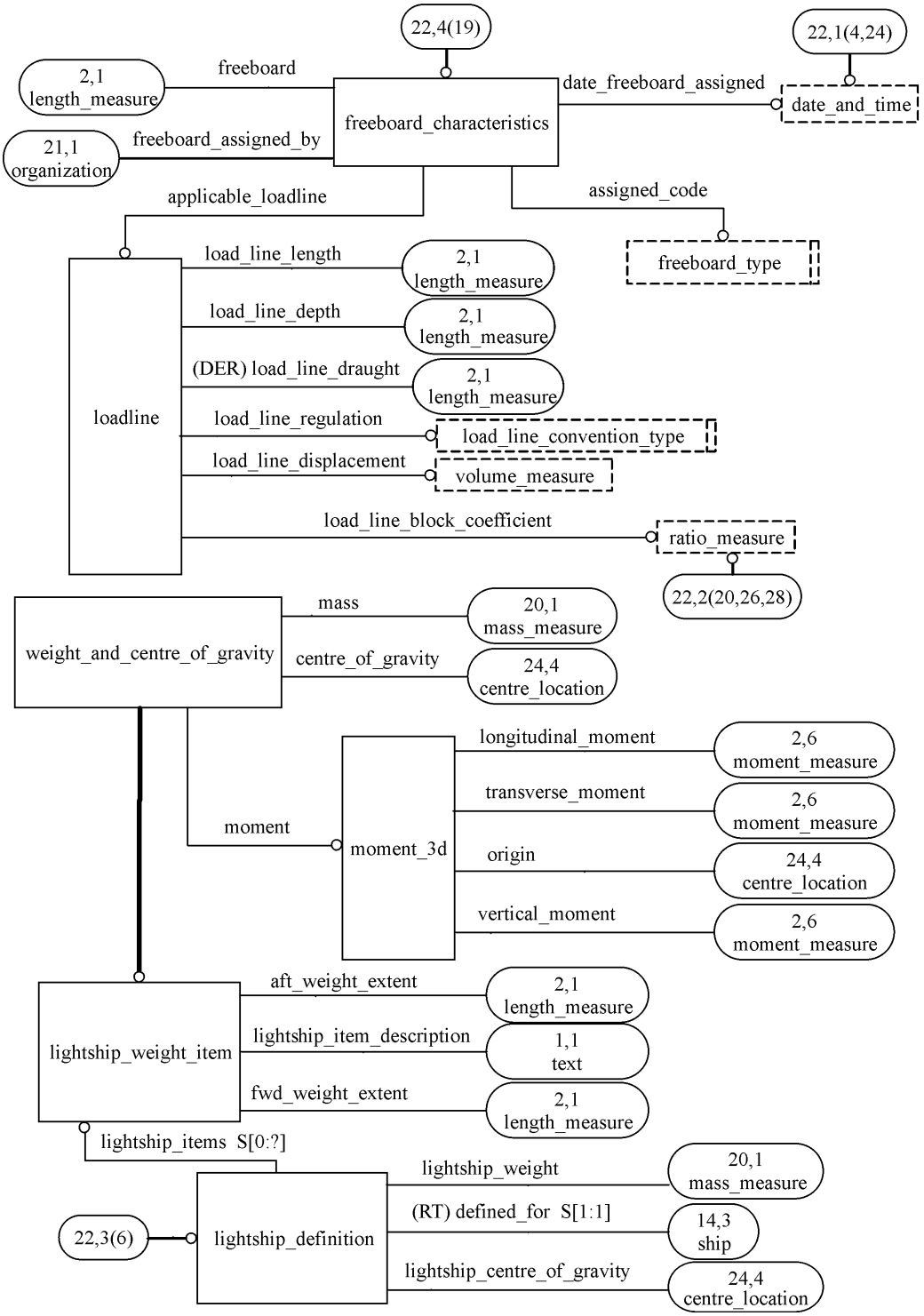


Figure G.22 — ARM EXPRESS-G diagram 22 of 49

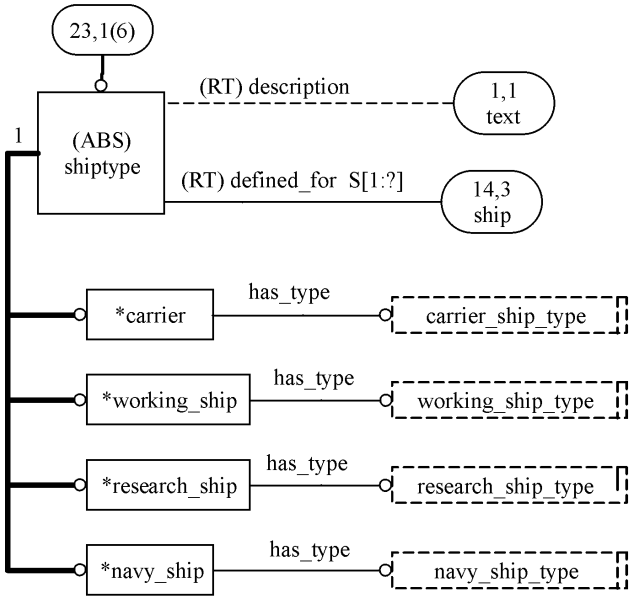


Figure G.23 — ARM EXPRESS-G diagram 23 of 49

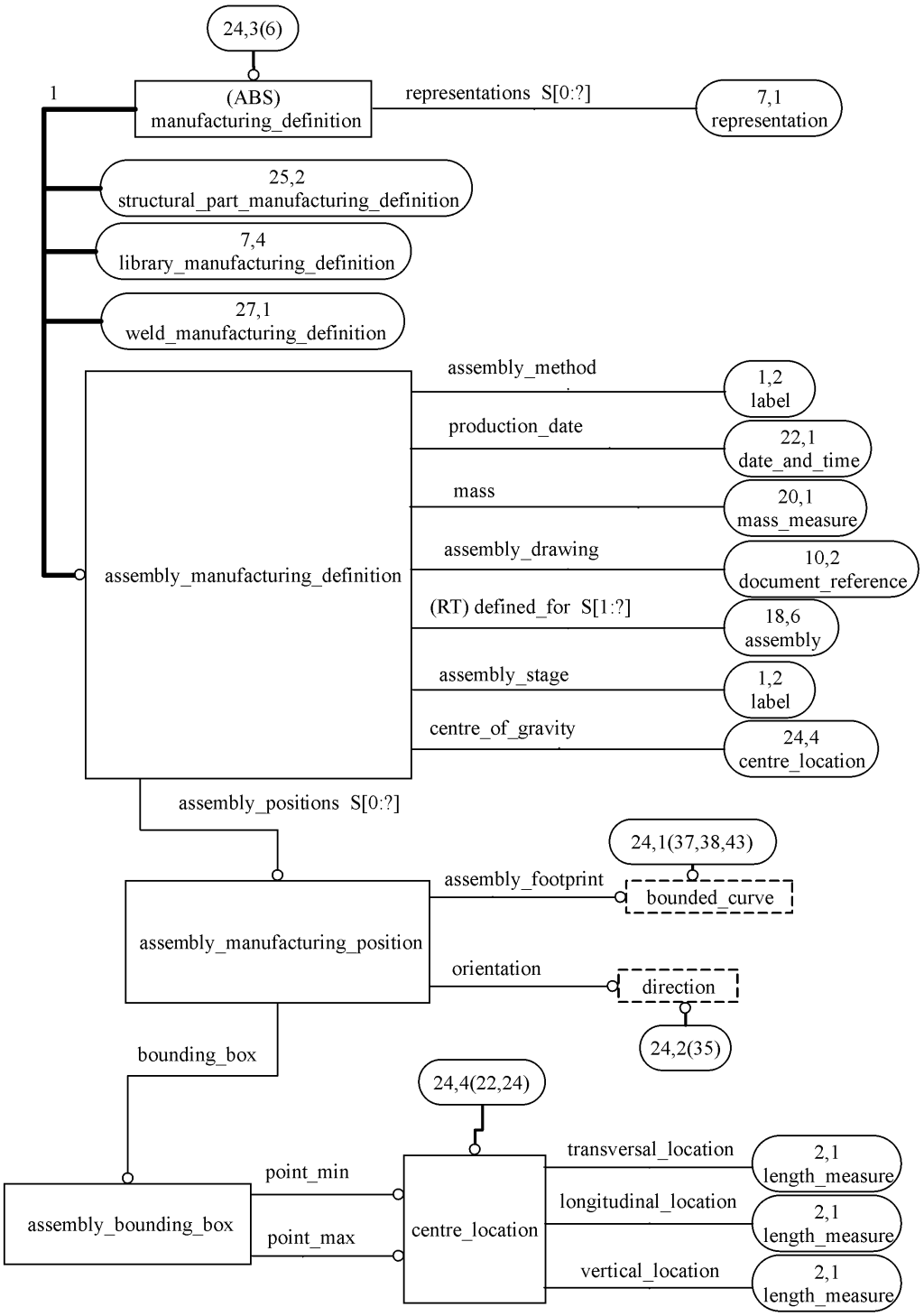


Figure G.24 — ARM EXPRESS-G diagram 24 of 49

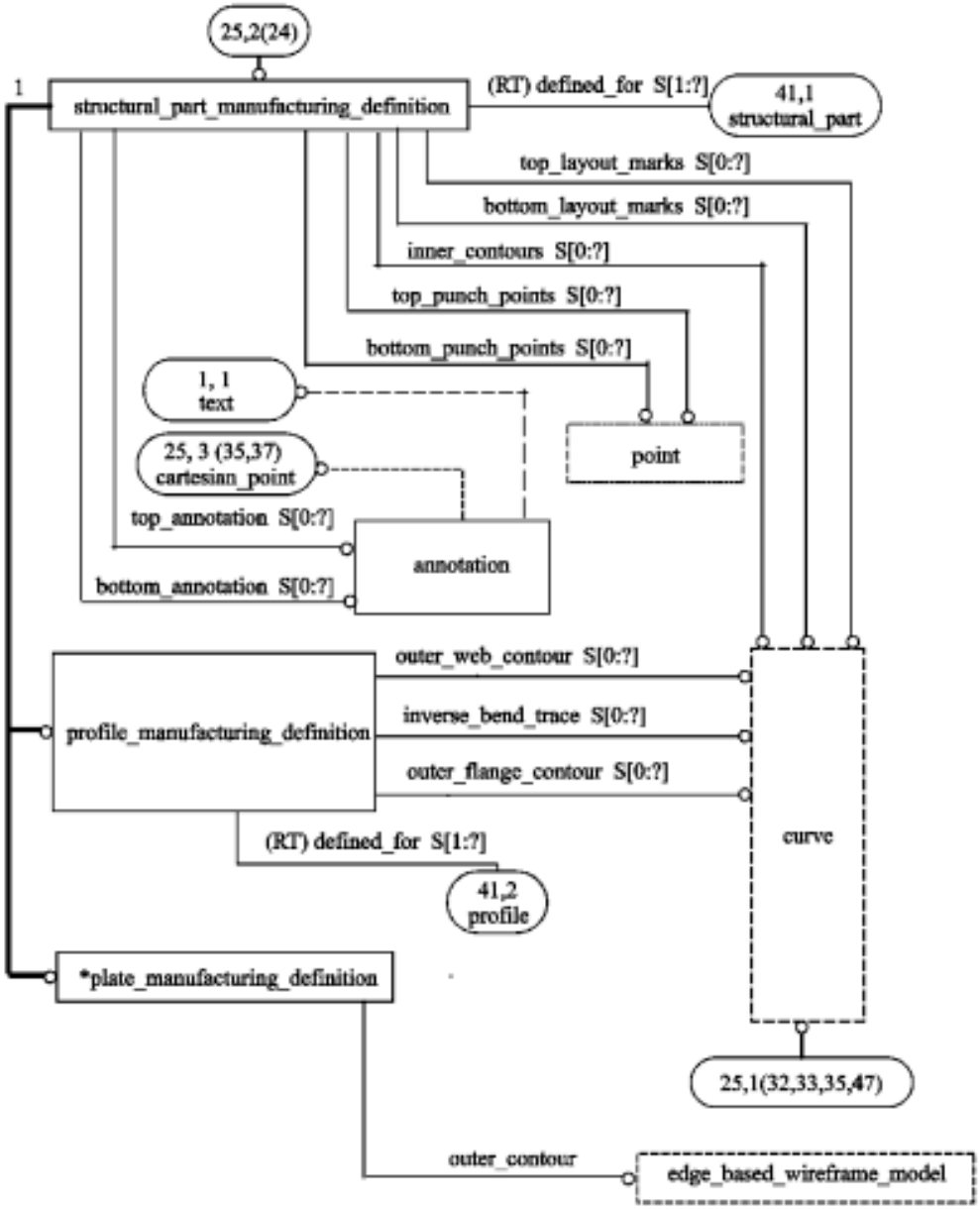


Figure G.25 — ARM EXPRESS-G diagram 25 of 49

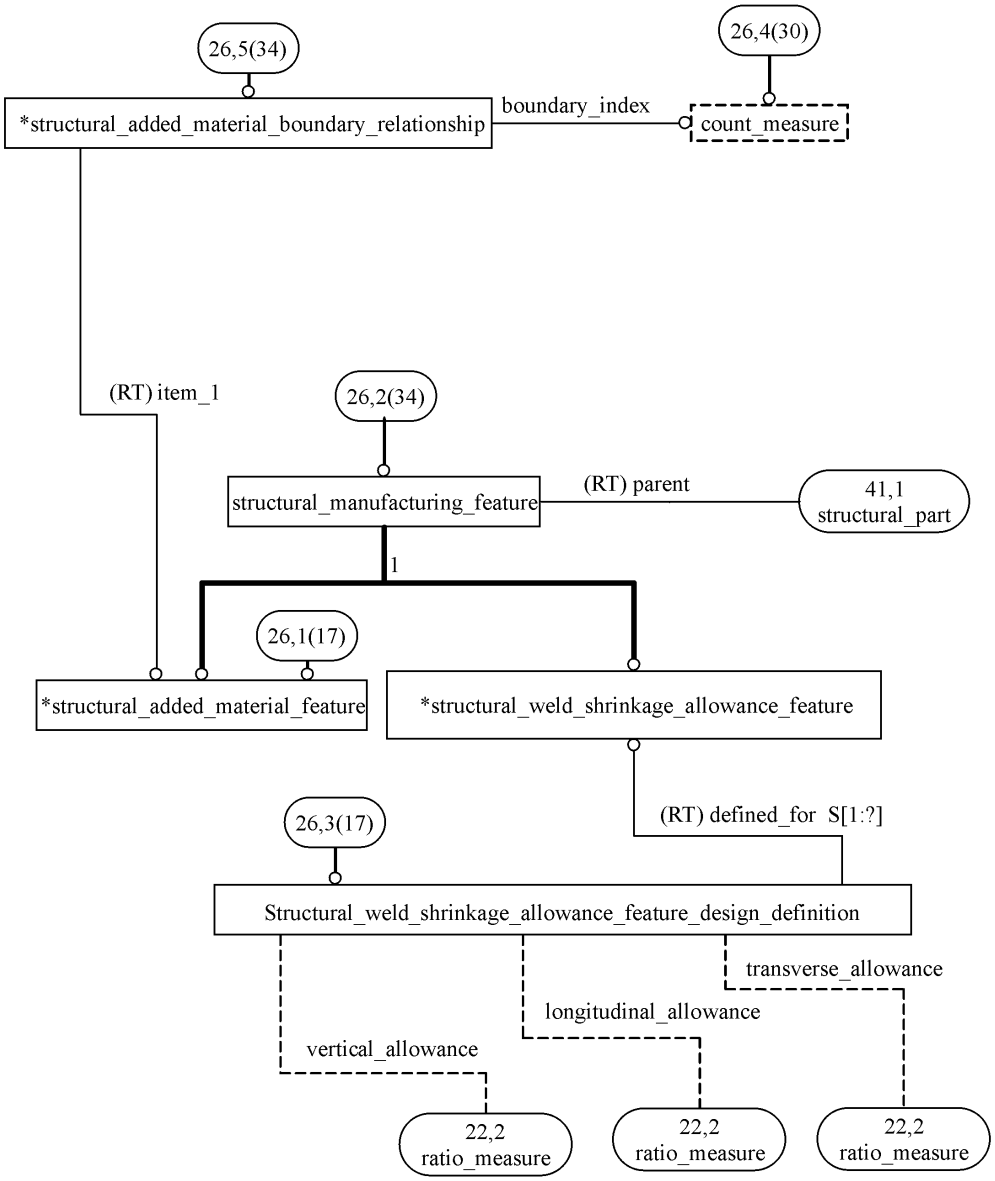


Figure G.26 — ARM EXPRESS-G diagram 26 of 49

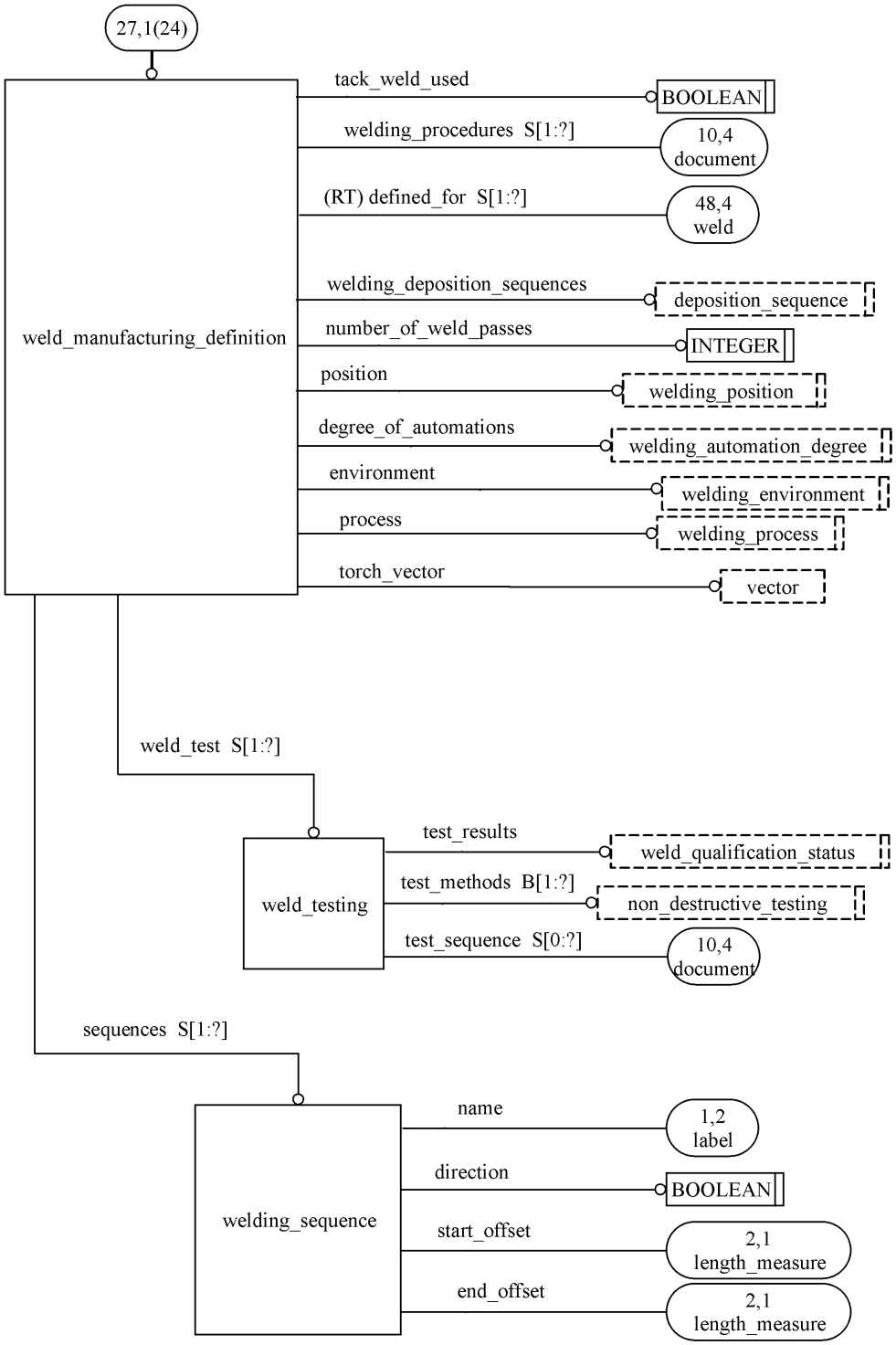


Figure G.27 — ARM EXPRESS-G diagram 27 of 49

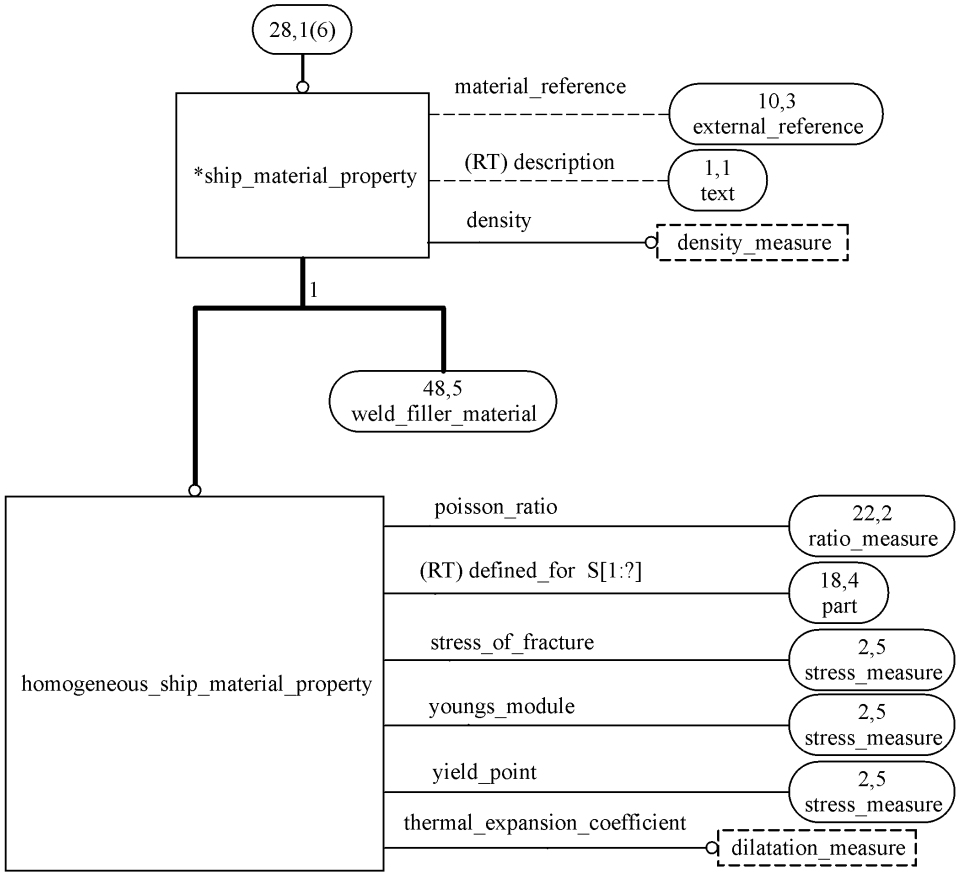


Figure G.28 — ARM EXPRESS-G diagram 28 of 49

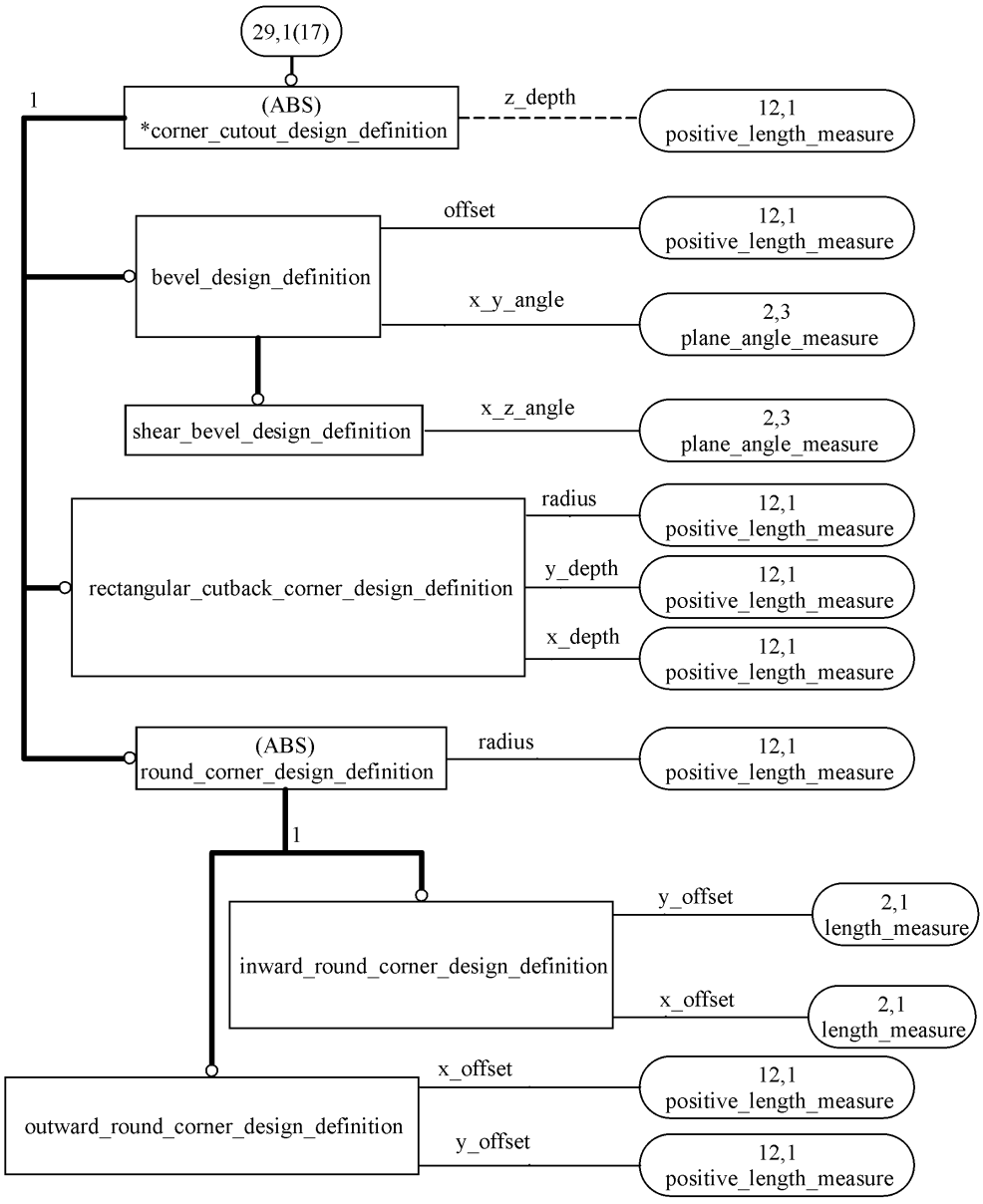


Figure G.29 — ARM EXPRESS-G diagram 29 of 49

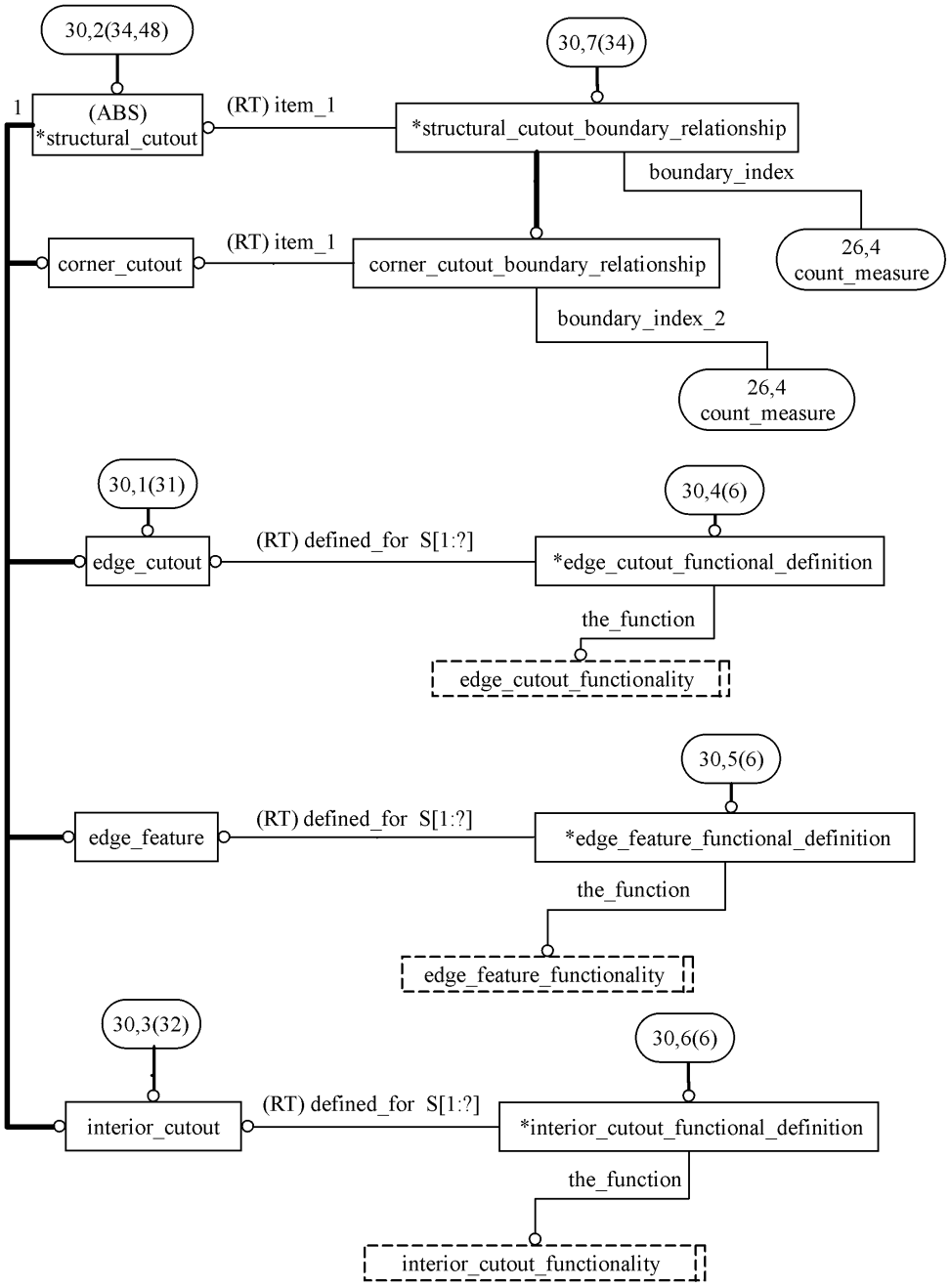


Figure G.30 — ARM EXPRESS-G diagram 30 of 49

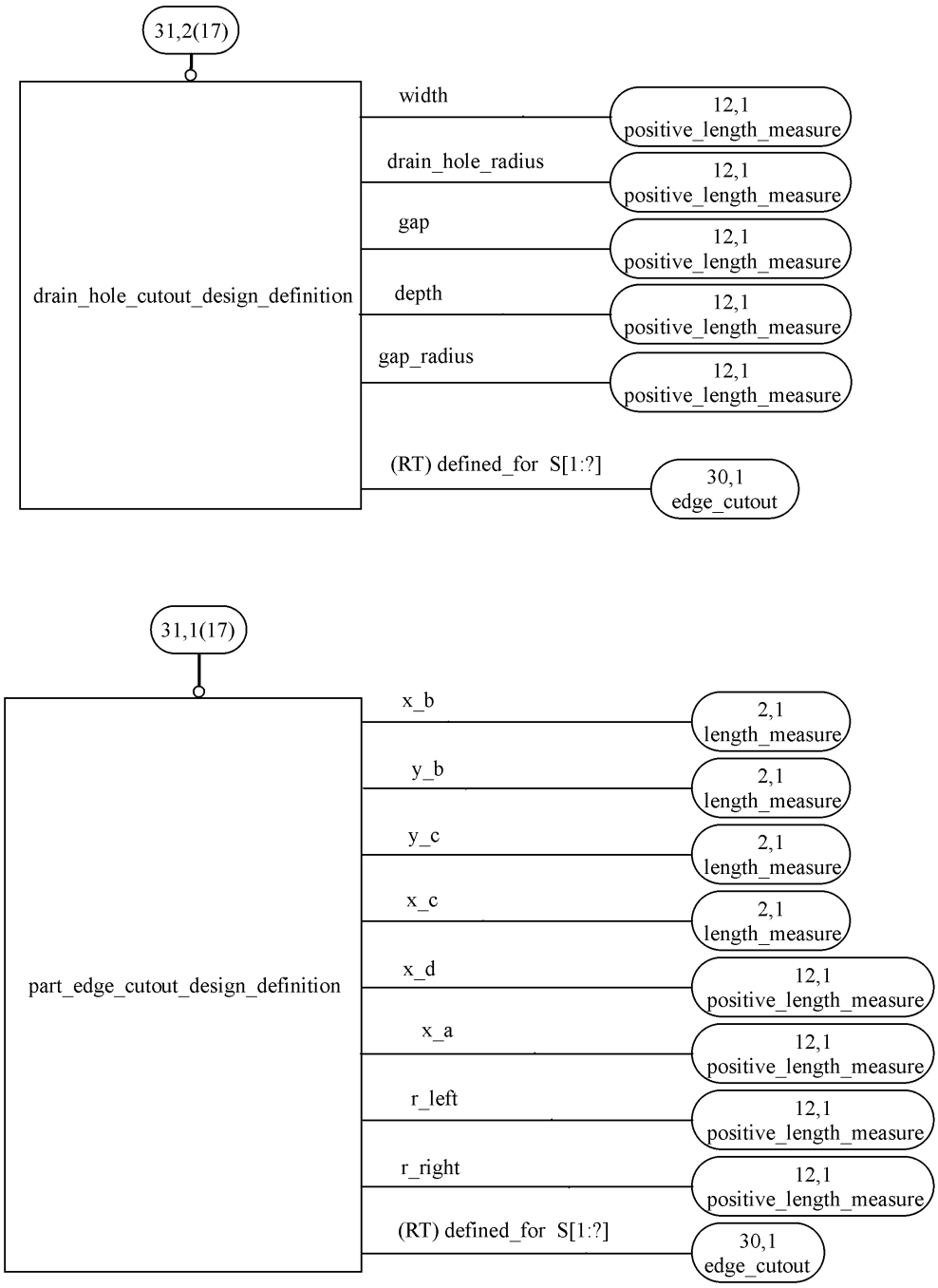


Figure G.31 — ARM EXPRESS-G diagram 31 of 49

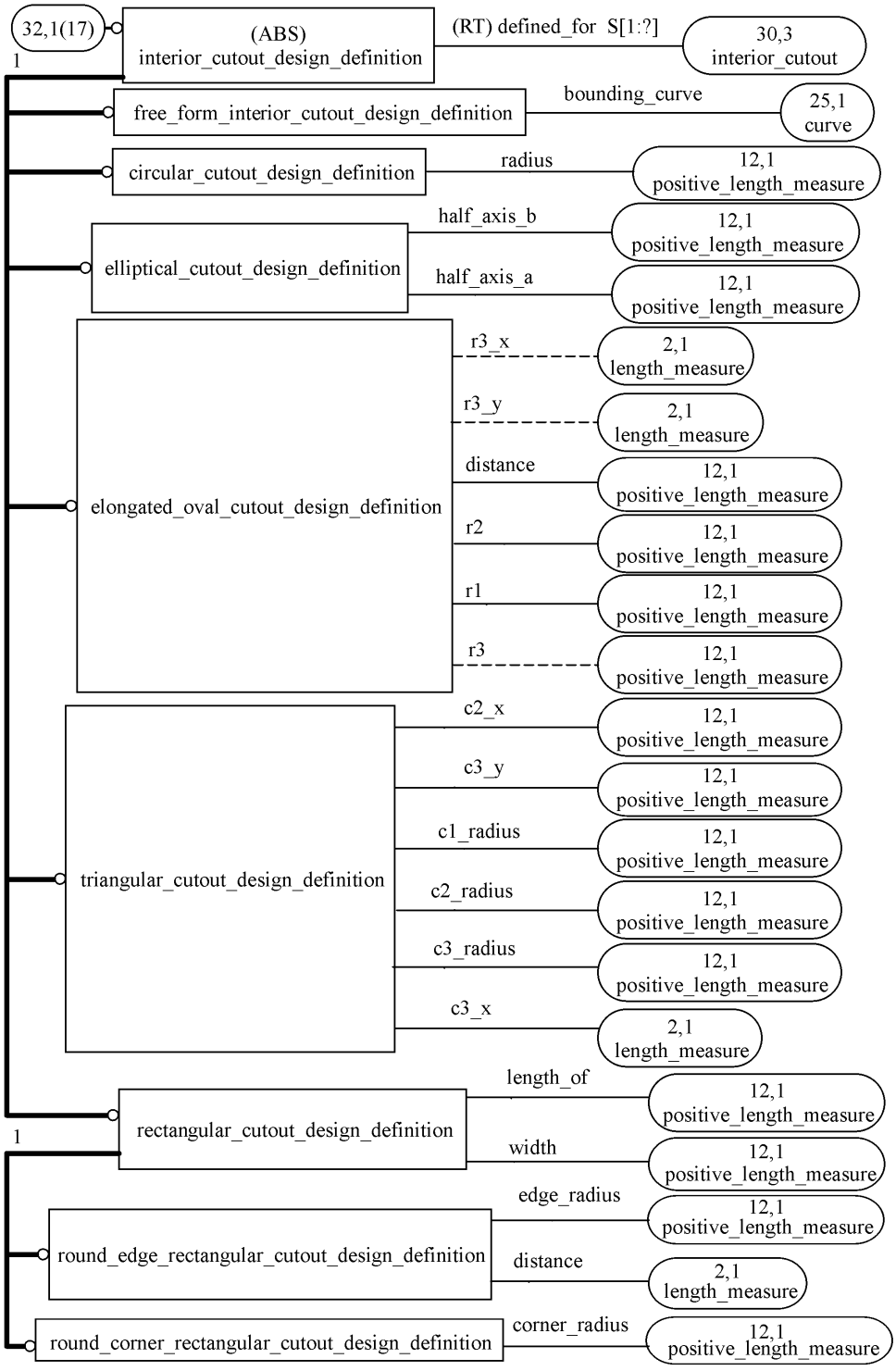


Figure G.32 — ARM EXPRESS-G diagram 32 of 49

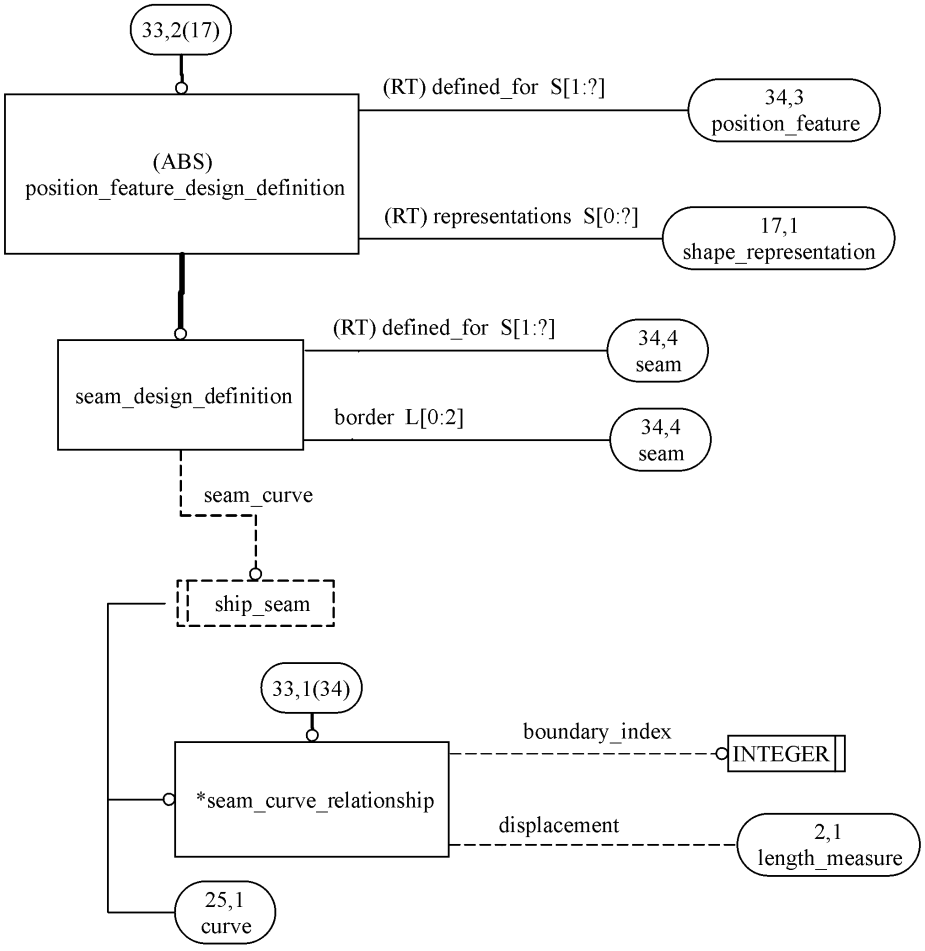


Figure G.33 — ARM EXPRESS-G diagram 33 of 49

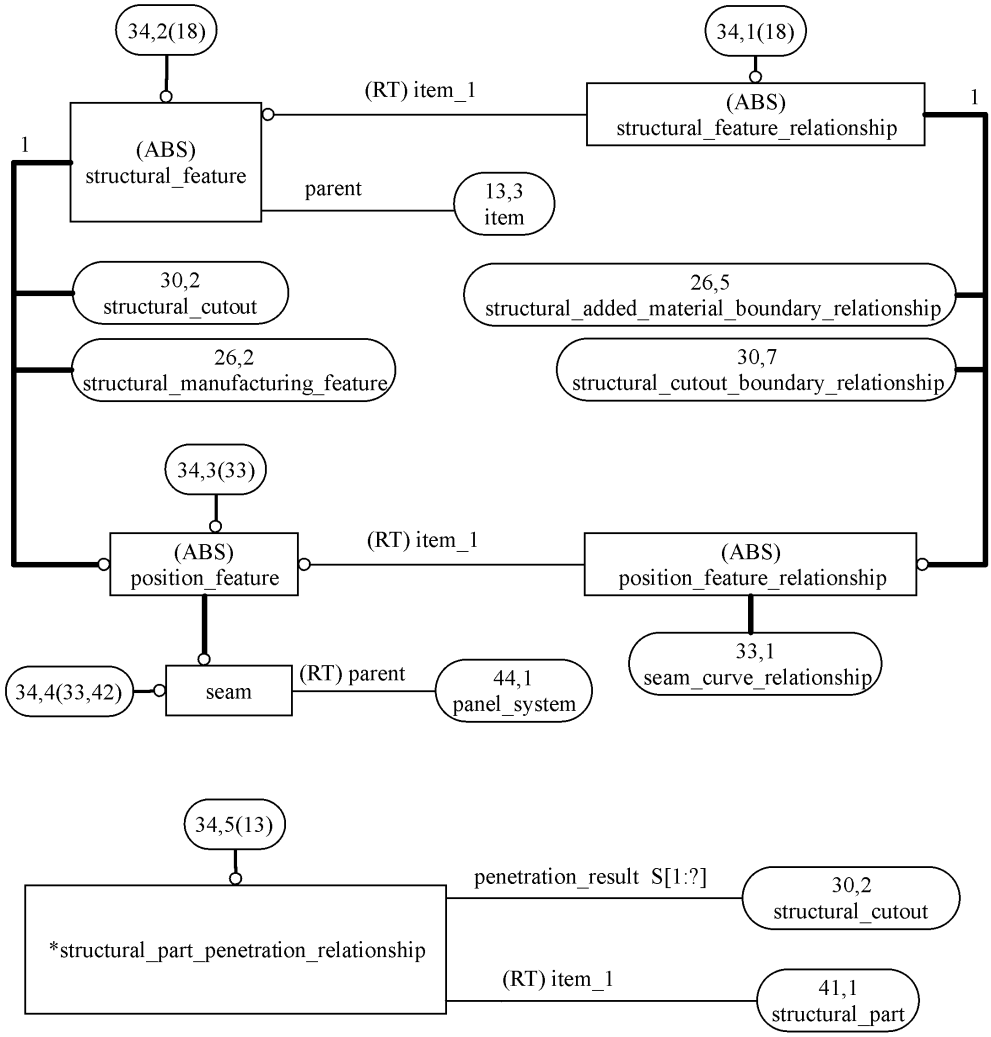


Figure G.34 — ARM EXPRESS-G diagram 34 of 49

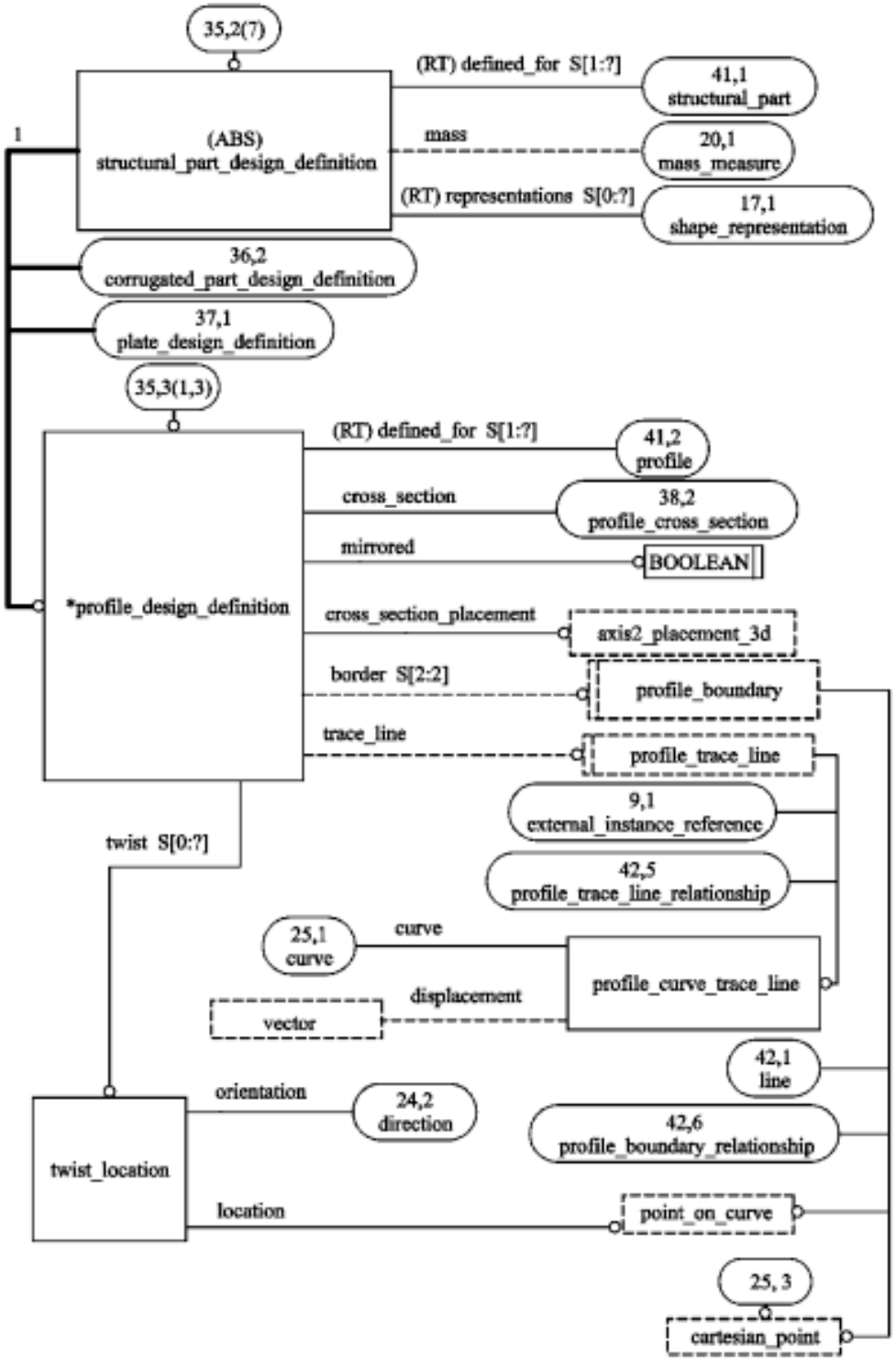


Figure G.35 — ARM EXPRESS-G diagram 35 of 49

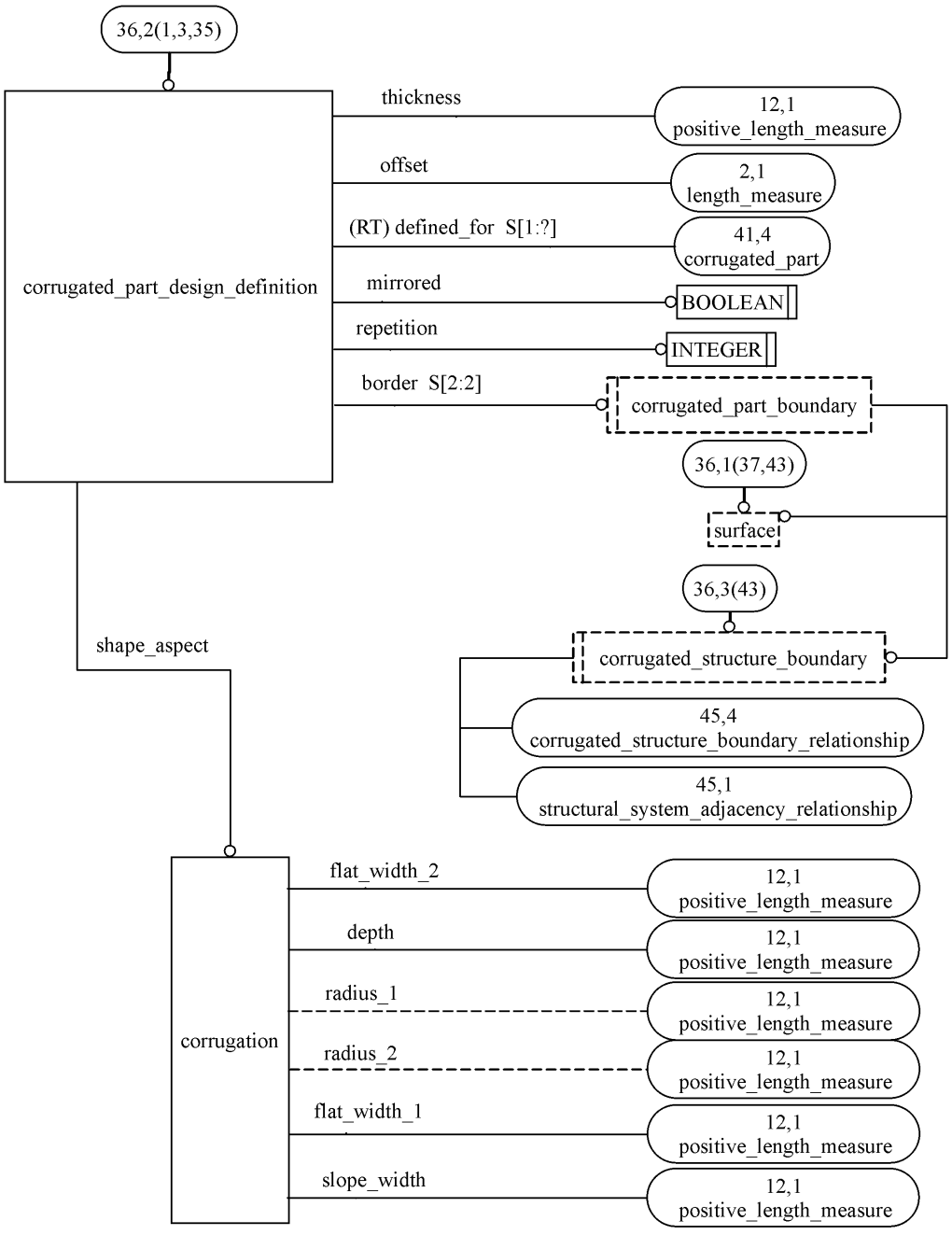


Figure G.36 — ARM EXPRESS-G diagram 36 of 49

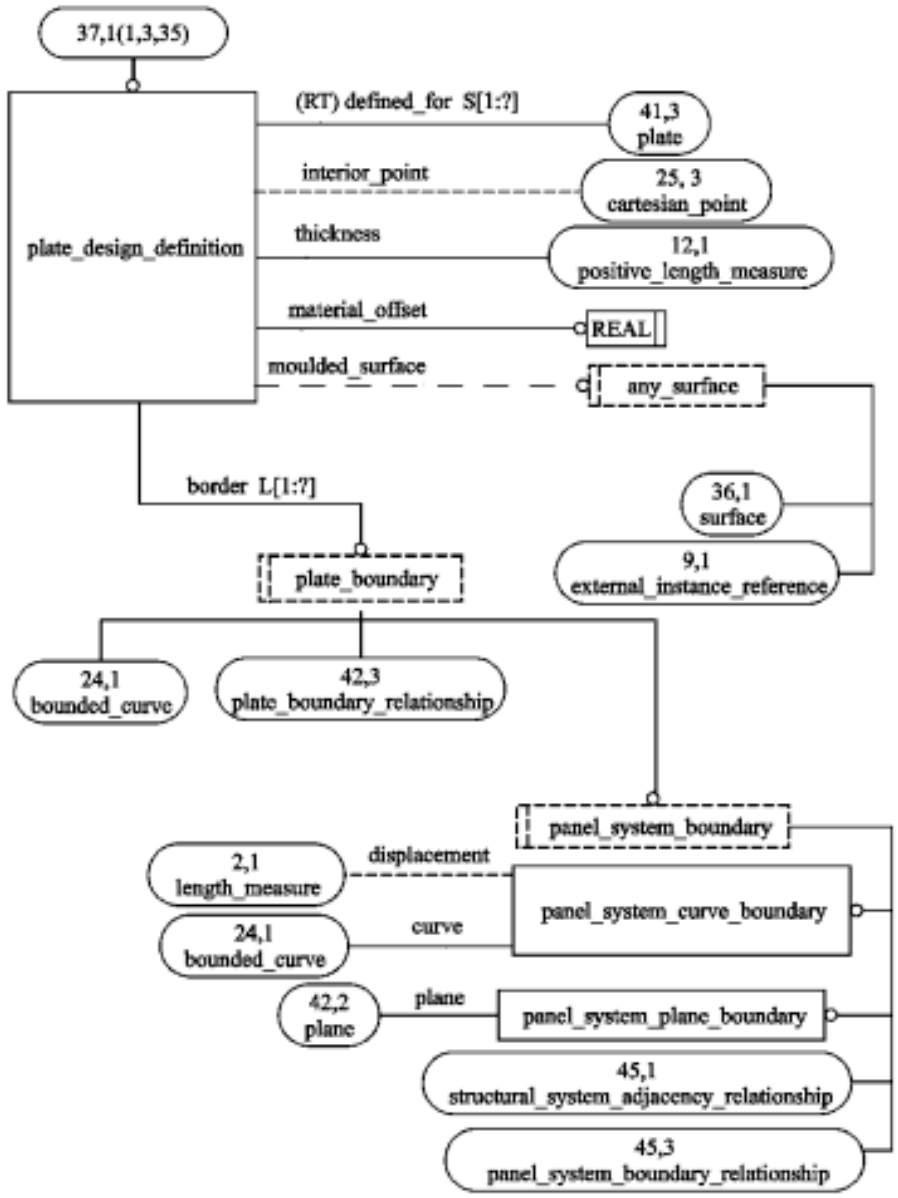


Figure G.37 — ARM EXPRESS-G diagram 37 of 49

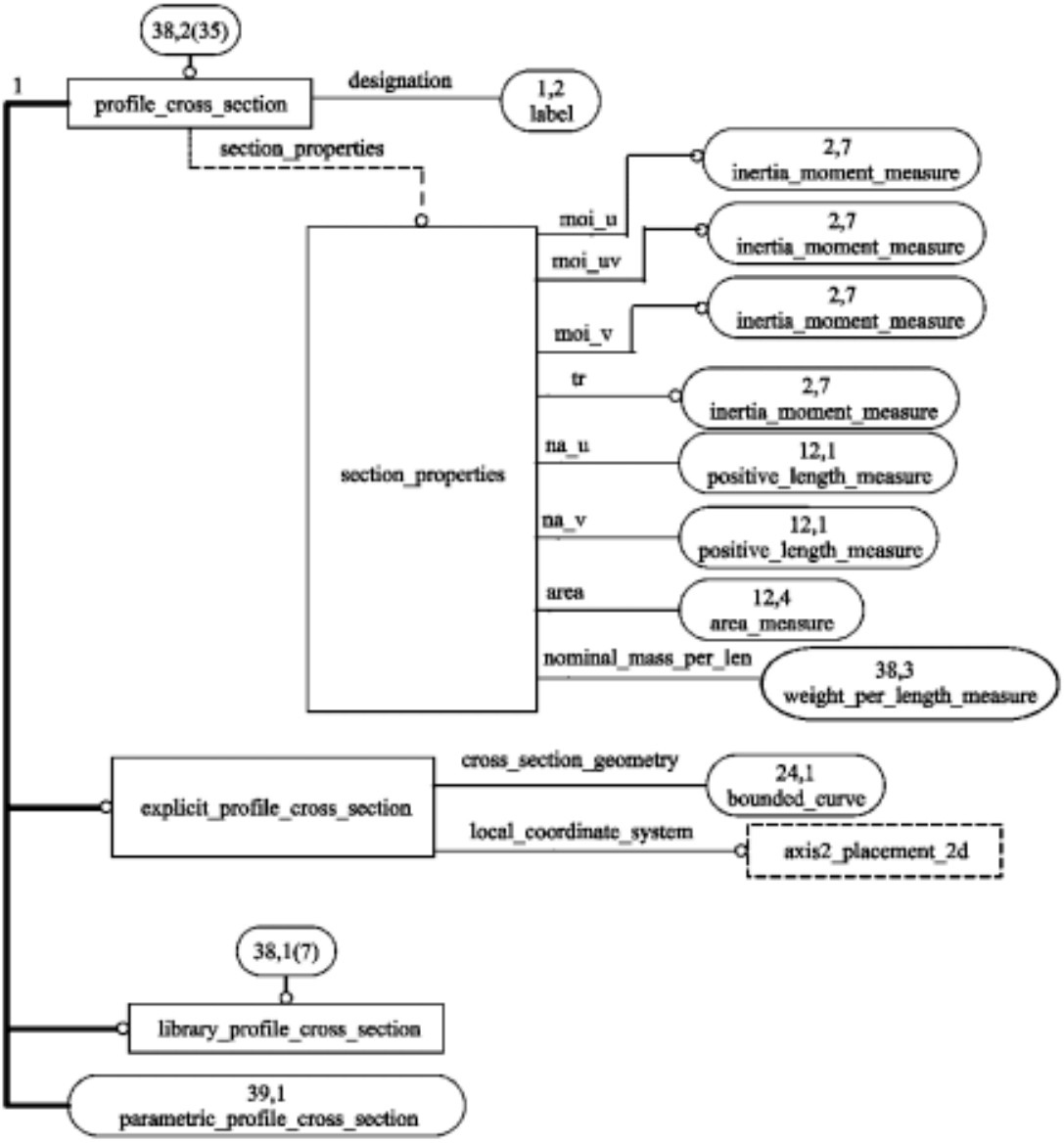


Figure G.38 — ARM EXPRESS-G diagram 38 of 49

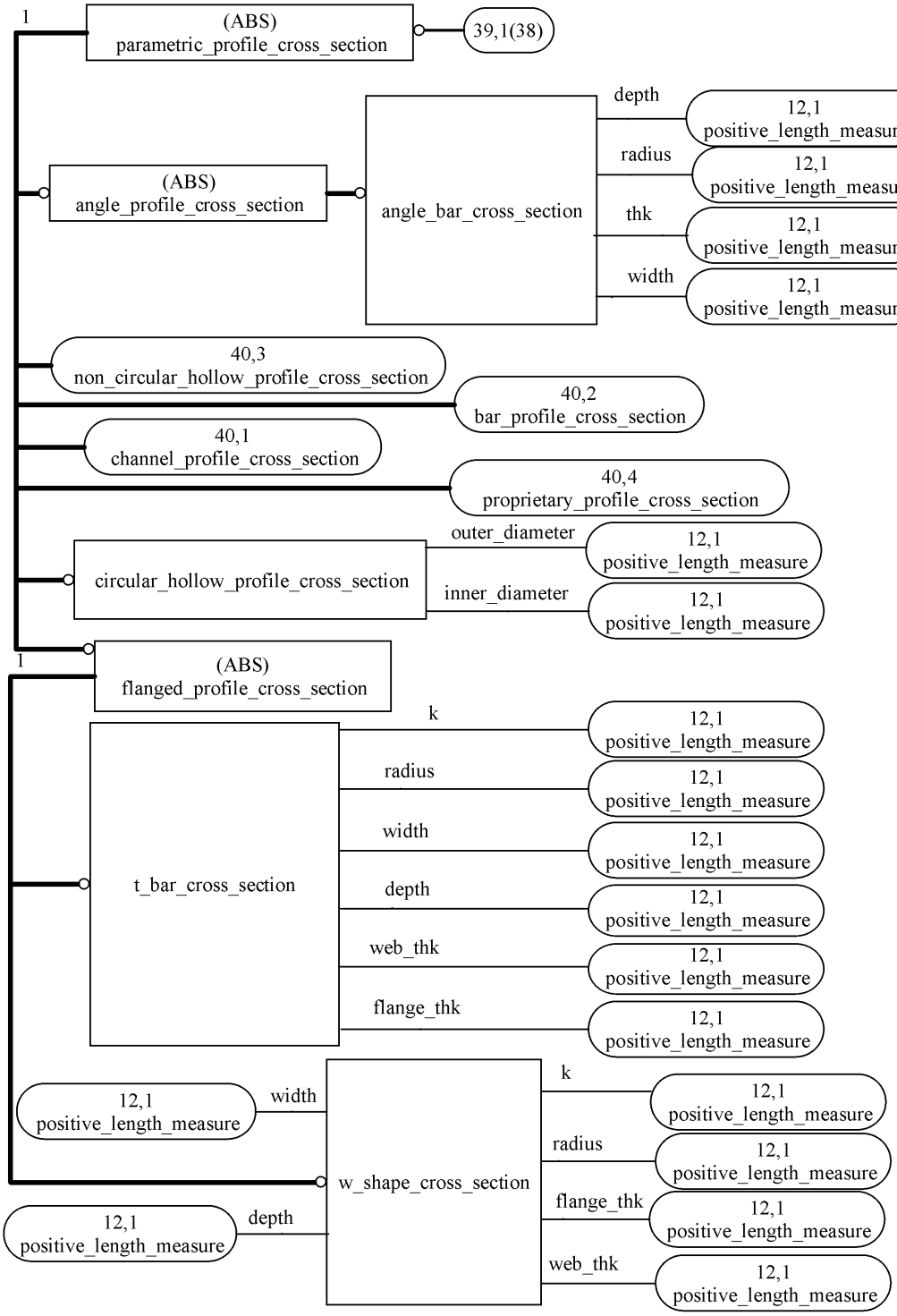


Figure G.39 — ARM EXPRESS-G diagram 39 of 49

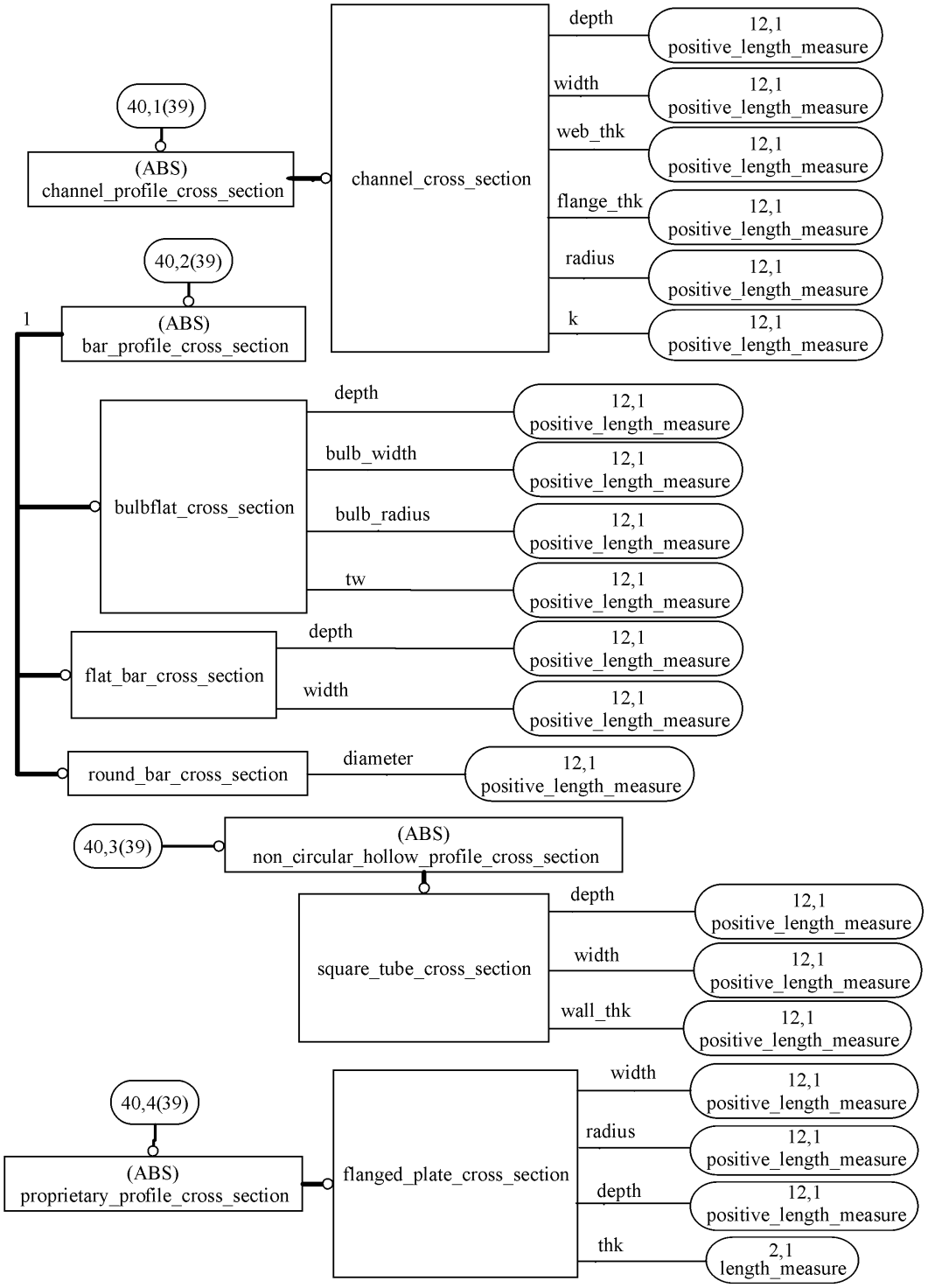


Figure G.40 — ARM EXPRESS-G diagram 40 of 49

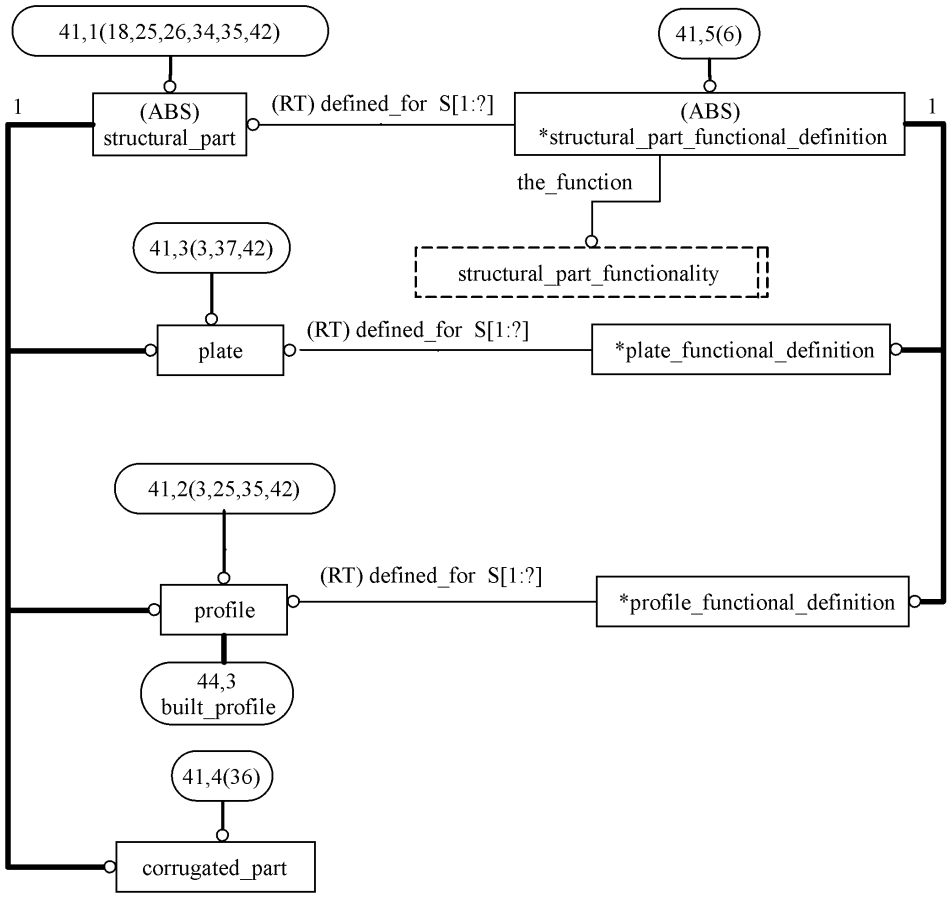


Figure G.41 — ARM EXPRESS-G diagram 41 of 49

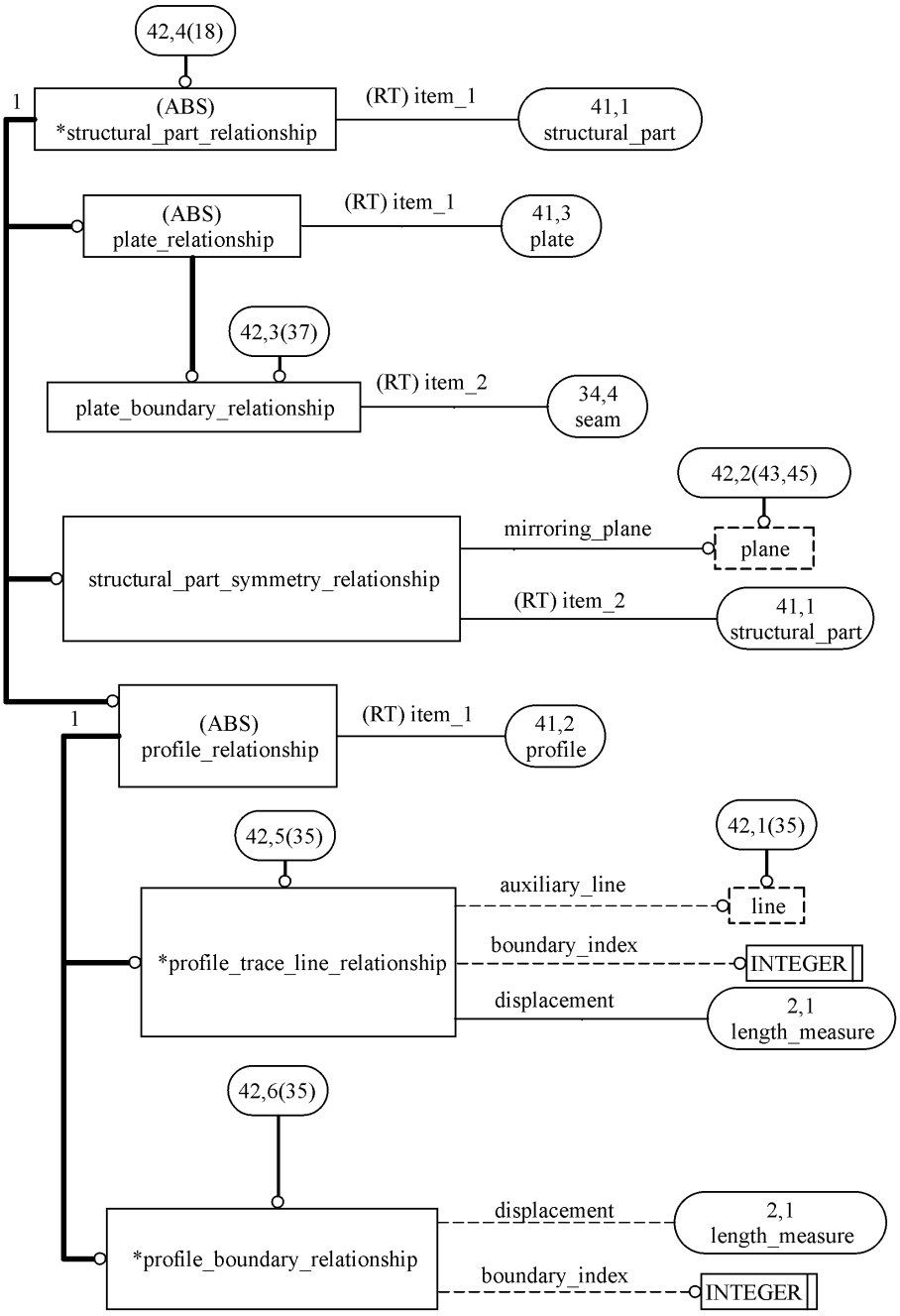


Figure G.42 — ARM EXPRESS-G diagram 42 of 49

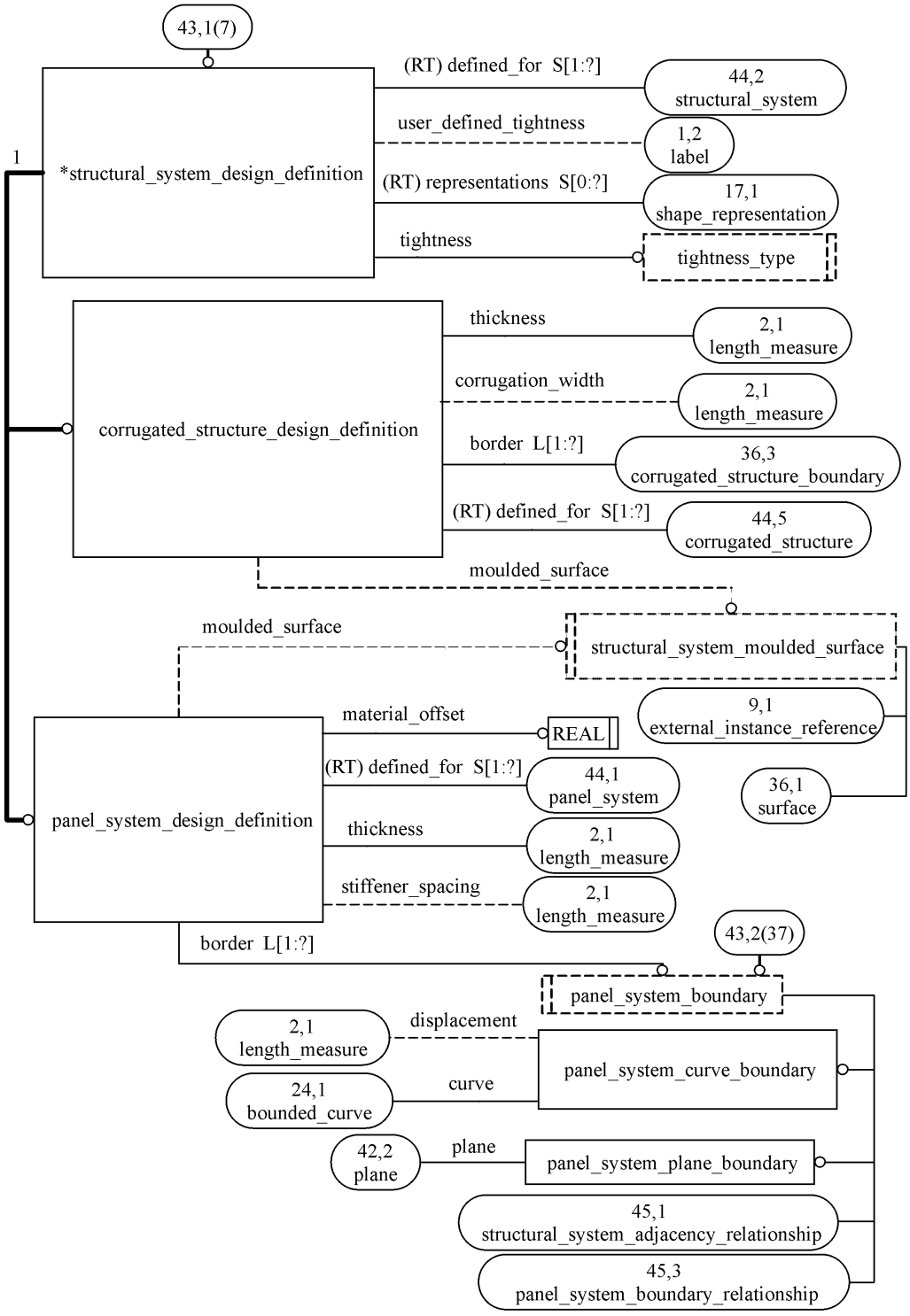


Figure G.43 — ARM EXPRESS-G diagram 43 of 49

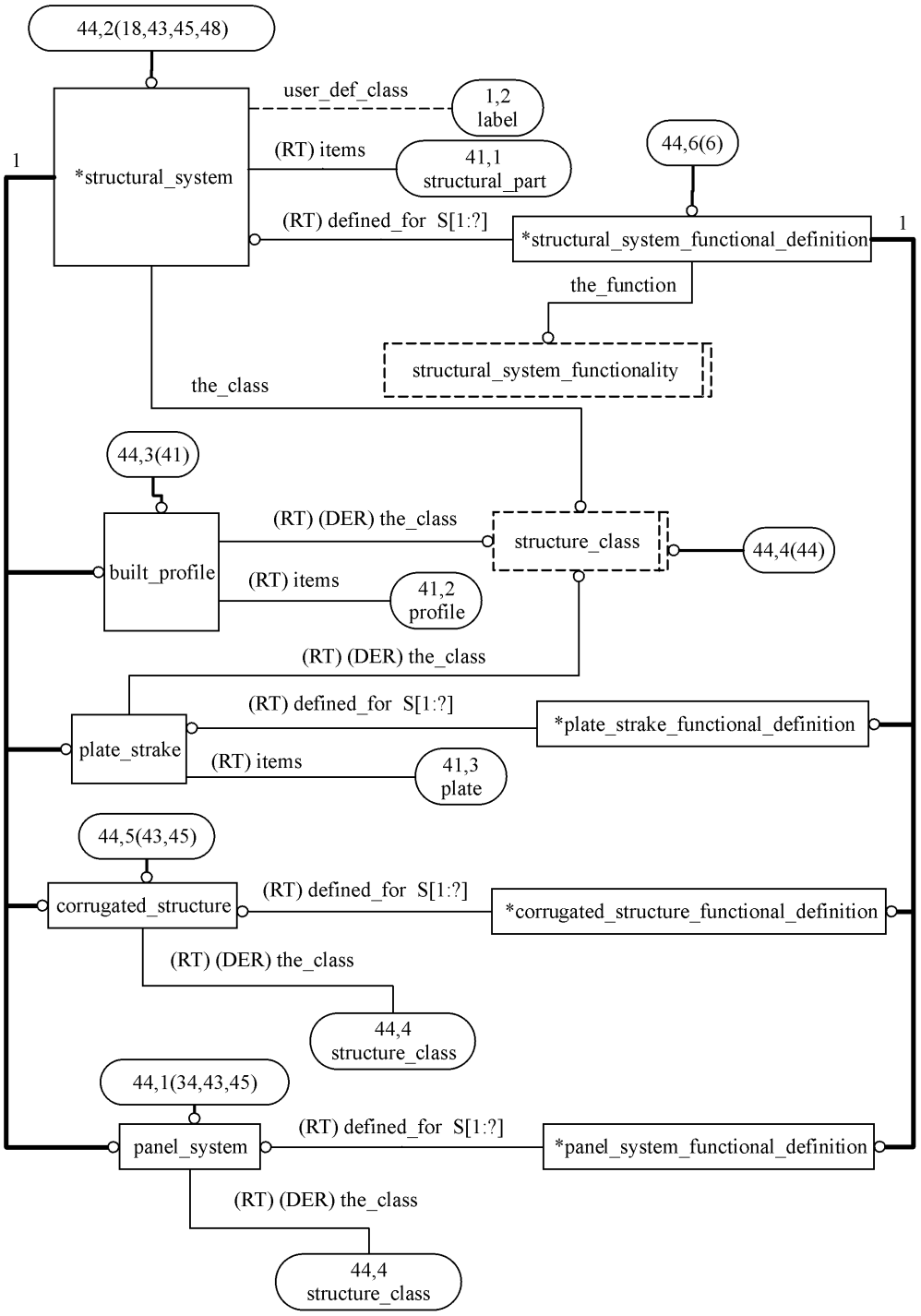


Figure G.44 — ARM EXPRESS-G diagram 44 of 49

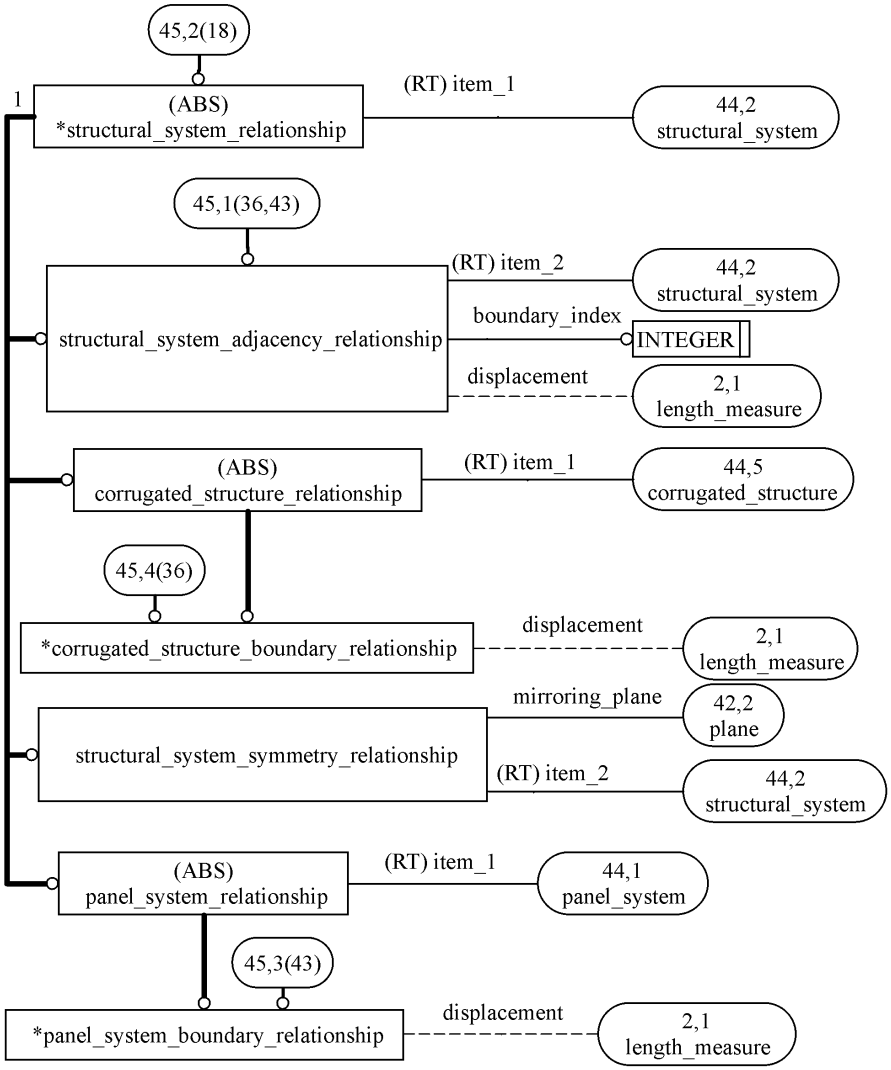


Figure G.45 — ARM EXPRESS-G diagram 45 of 49

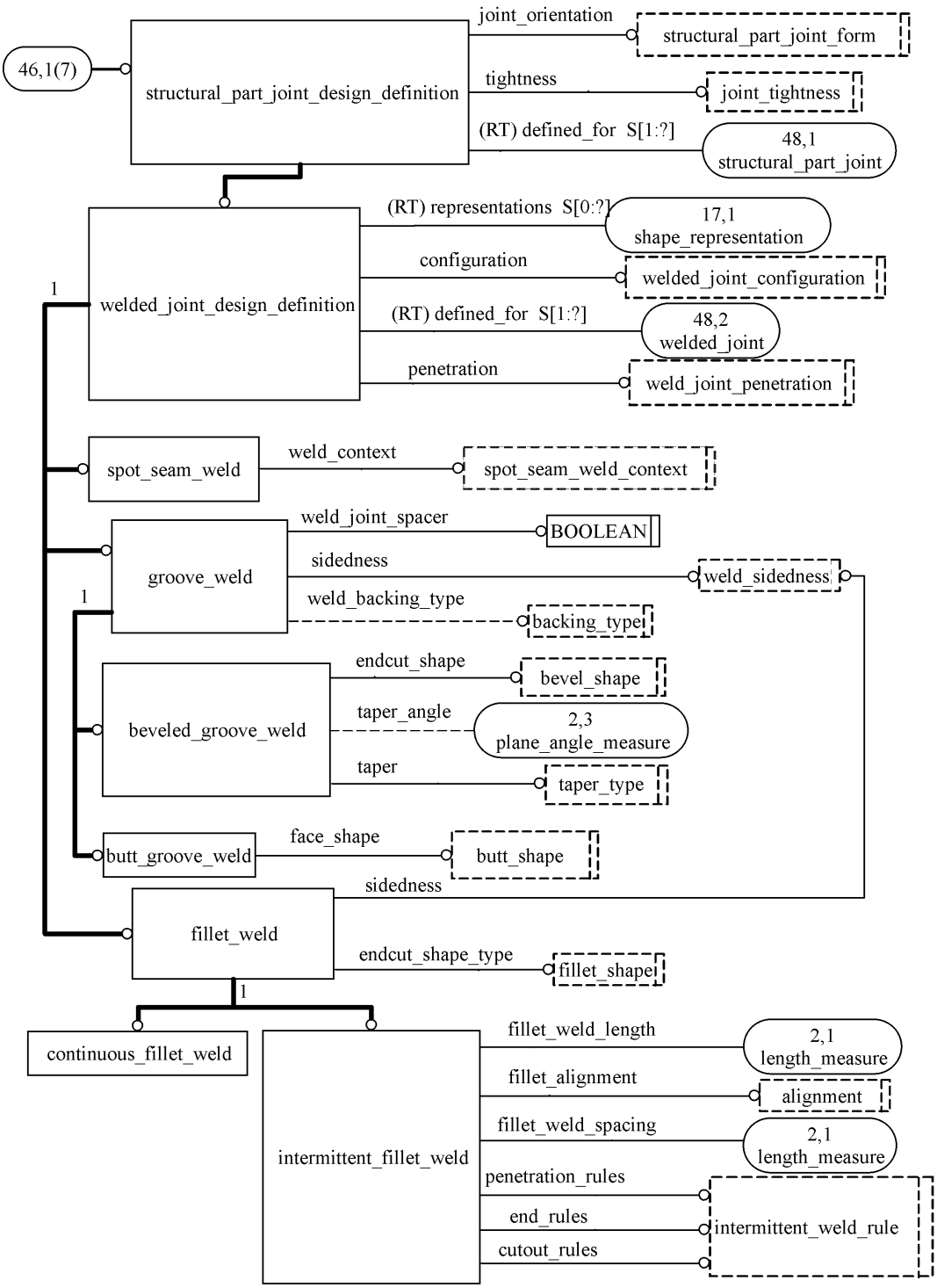


Figure G.46 — ARM EXPRESS-G diagram 46 of 49

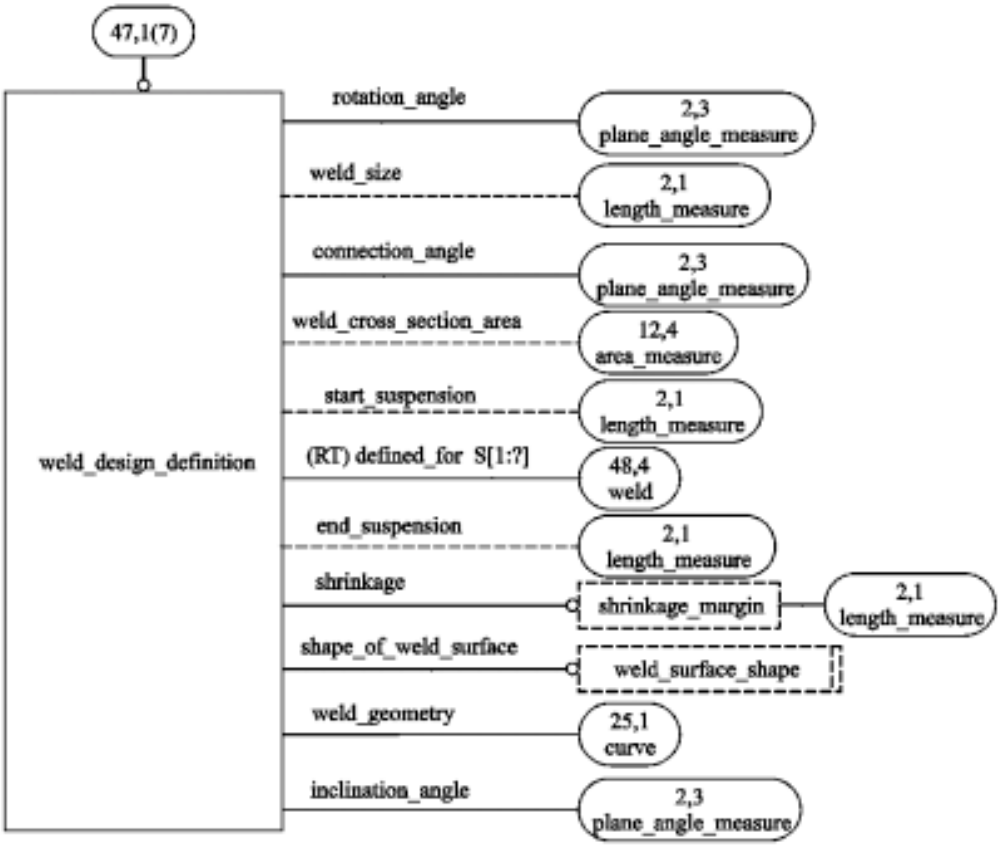


Figure G.47 — ARM EXPRESS-G diagram 47 of 49

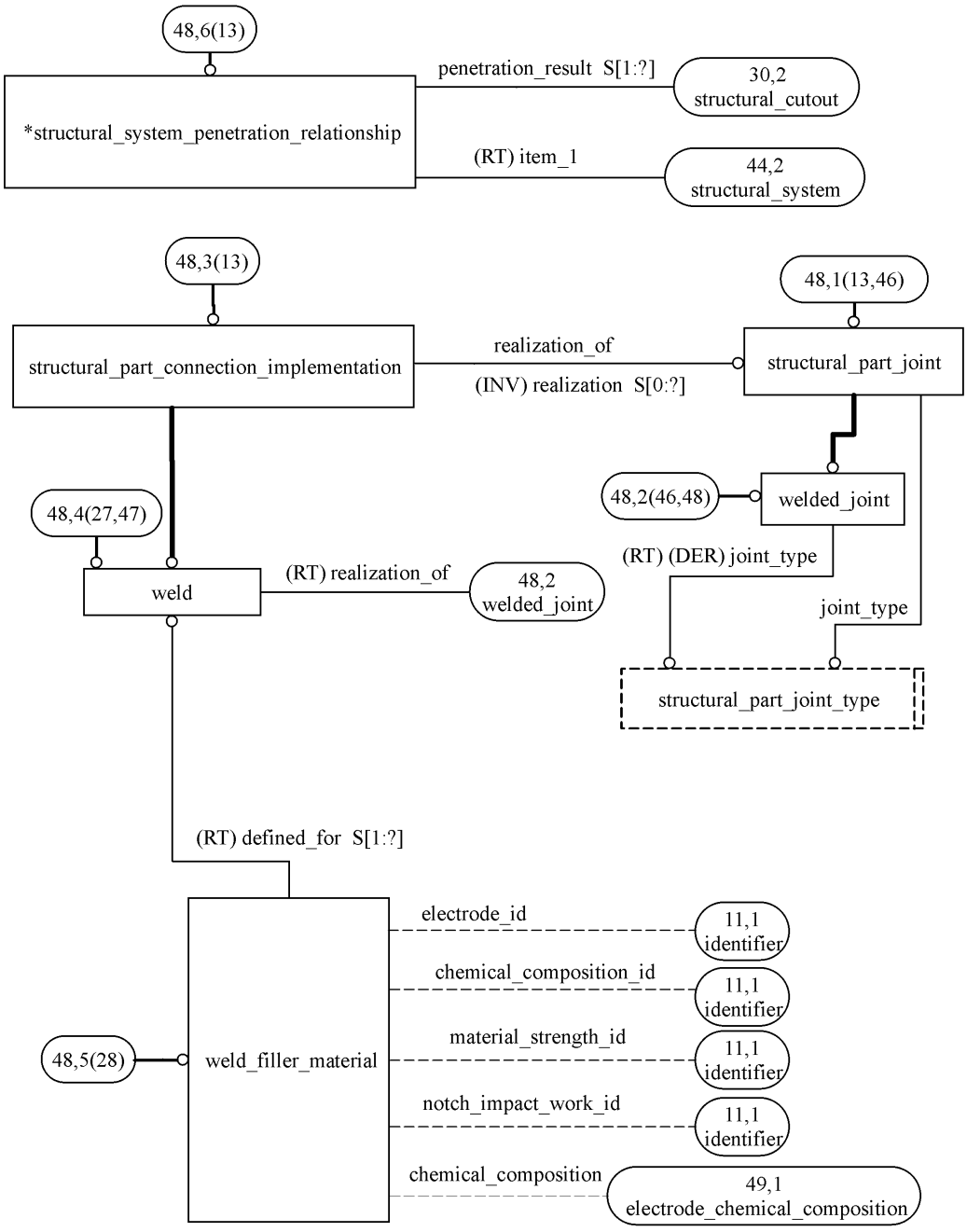


Figure G.48 — ARM EXPRESS-G diagram 48 of 49

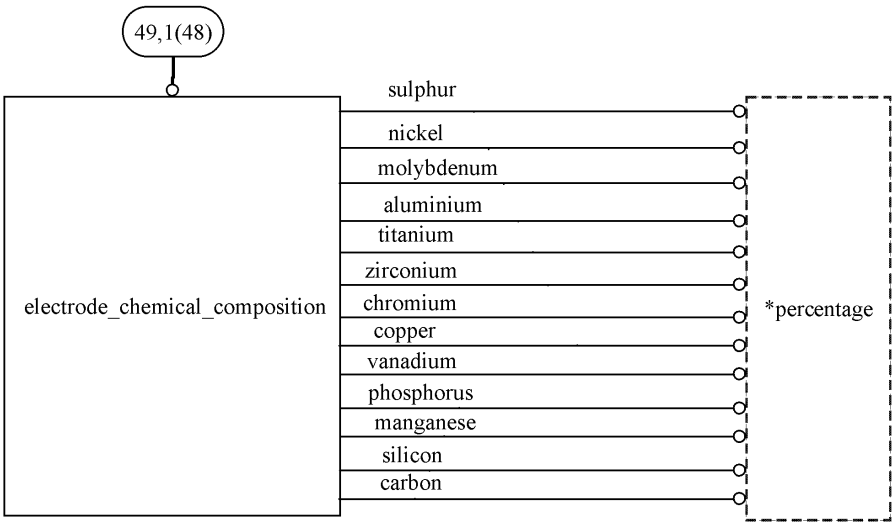


Figure G.49 — ARM EXPRESS-G diagram 49 of 49

Annex H
(informative)

EXPRESS-G Diagrams

Figure H.1 through H.29 correspond to the AIM EXPRESS annotated listing given in annex A. The figures use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined an annex A of ISO 10303-11.

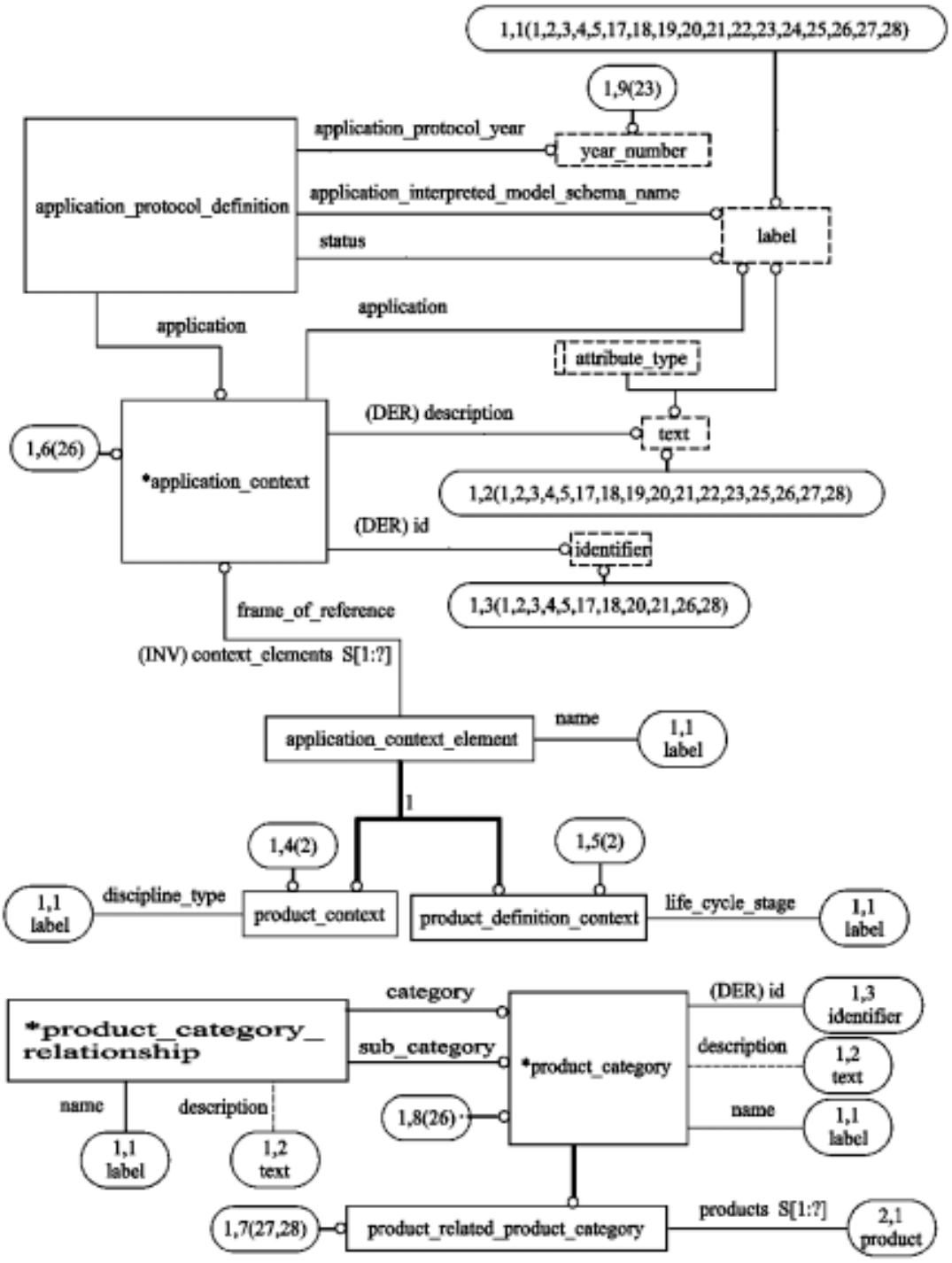


Figure H.1 — AIM diagram 1 of 28 in EXPRESS-G

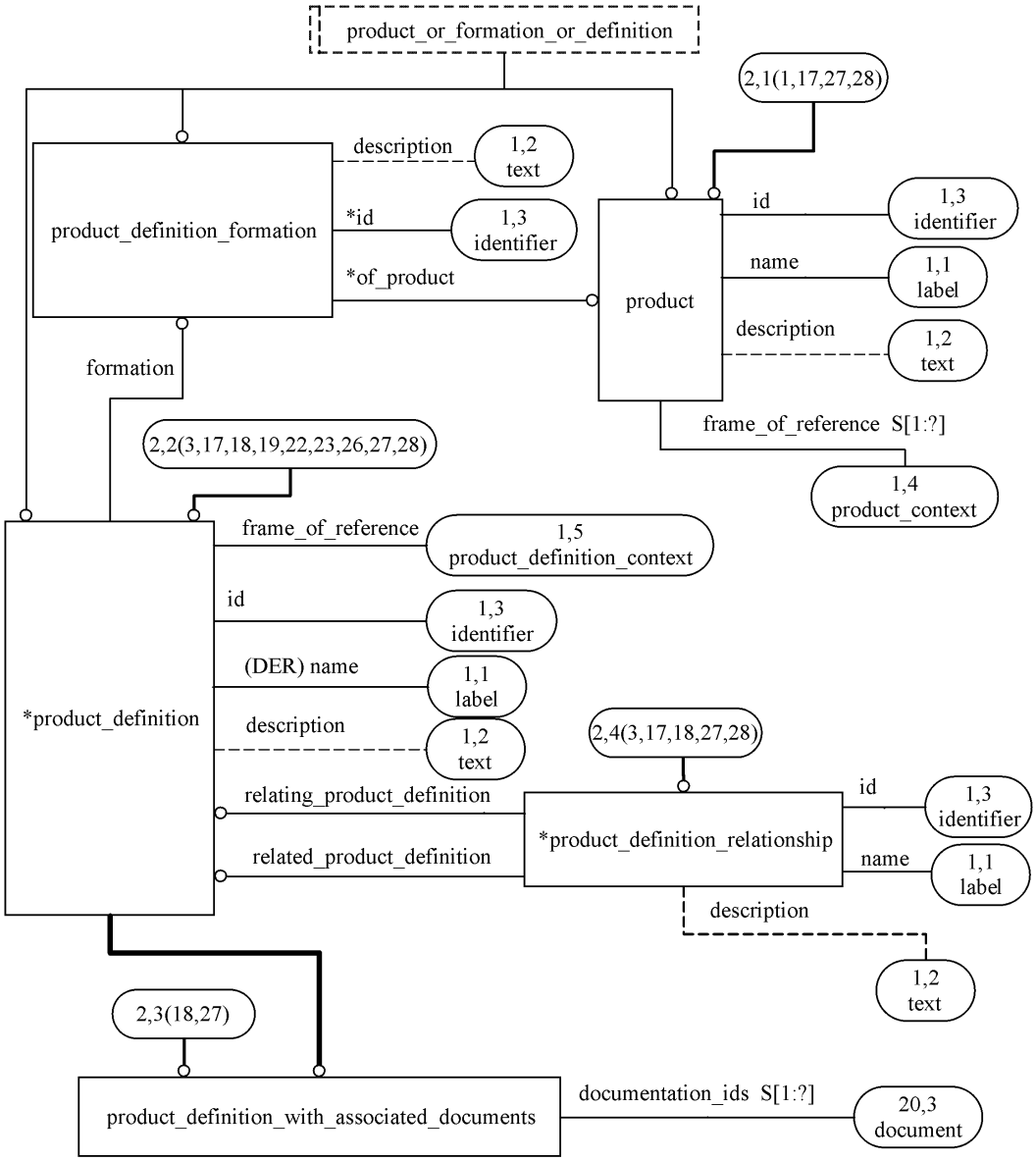


Figure H.2 — AIM diagram 2 of 28 in EXPRESS-G

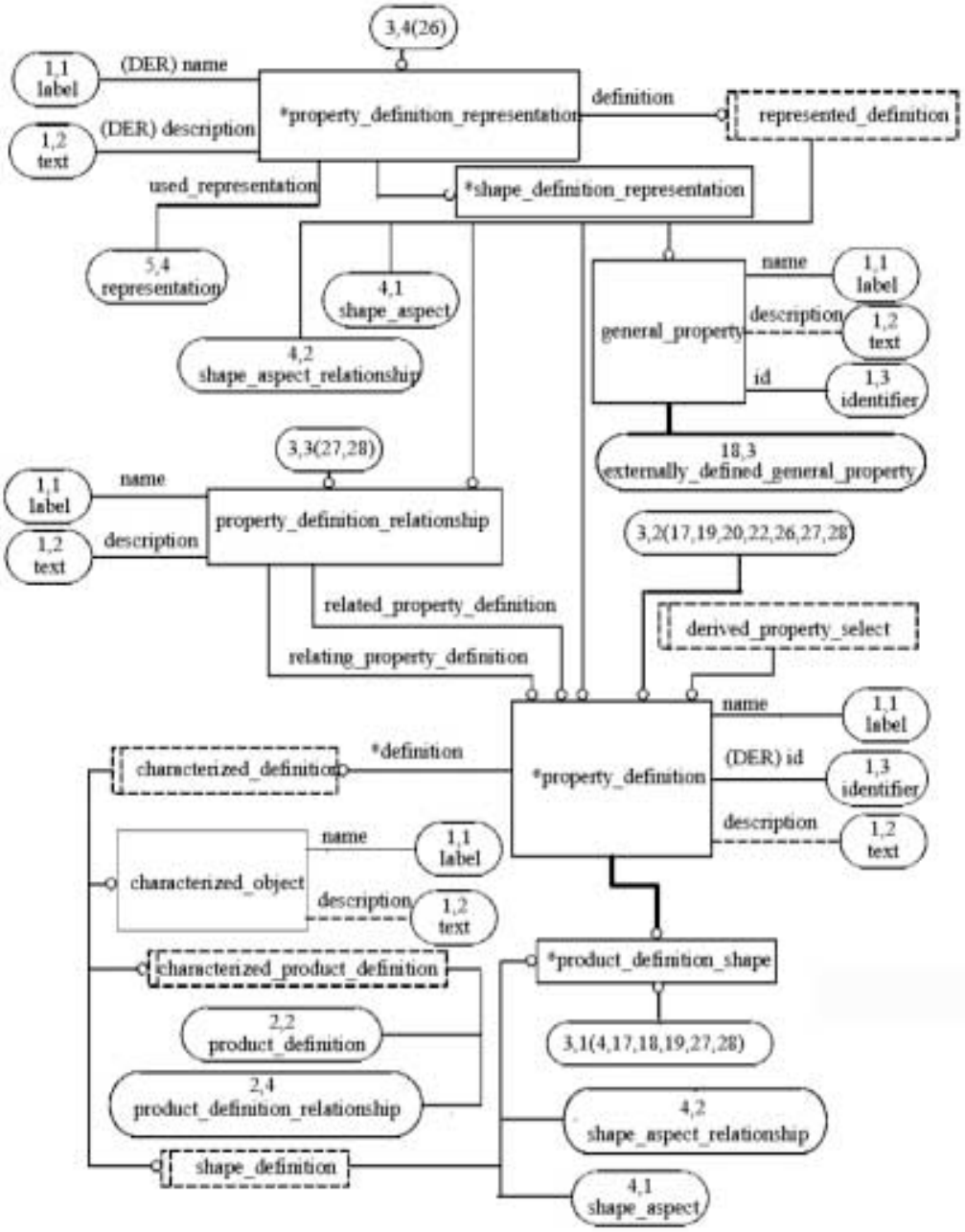


Figure H.3 — AIM diagram 3 of 28 in EXPRESS-G

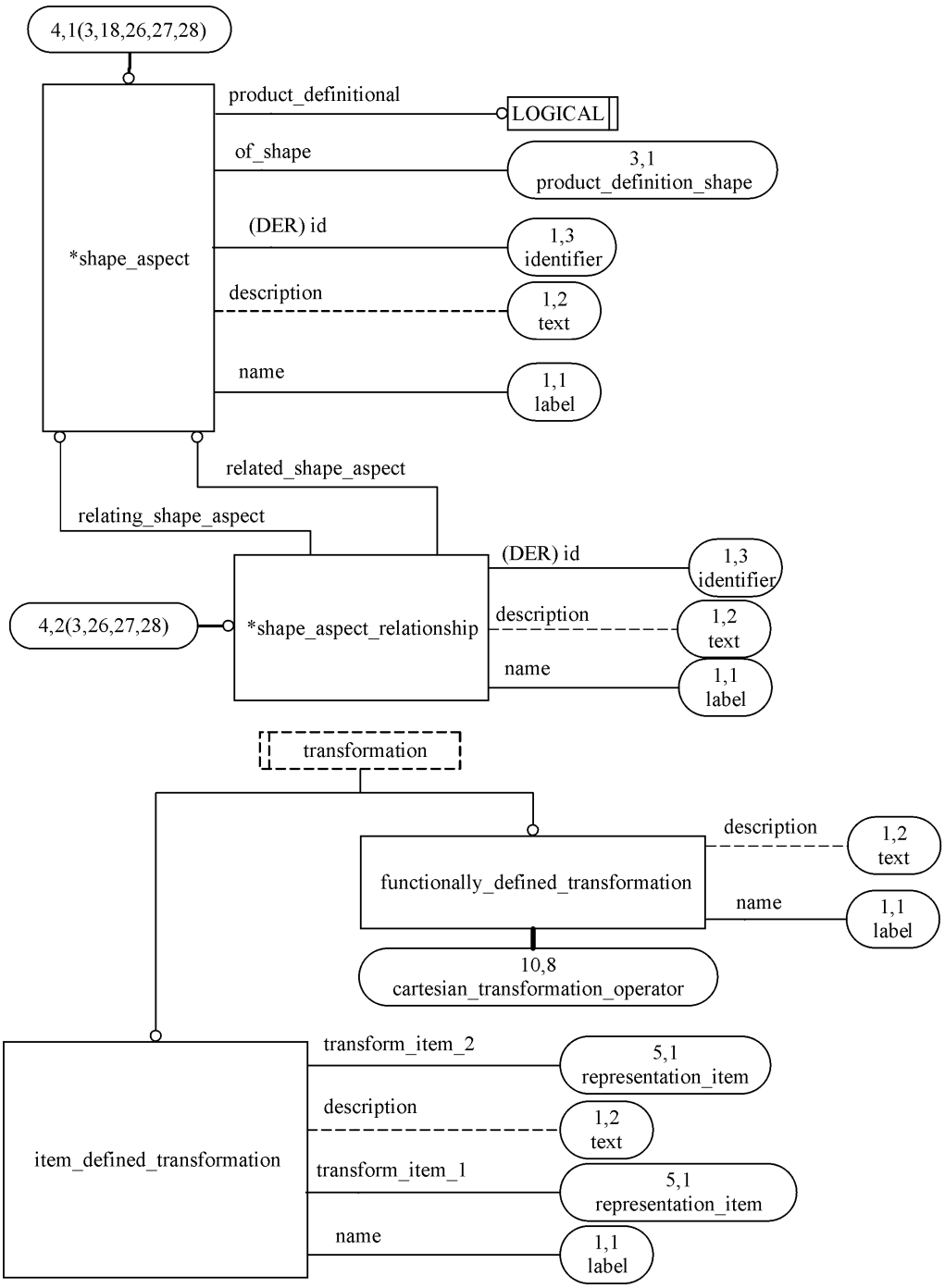


Figure H.4 — AIM diagram 4 of 28 in EXPRESS-G

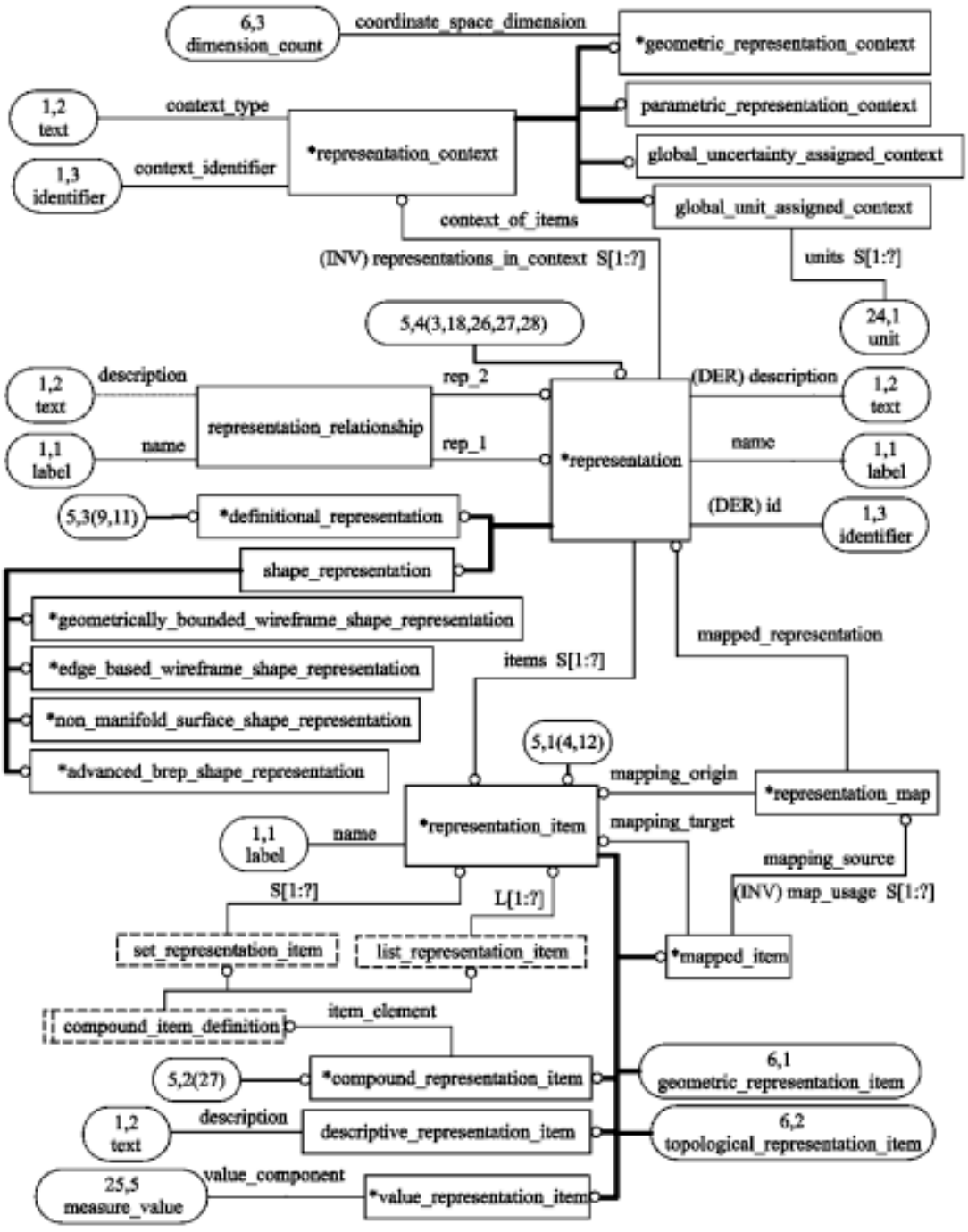


Figure H.5 — AIM diagram 5 of 28 in EXPRESS-G

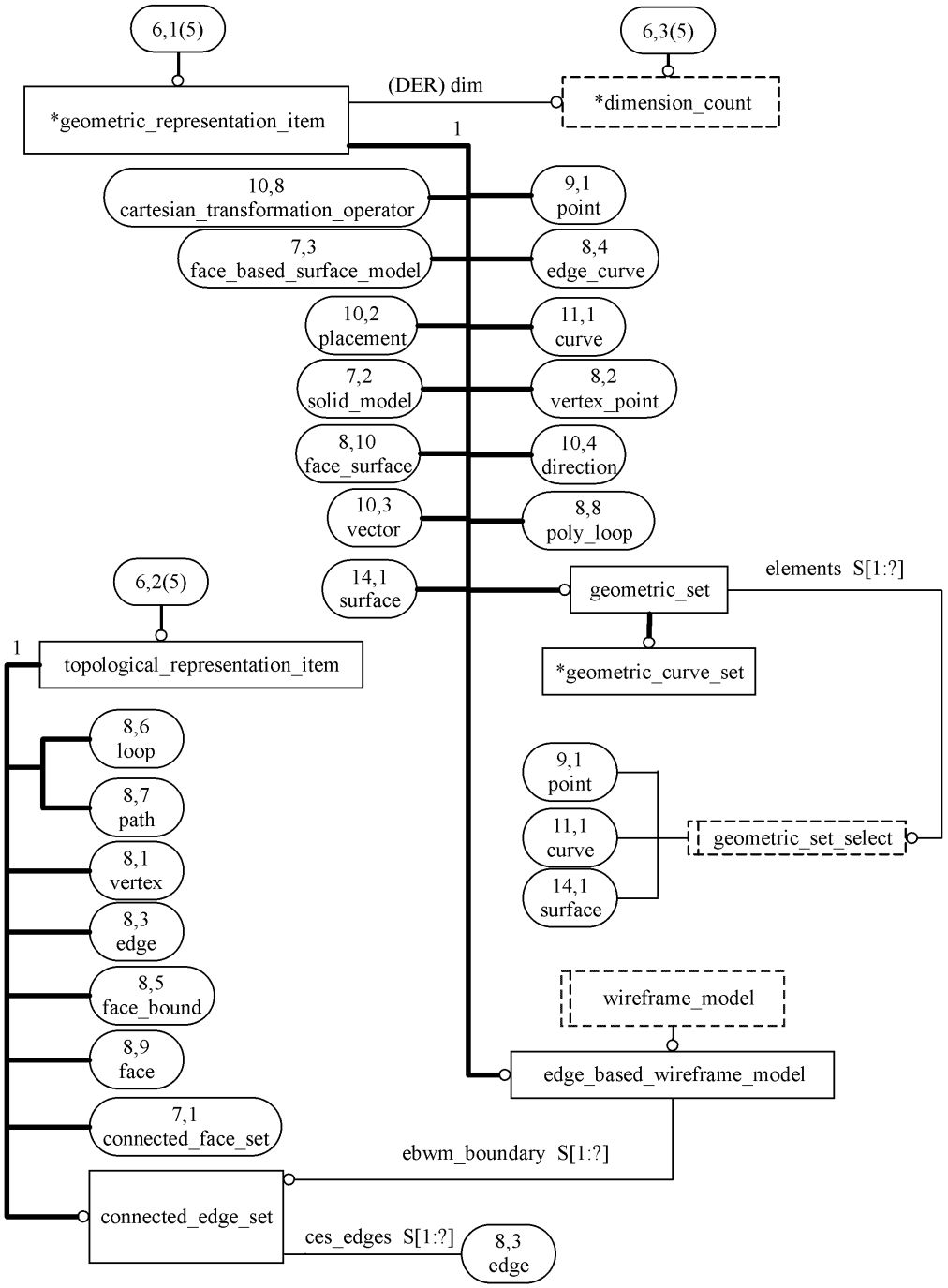


Figure H.6 — AIM diagram 6 of 28 in EXPRESS-G

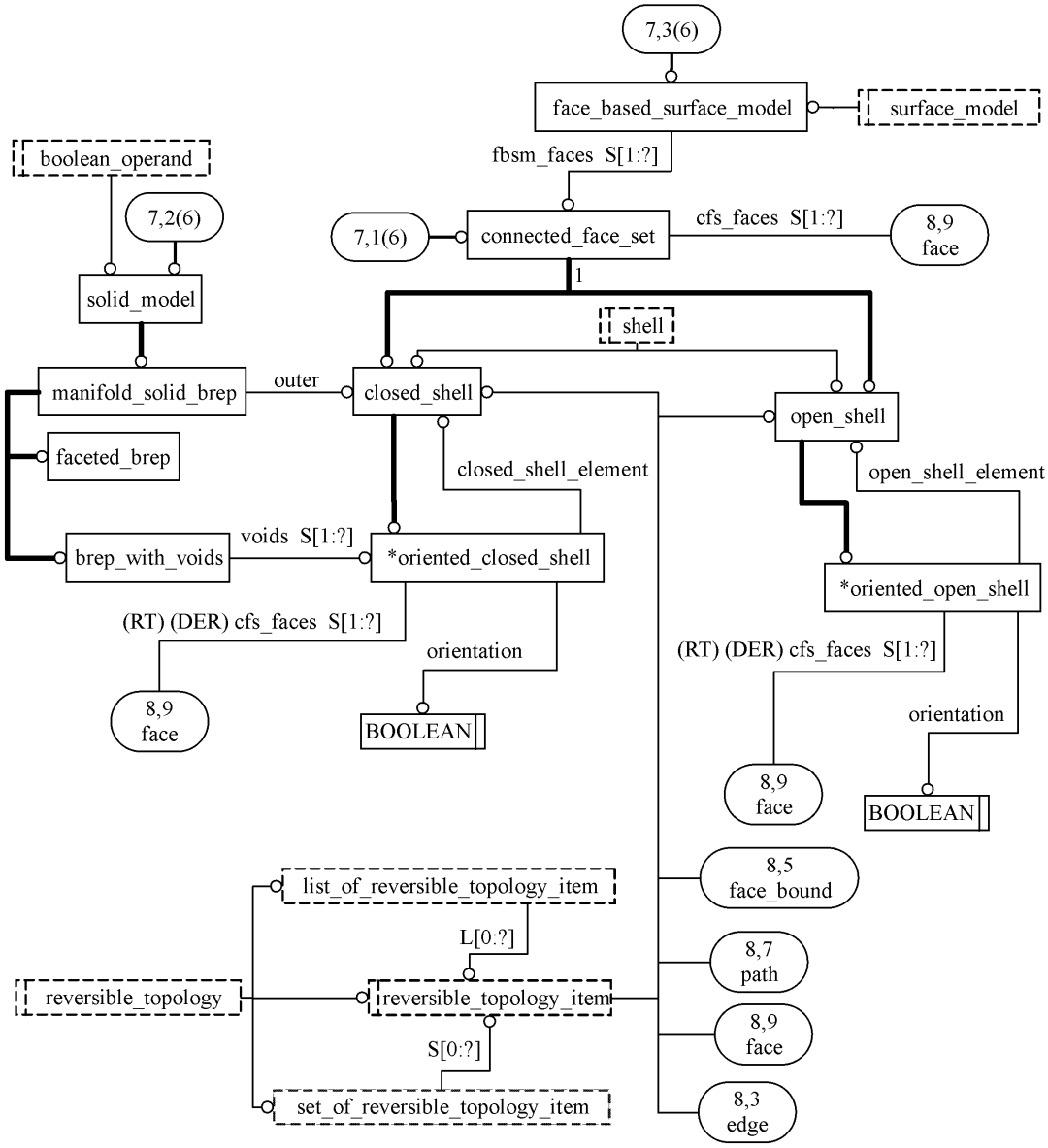


Figure H.7 — AIM diagram 7 of 28 in EXPRESS-G

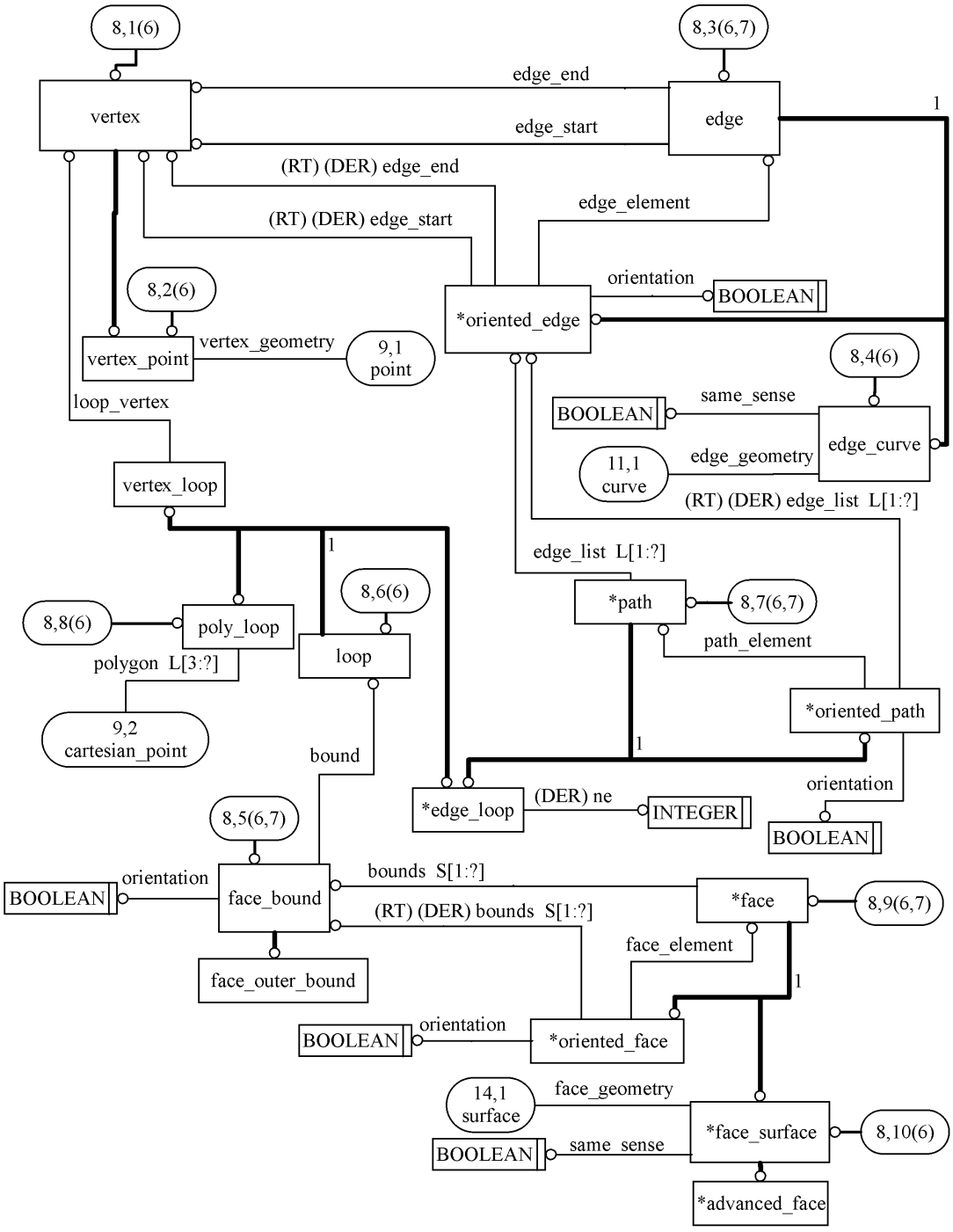


Figure H.8 — AIM diagram 8 of 28 in EXPRESS-G

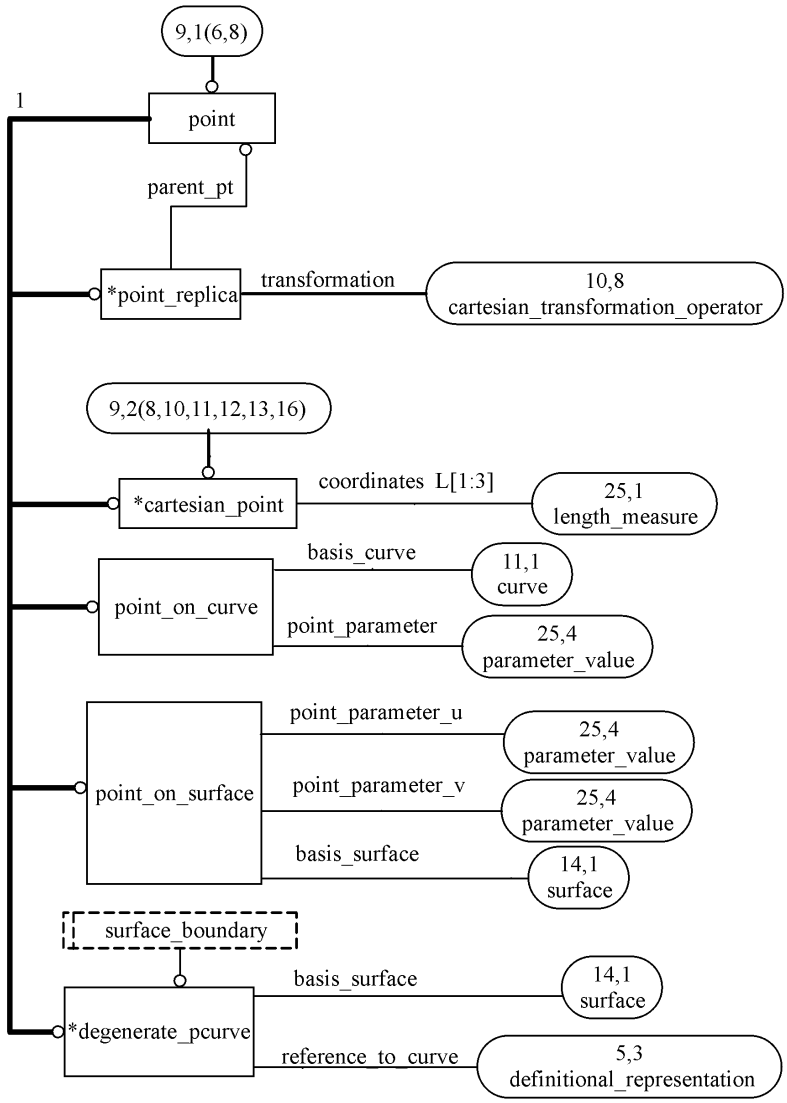


Figure H.9 — AIM diagram 9 of 28 in EXPRESS-G

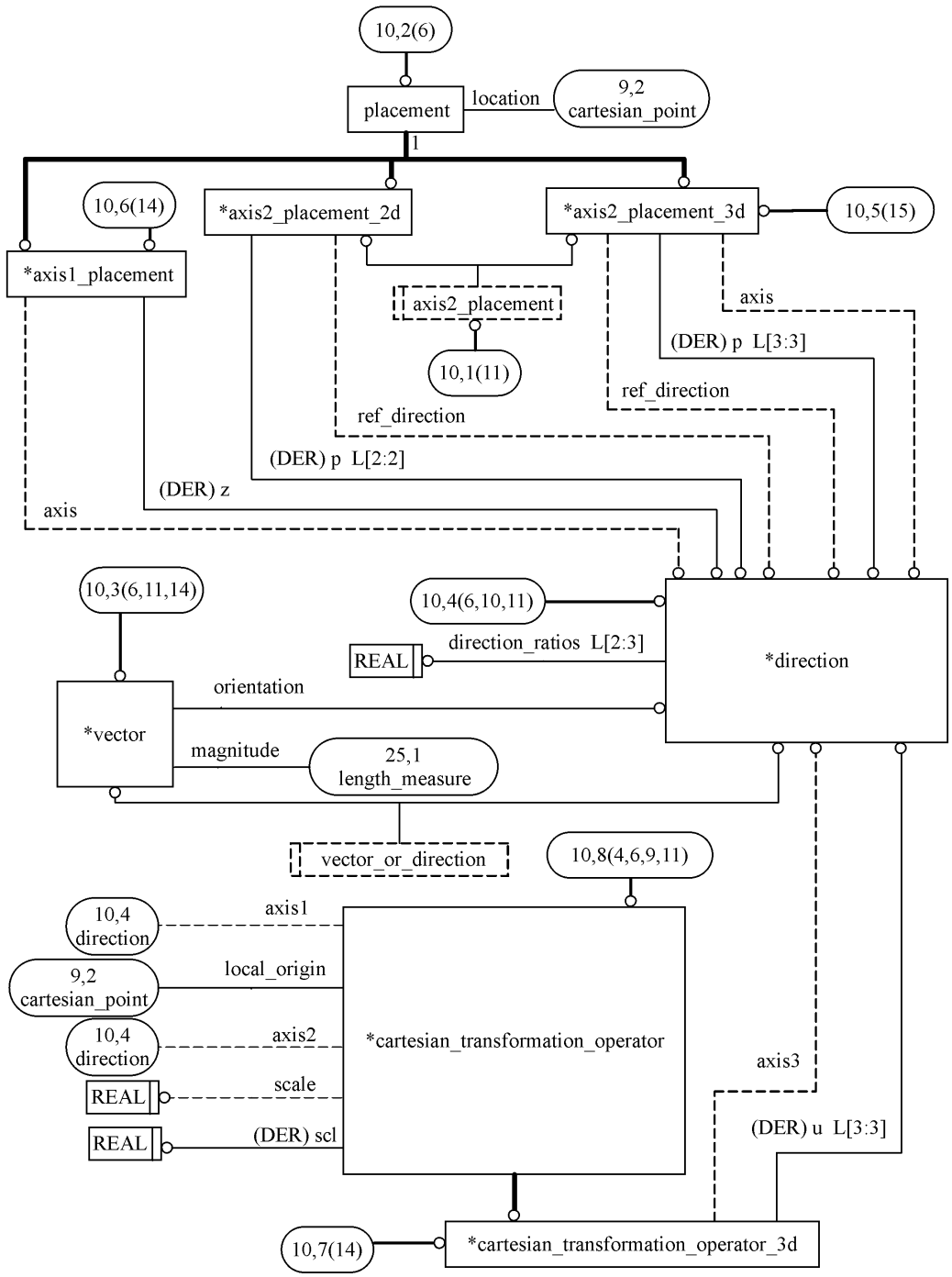


Figure H.10 — AIM diagram 10 of 28 in EXPRESS-G

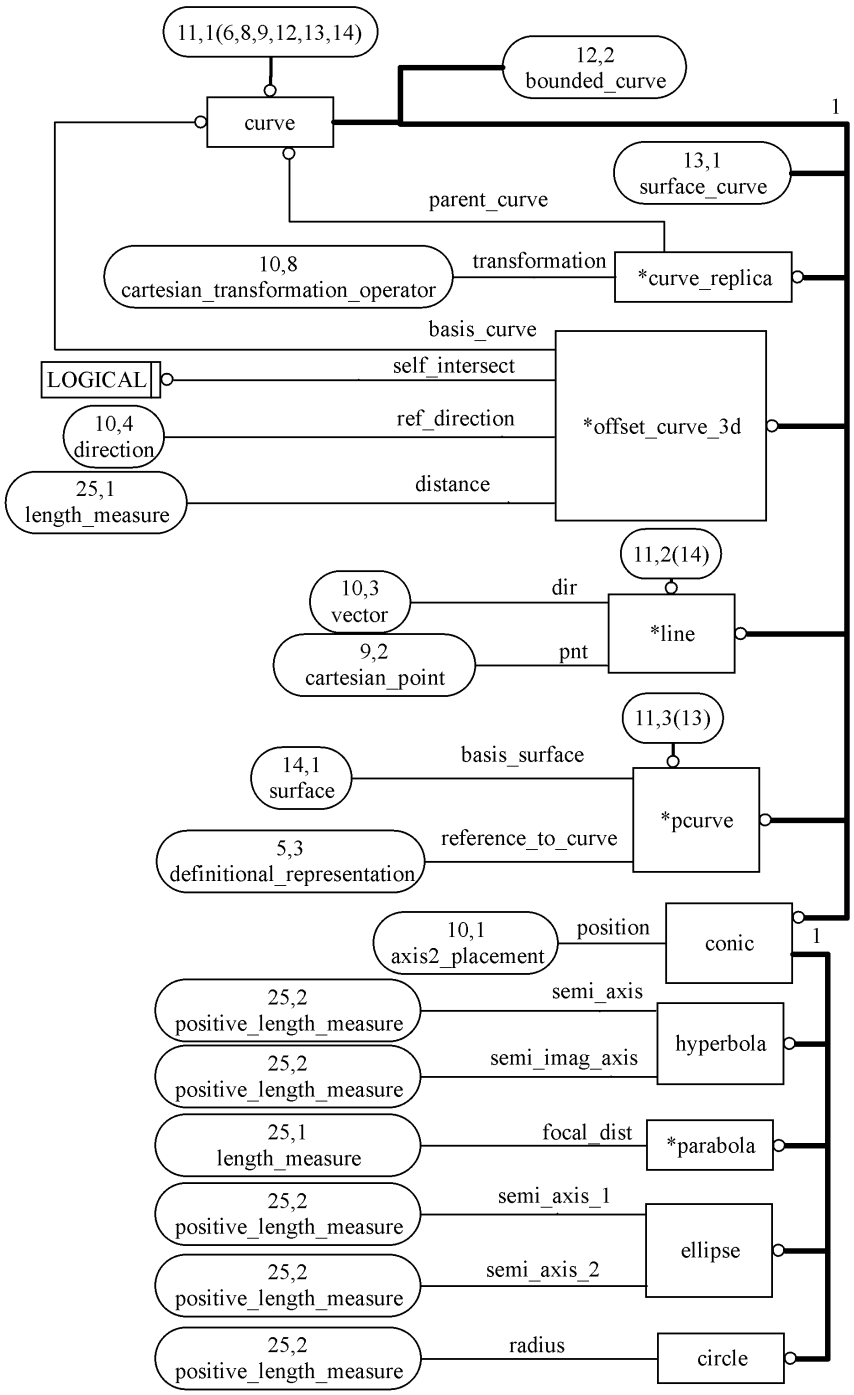


Figure H.11 — AIM diagram 11 of 28 in EXPRESS-G

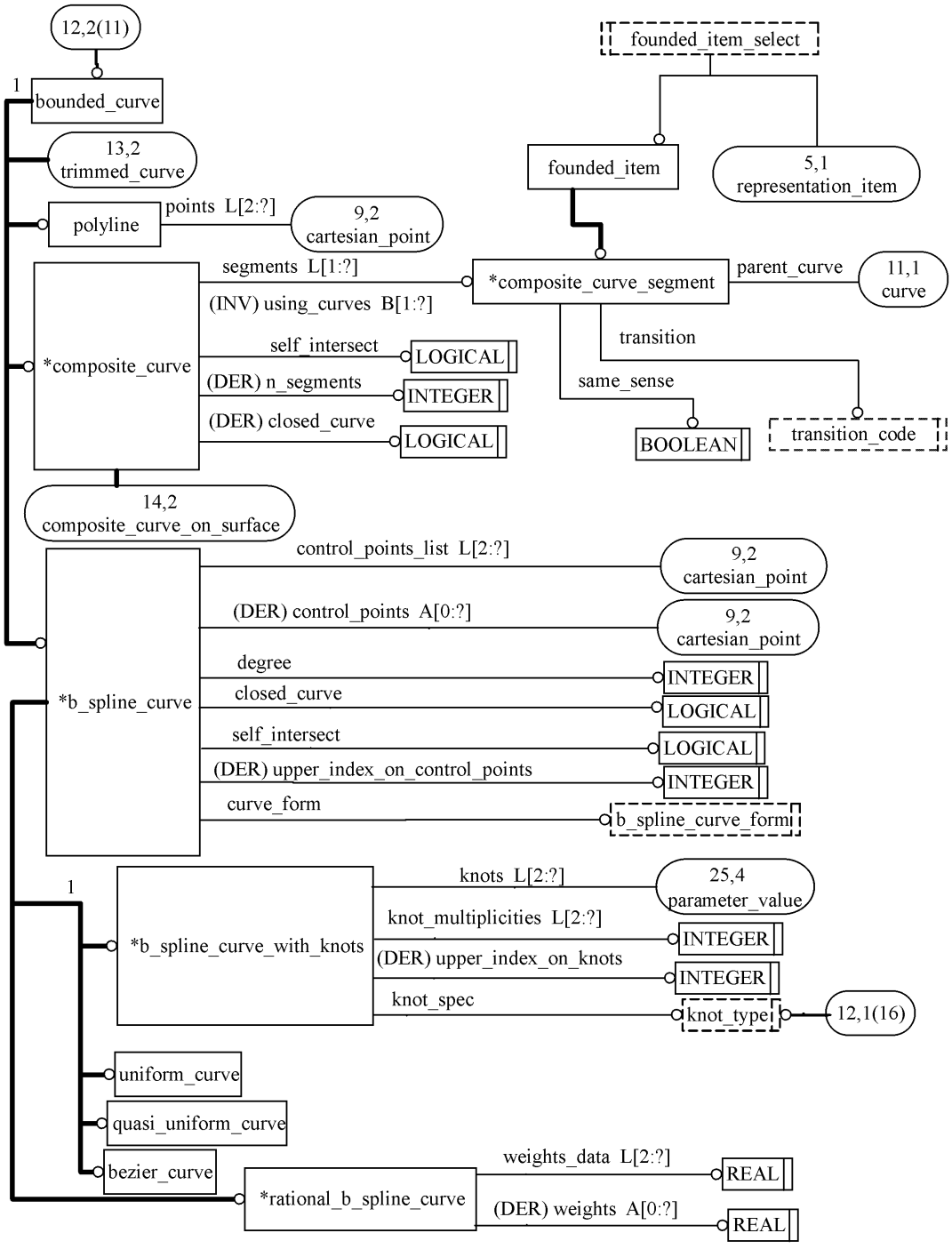


Figure H.12 — AIM diagram 12 of 28 in EXPRESS-G

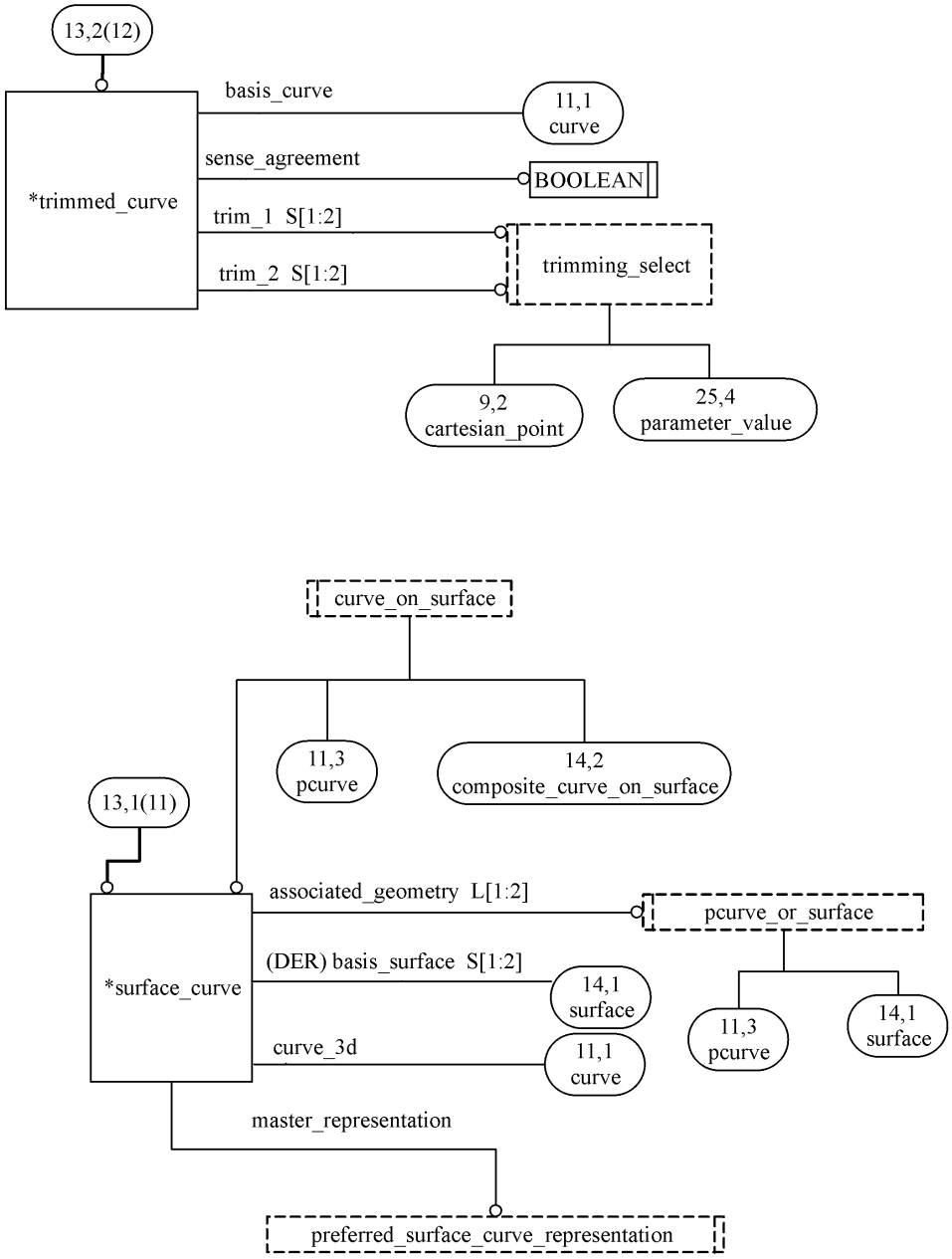


Figure H.13 — AIM diagram 13 of 28 in EXPRESS-G

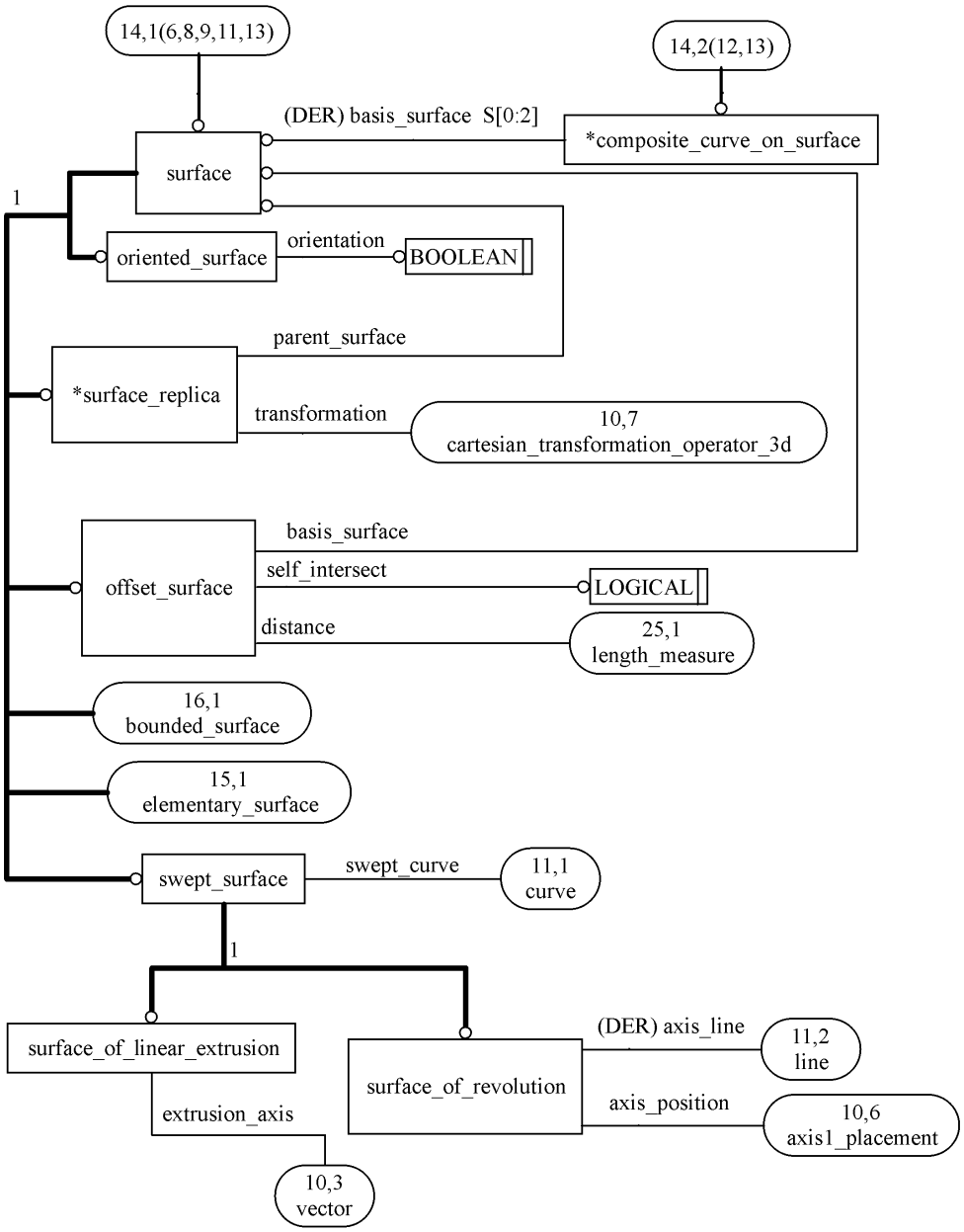


Figure H.14 — AIM diagram 14 of 28 in EXPRESS-G

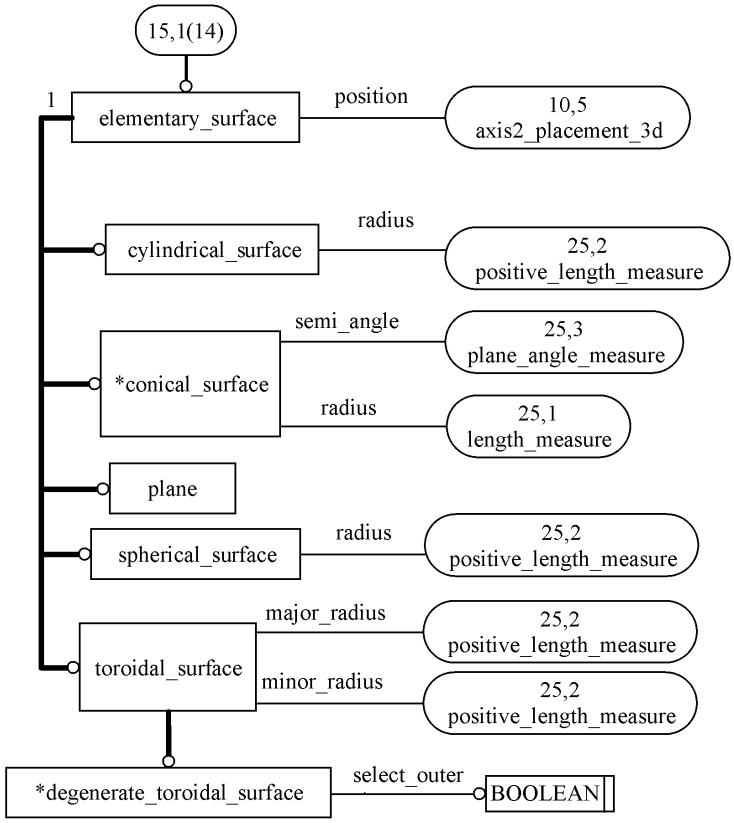


Figure H.15 — AIM diagram 15 of 28 in EXPRESS-G

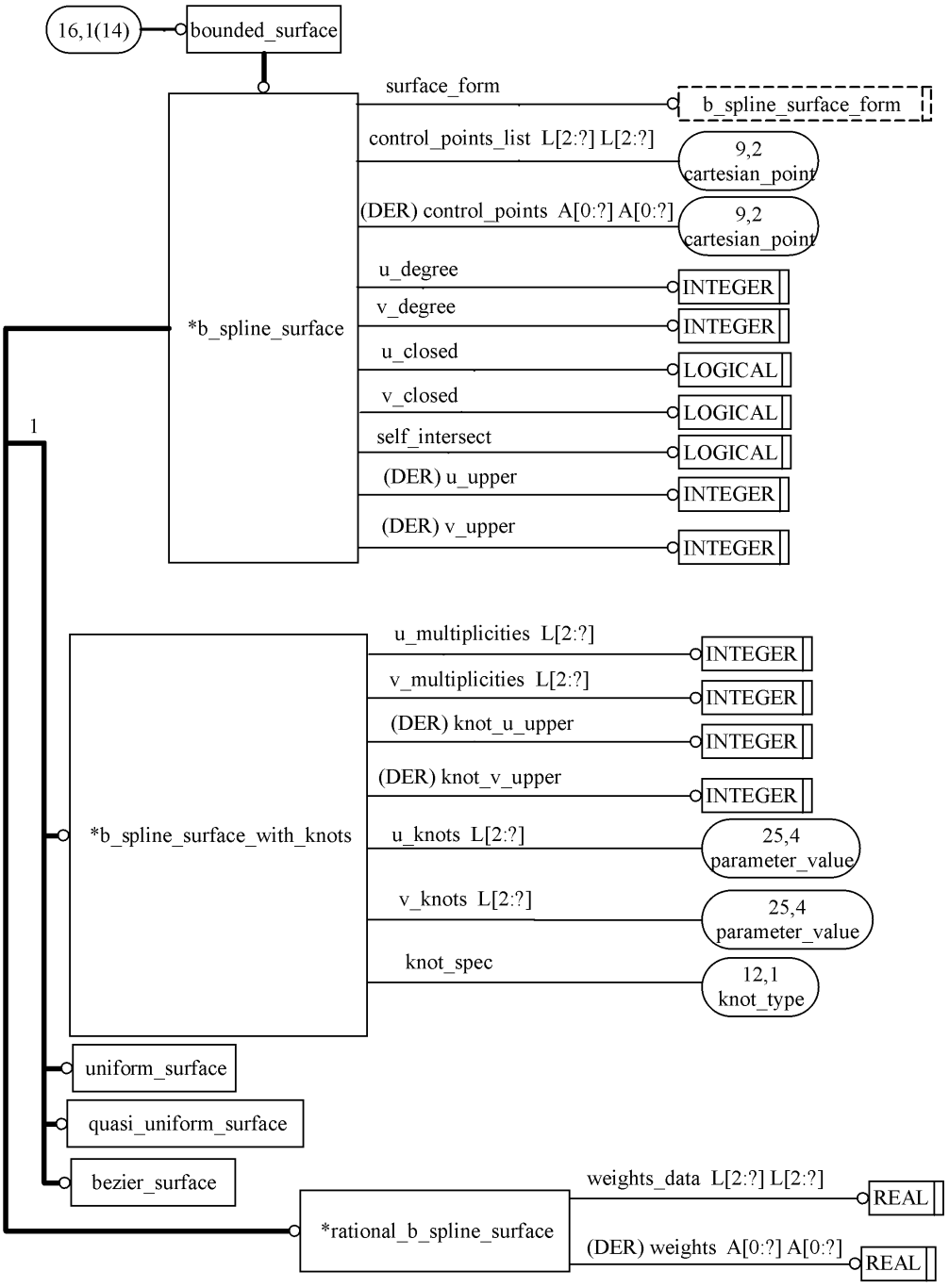


Figure H.16 — AIM diagram 16 of 28 in EXPRESS-G

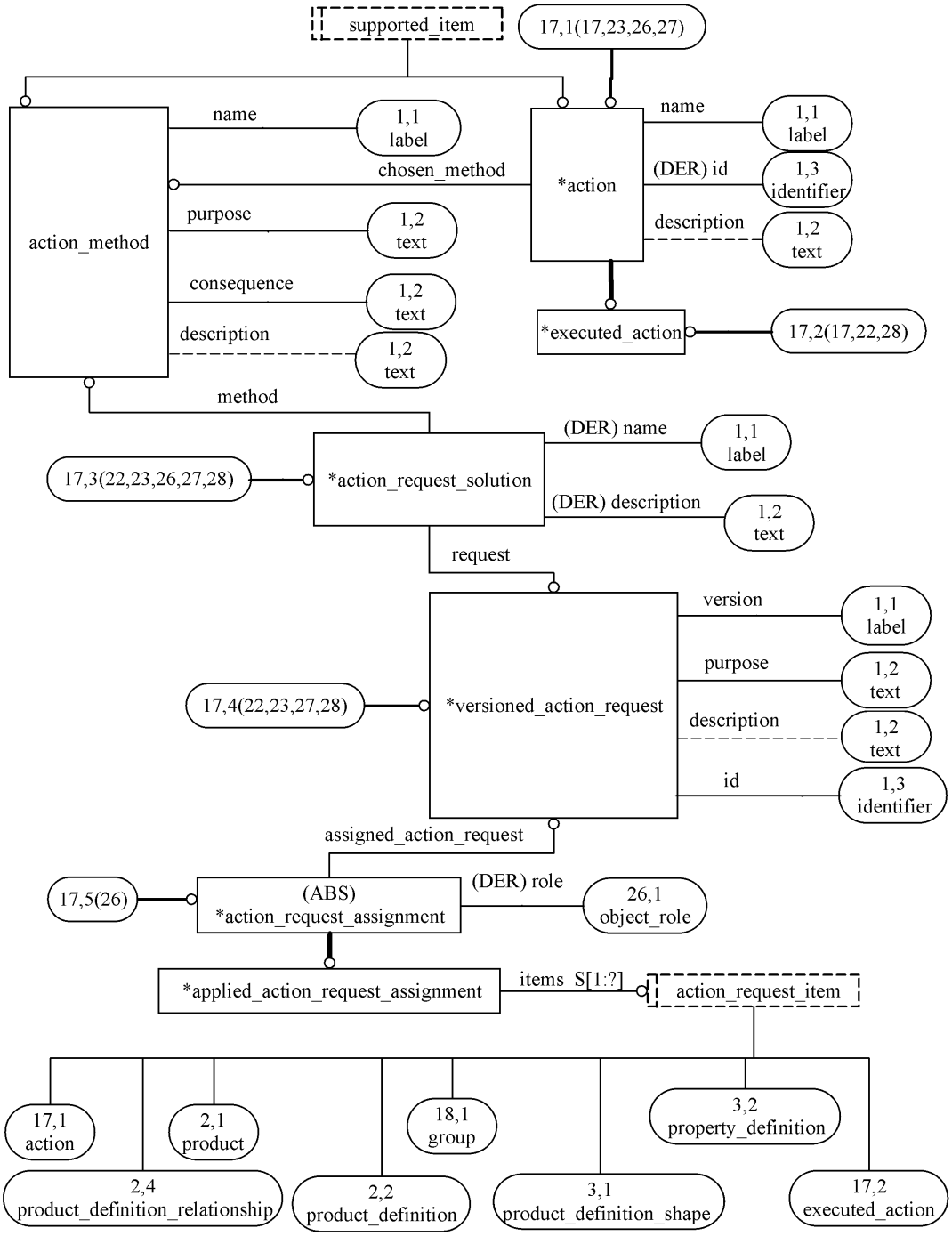


Figure H.17 — AIM diagram 17 of 28 in EXPRESS-G

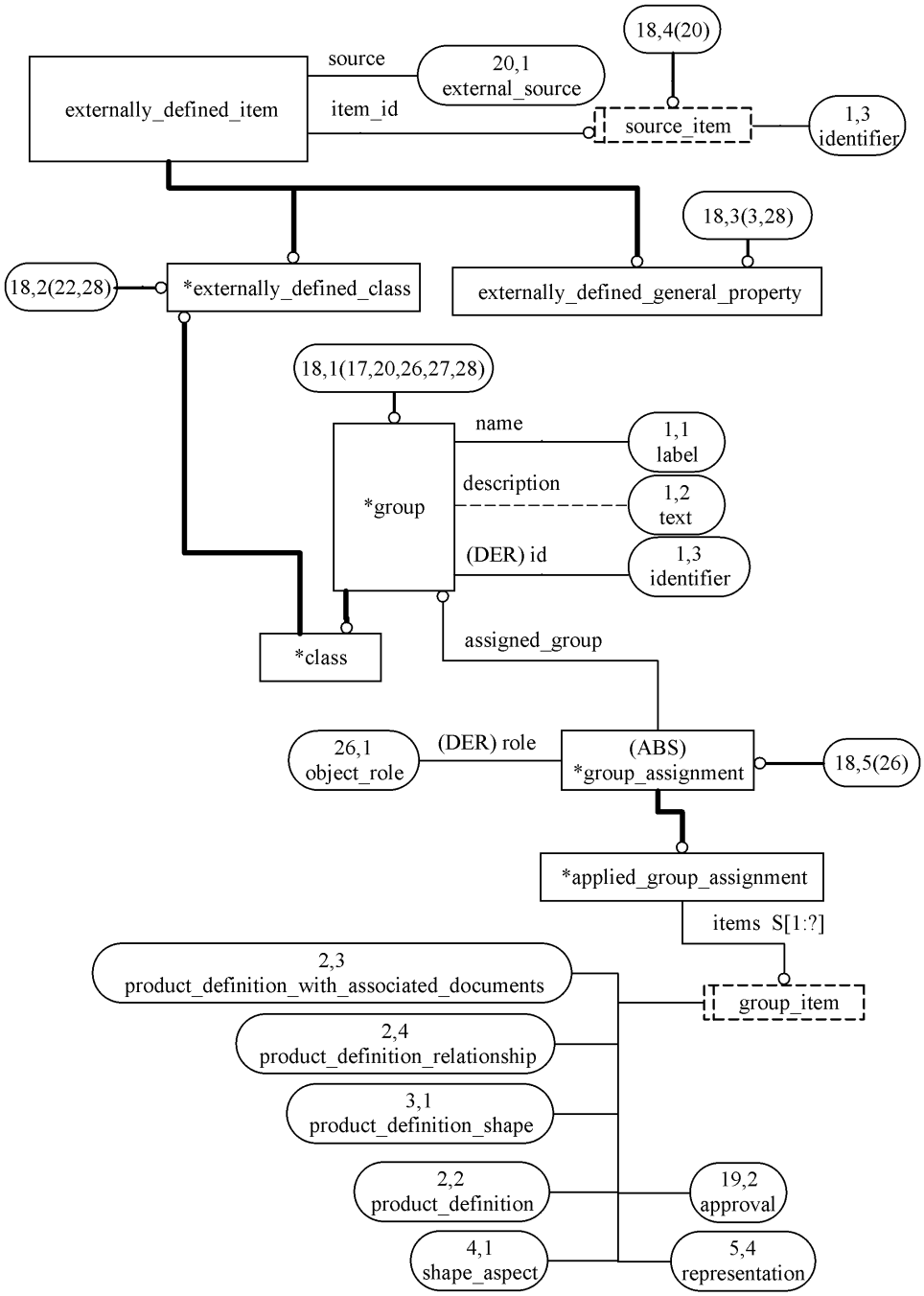


Figure H.18 — AIM diagram 18 of 28 in EXPRESS-G

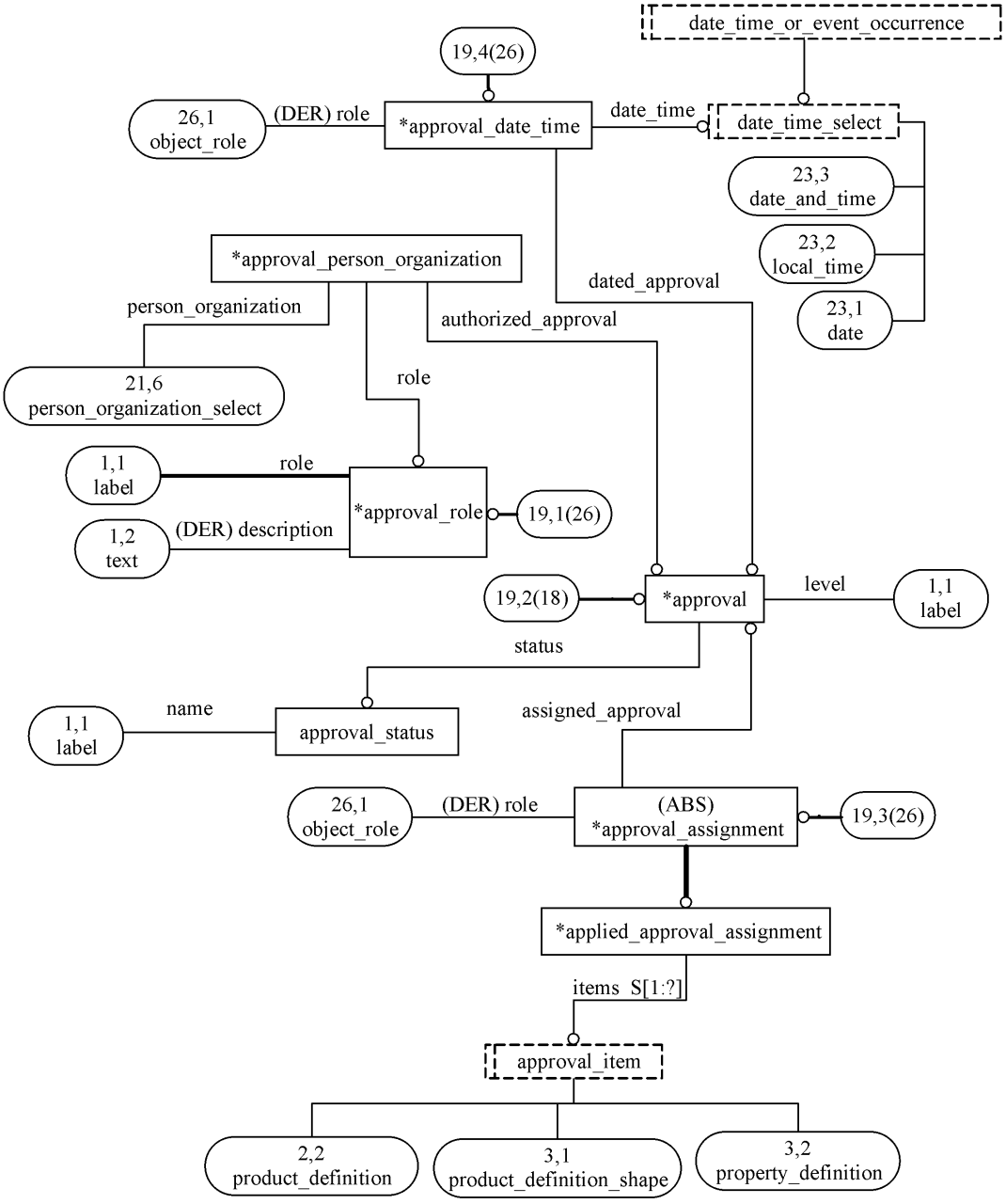


Figure H.19 — AIM diagram 19 of 28 in EXPRESS-G

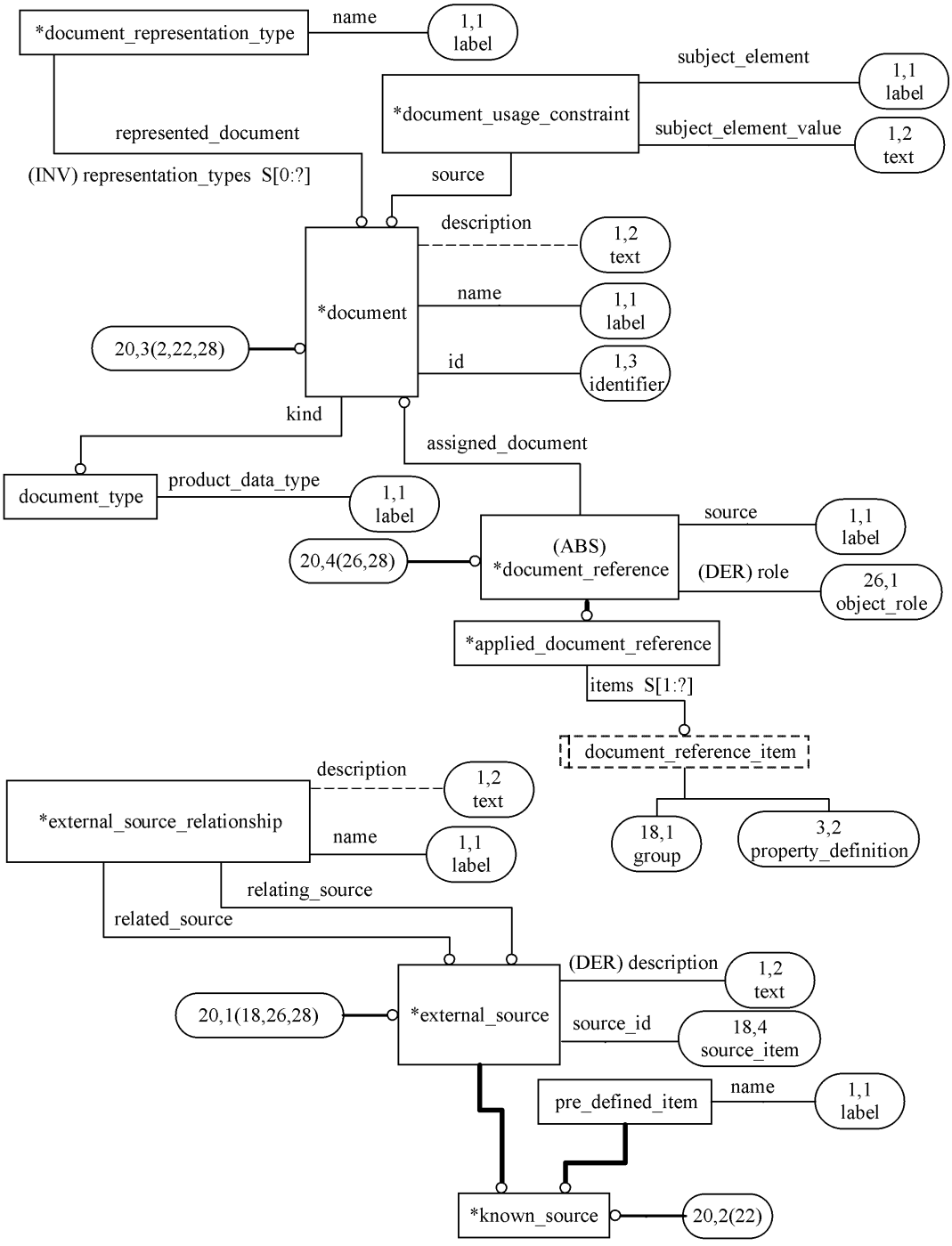


Figure H.20 — AIM diagram 20 of 28 in EXPRESS-G

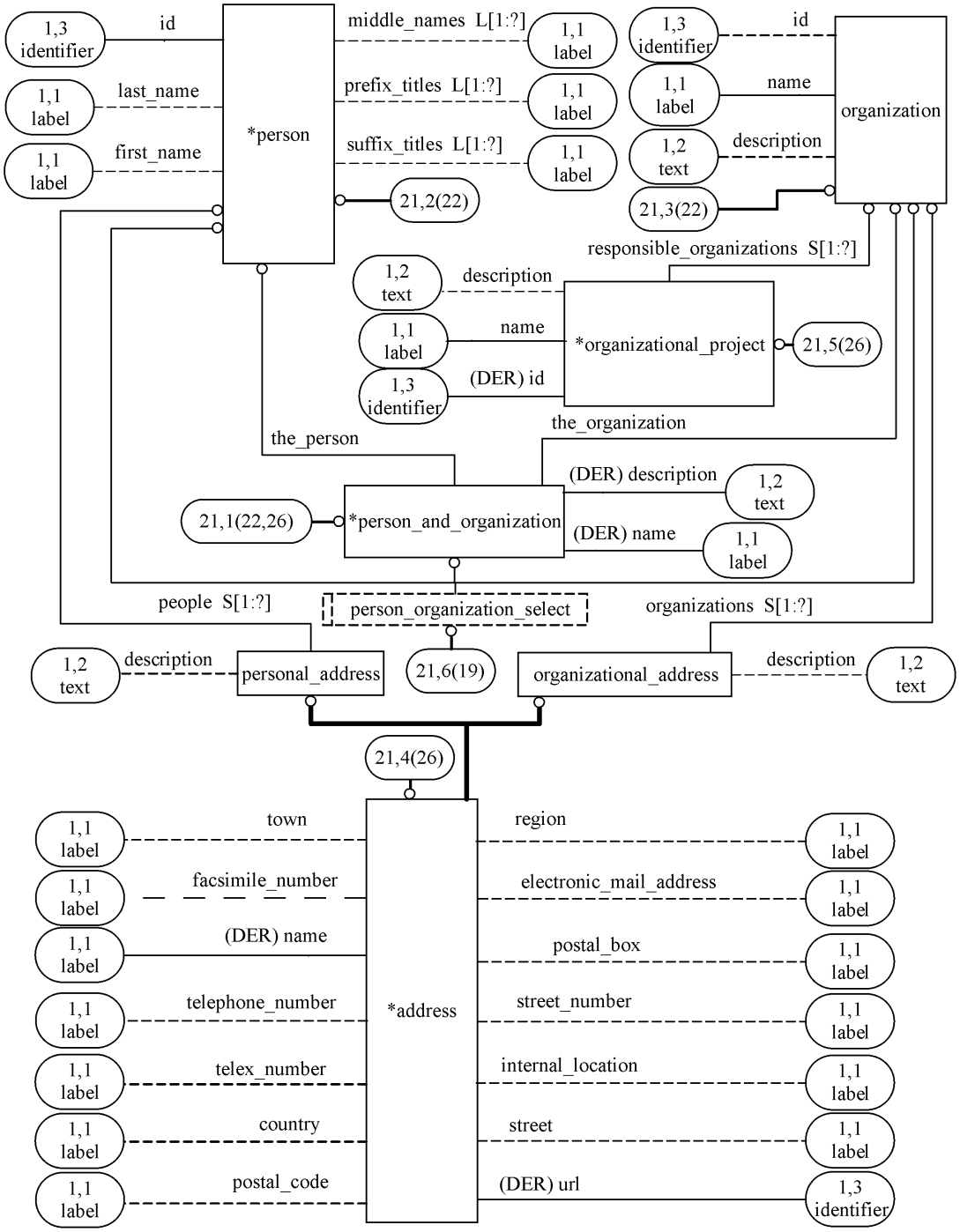


Figure H.21 — AIM diagram 21 of 28 in EXPRESS-G

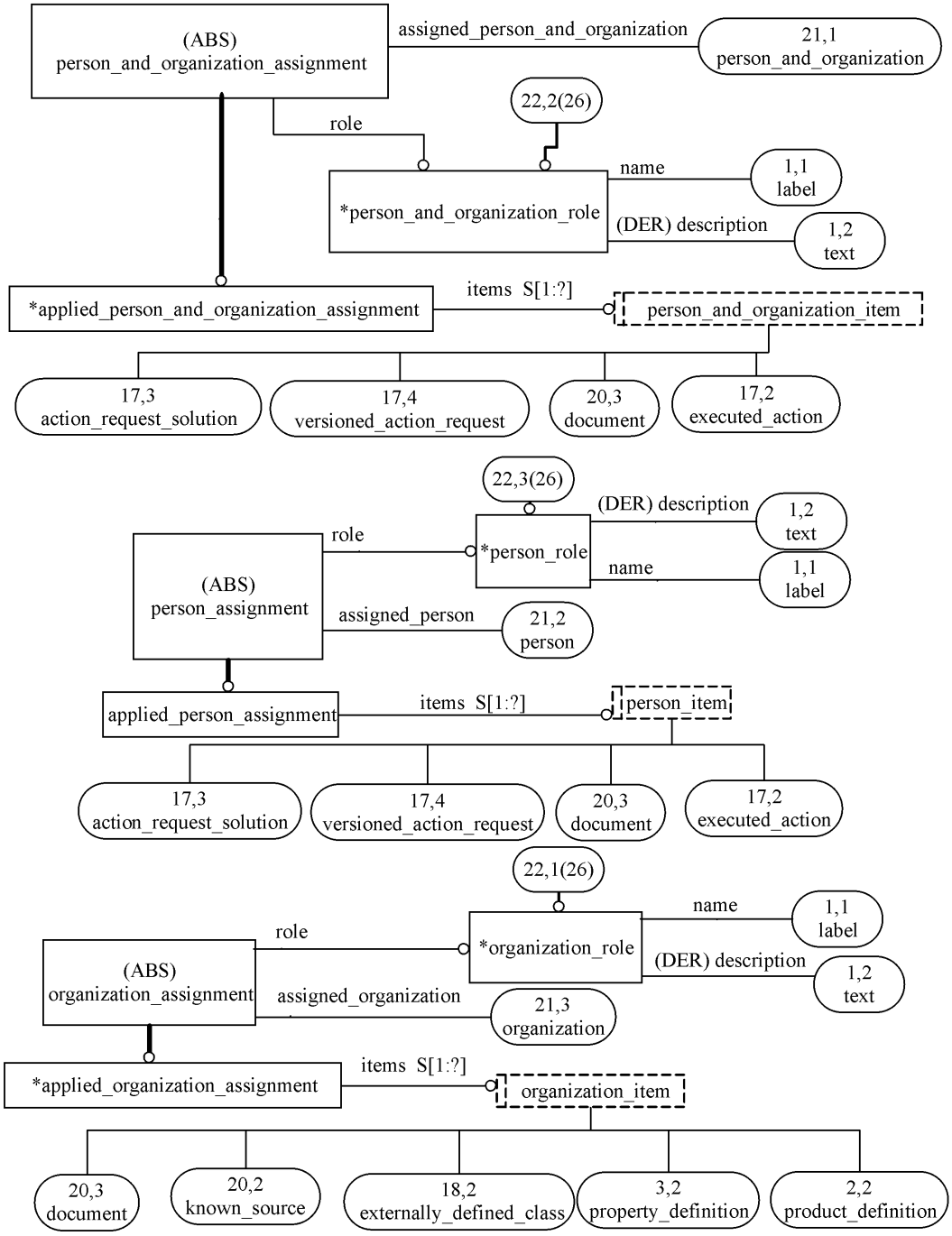


Figure H.22 — AIM diagram 22 of 28 in EXPRESS-G

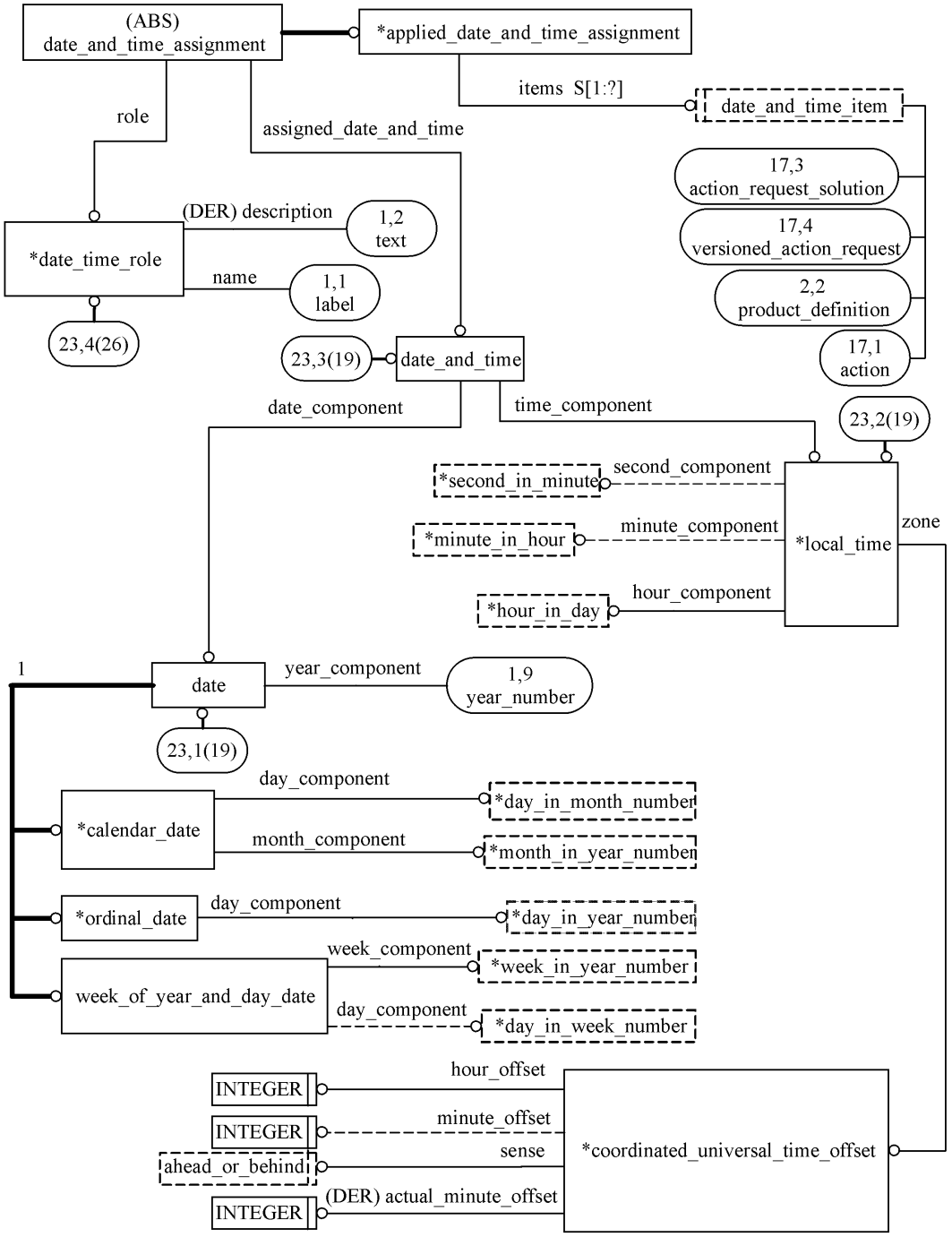


Figure H.23 — AIM diagram 23 of 28 in EXPRESS-G

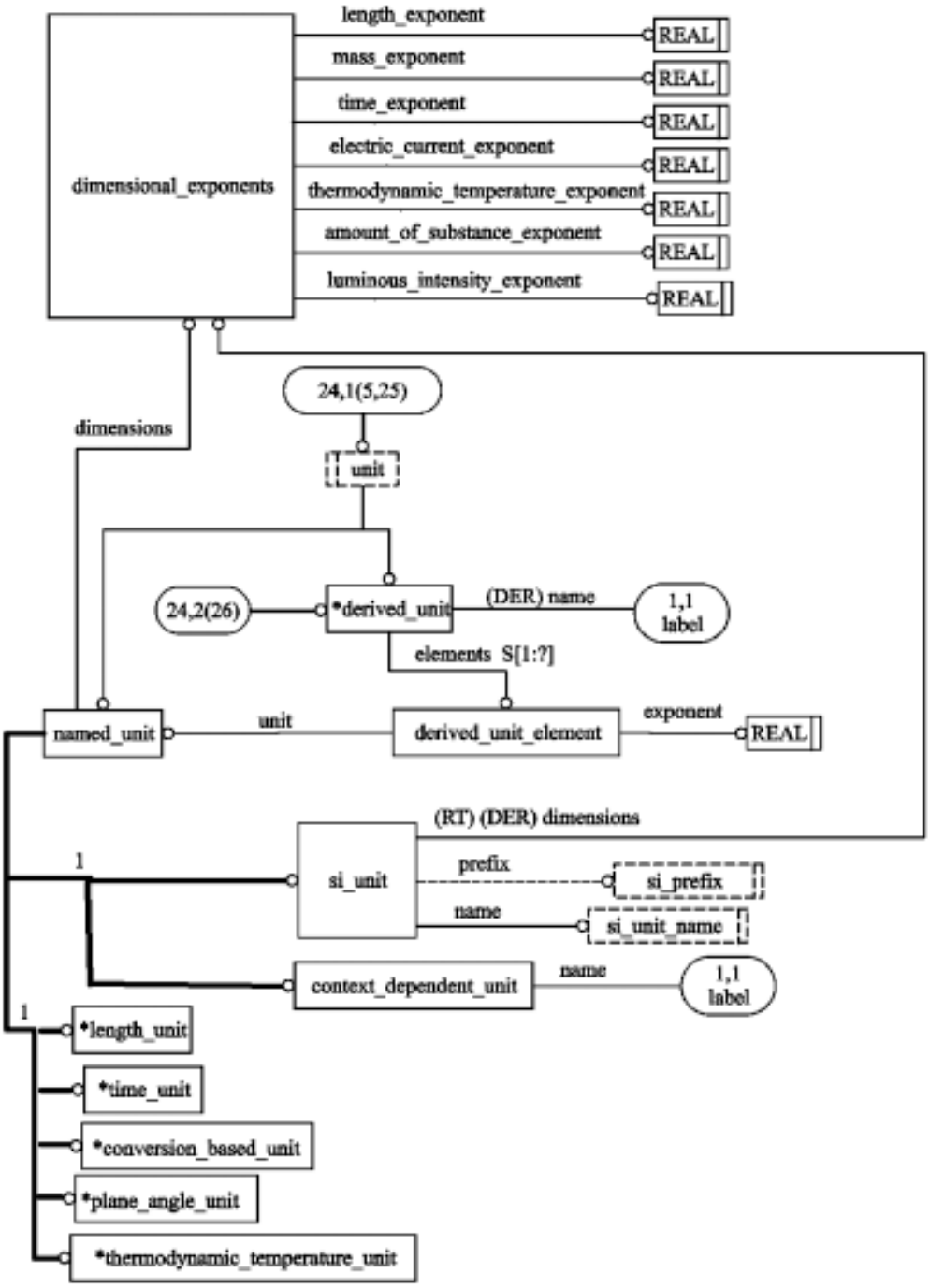


Figure H.24 — AIM diagram 24 of 28 in EXPRESS-G

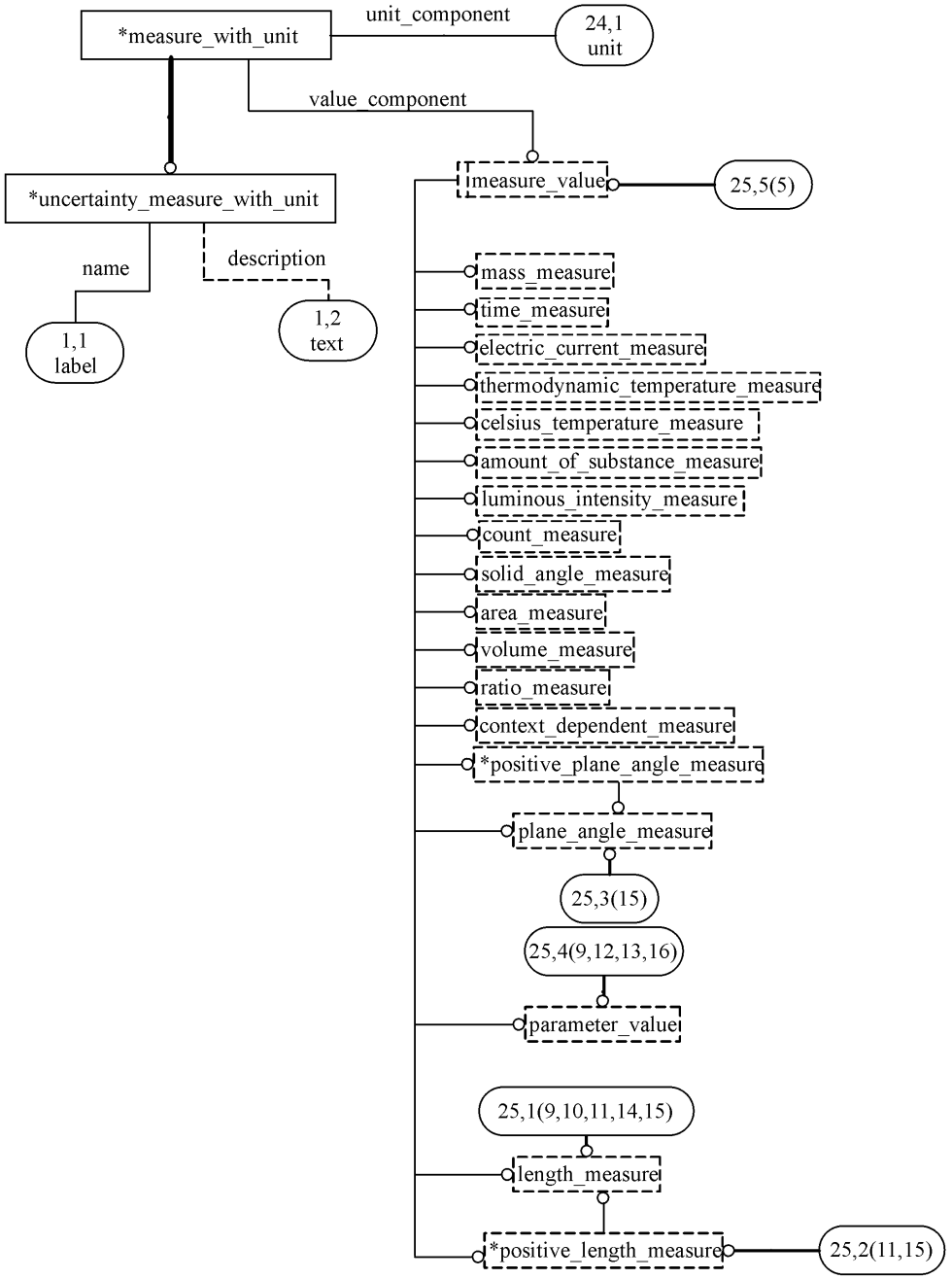


Figure H.25 — AIM diagram 25 of 28 in EXPRESS-G

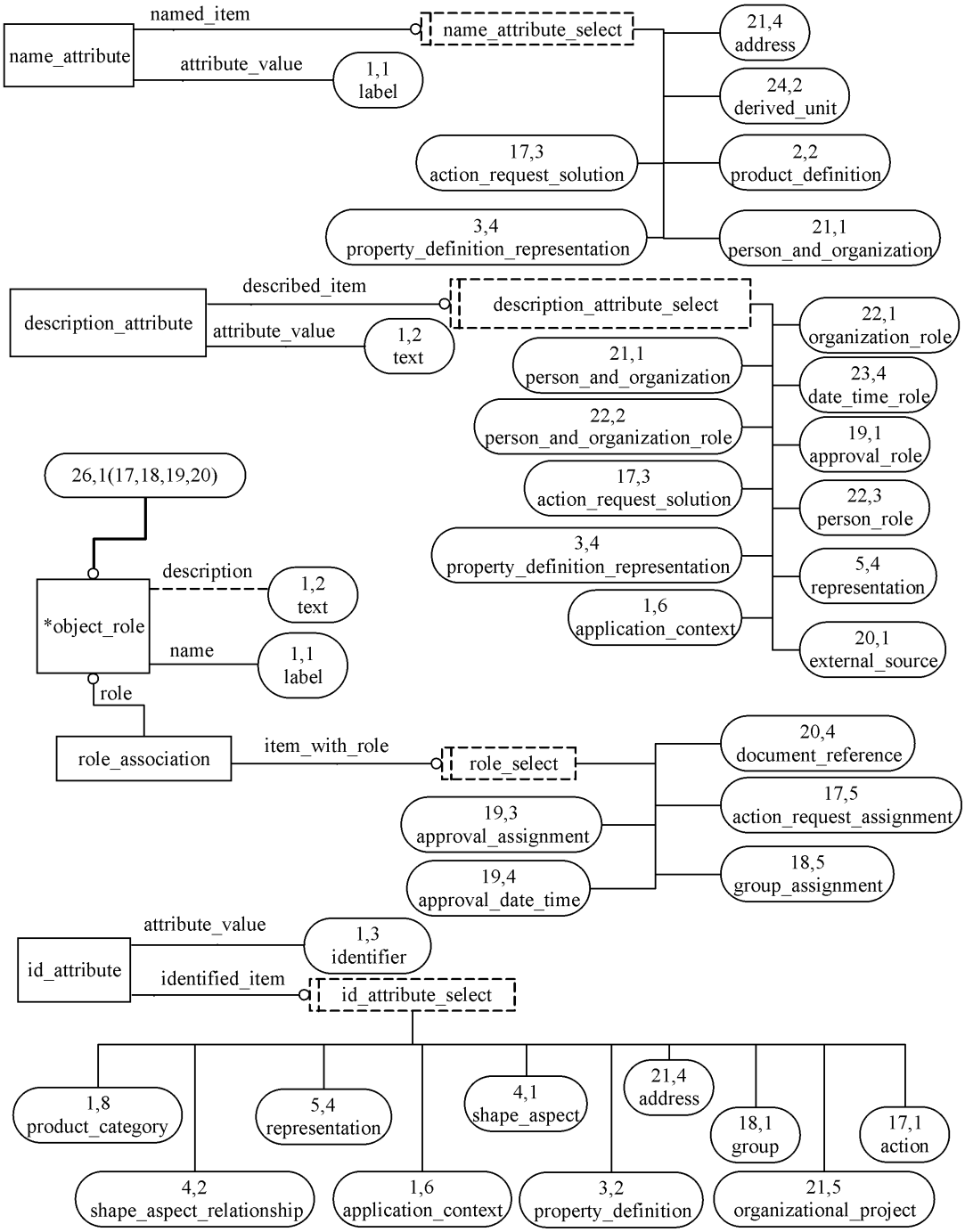


Figure H.26 — AIM diagram 26 of 28 in EXPRESS-G

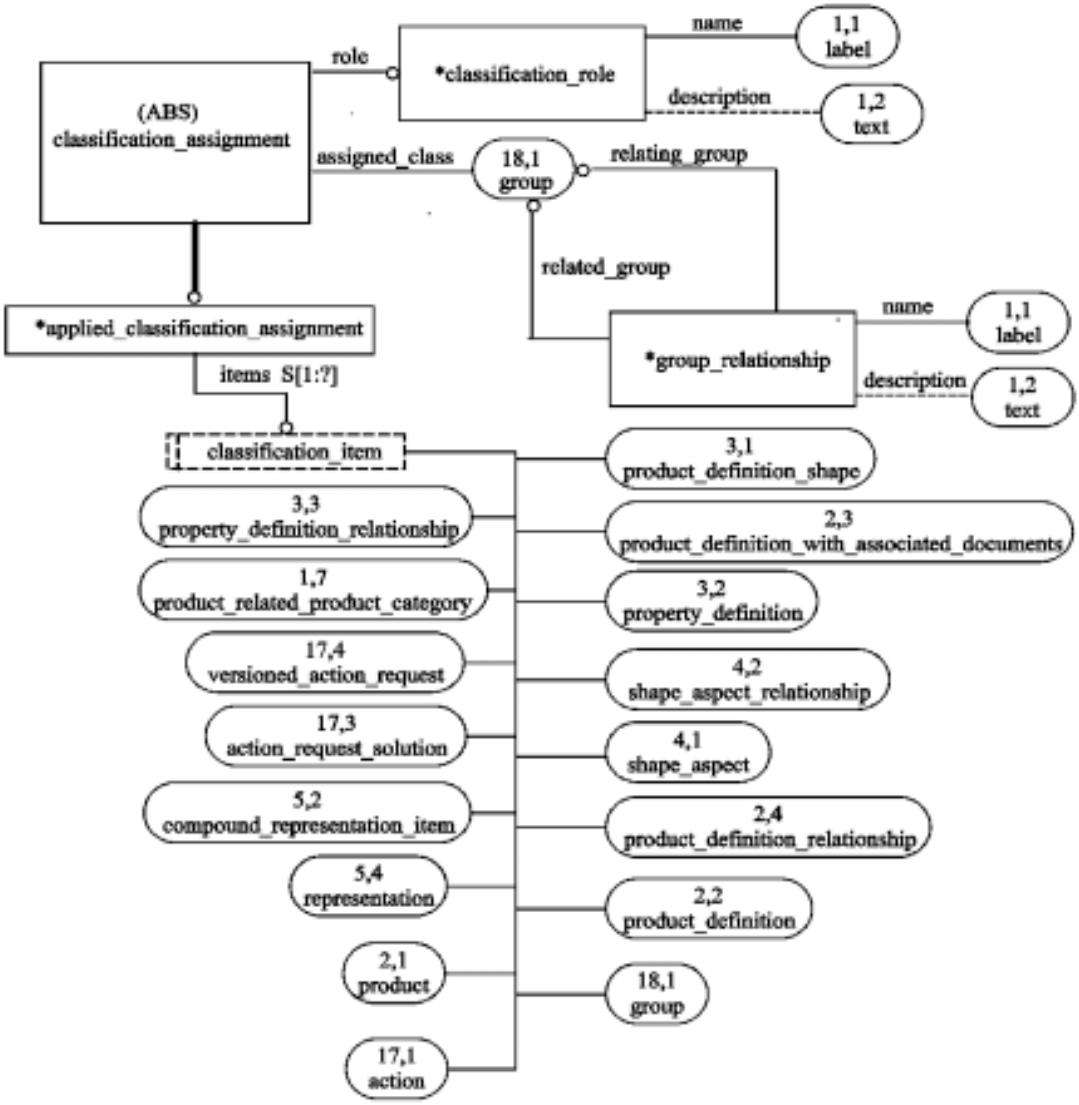


Figure H.27 — AIM diagram 27 of 28 in EXPRESS-G

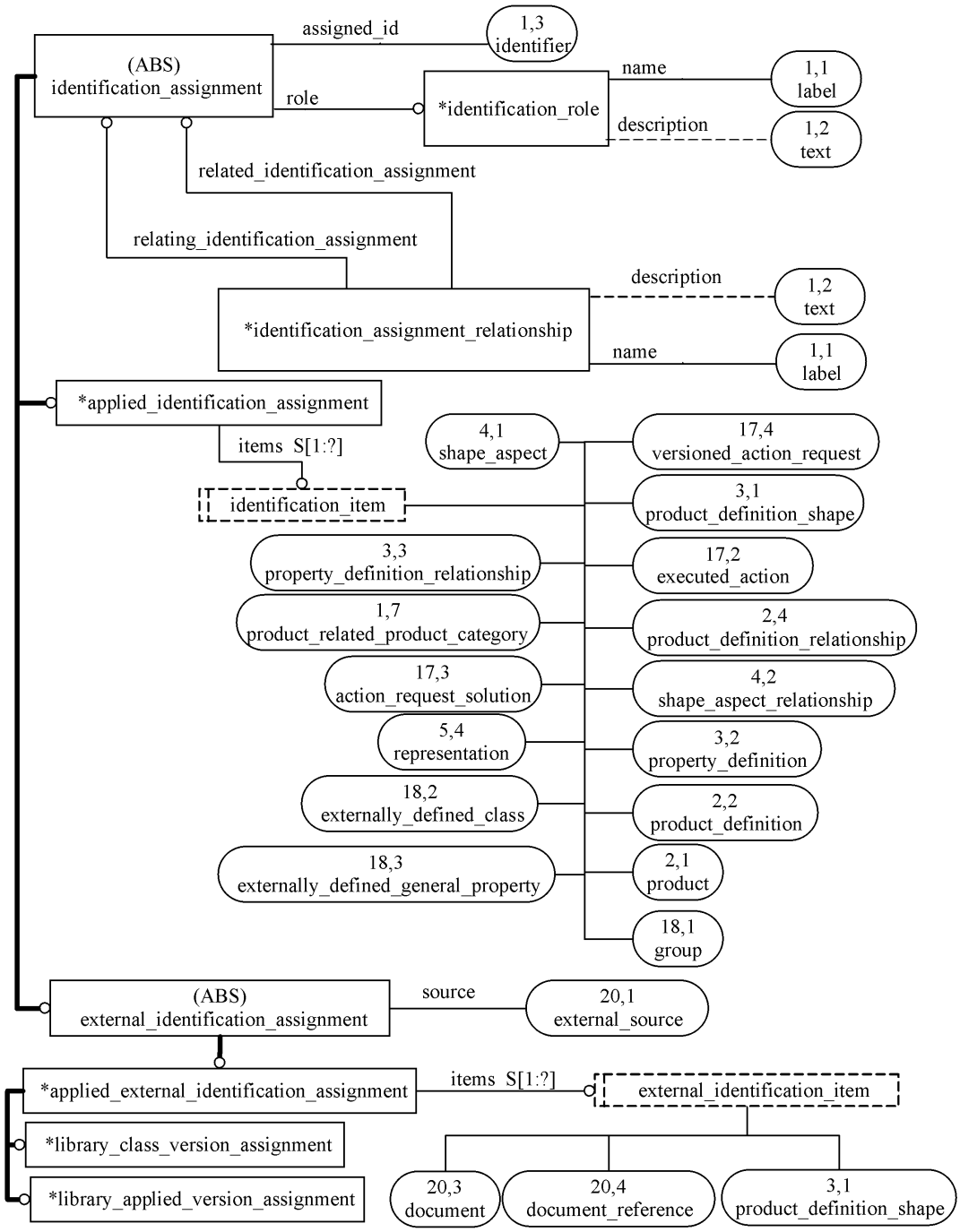


Figure H.28 — AIM diagram 28 of 28 in EXPRESS-G

Annex J
(informative)
Computer interpretable listings

This annex provides a listing of the complete EXPRESS schema specified in annex A of this part of ISO 10303 without comments or other explanatory text. It also provides a listing of each EXPRESS schema specified in this part of ISO 10303 without comments or other explanatory text. These listings are available in computer-interpretable form and can be found at the following URLs:

Short names: http://www.tc184-sc4.org/Short_Names/

EXPRESS: http://www.tc184-sc4.org/EXPRESS/

If there is difficulty accessing these sites contact ISO Central Secretariat or contact the ISO TC 184/SC4 Secretariat directly at: sc4sec@tc184-sc4.org.

NOTE - The information provided in computer-interpretable form at the above URLs is informative. The information that is contained in the body of this part of ISO 10303 is normative.

Annex K
(informative)

Application protocol usage guide

The usage guide is documented in the ESTEP AP 218 Test Case Definitions [3]. These guidelines describe the usage scenarios, and example test cases.

Annex L
(informative)

Technical discussions

L.1 Introduction

In view of the complexity of a ship it is necessary to subdivide the product model for the ship into distinct functional areas, allowing for partition into a fixed number of Application Protocols. The current version of the ship product model is shown schematically in Figure L.1.

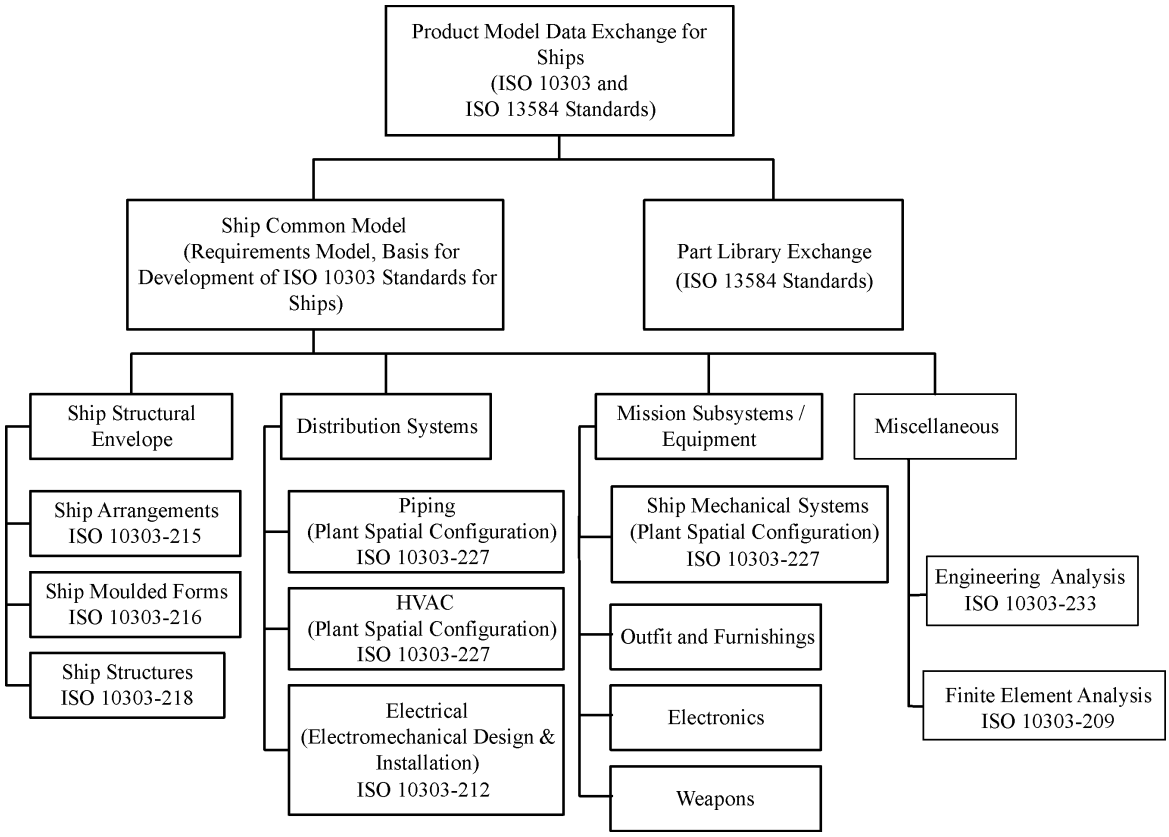


Figure L.1 — Ship product model

The key elements of the ship product model are:

— Arrangements;

— Moulded forms;

— Mechanical systems (machinery, propulsion, cargo handling);

— Structures;

— Distribution systems (piping, HVAC, electrical, hydraulics/pneumatics);

— Outfitting \& furnishings;

— Communication;

— Combat systems;

— Navigation;

— Operation.

Each functional area of the ship product model is described by one or more different application protocols. The currently available shipbuilding application protocols under development at the ISO level are:

— AP 215 Ship Arrangements;

— AP 216 Ship Moulded Forms;

— AP 218 Ship Structures;

— AP 227 Plant Spatial Configuration (used for Ship Piping and HVAC).

The subdivision of the entire ship product model allows a distributed modeling work. It is also possible to start the modeling work with the functional areas reflecting the early design stages of the life cycle of the ship and validate these models before starting the modeling work at areas in the later life cycle stages.

Each shipbuilding AP covers only a part of the ship product model. The consequence of this is that if the APs should work together as an entire product model for the ship then an overall mechanism, around which all the shipbuilding APs can be integrated, has to be defined. Only then the product model can be implemented in a product data management system, where all data are hold outside of application systems in the neutral data format of the ship product model. This mechanism for integrating different APs is called the Ship Common Model, which is described in the clause L.2.

ISO 10303-218 was originally developed by the European MARITIME and SeaSprite Projects, and supported by the Marine E-Business Standards Association in cooperation with shipbuilders, designers and software developers world wide. Its completion and approval have been led by the South Carolina Research Authority (SCRA) and the HARVEST Project of the National Shipbuilding Research Program (NSRP) in the United States. The intent of this application protocol is to support specific application involving the ship hull structures data common in various stages of the ship life cycle (see Figure L.2), especially from the preliminary design, detailed design to the manufacturing processes of ship structures.

L.2 The ship common model

The Ship Common Model defines a common framework and modeling basis for all shipbuilding APs to ensure interoperability between these APs.

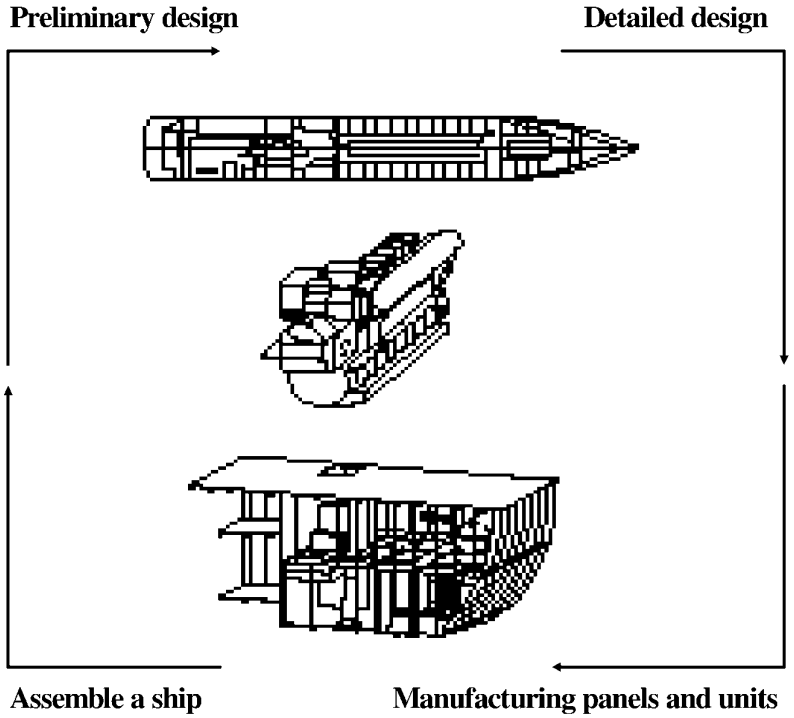


Figure L.2 — Ship life cycle

The Ship Common Model (SCM) is a set of Building Blocks which are used in the ship product model context. The SCM provides a modeling framework, a set of domain (independent and re-usable) product-structure models that are required for more then one Application Protocol, as well as a set of commonly used constructs or utilities such as those used for configuration control and management concepts. The goal of the SCM is to contribute to the integration and overall consistency of the Application Reference Models of the different ship APs.

The SCM is documented in the AP Development Guidelines for Shipbuilding [2].

L.2.1 SCM framework

The modeling framework, which is part of the Ship Common Model, provides the realization of the general concepts of how to relate things, how to define their properties and how to represent them.

This framework introduces and resides in the following Building Blocks:

— definitions;

— generic_product_structures;

— representation_resources.

Effectively, this high level approach forces the product model to be split up across the main constructs of the framework namely the definable_objects, definitions and representations whilst being linked via a number of generic relationships. One of the benefits of this approach is that it enables a better management of information such as to organize the data according to different viewpoints and in the representation of life-cycle dependent requirements. Figure L.3 shows the structure and relationship of the SCM framework elements.

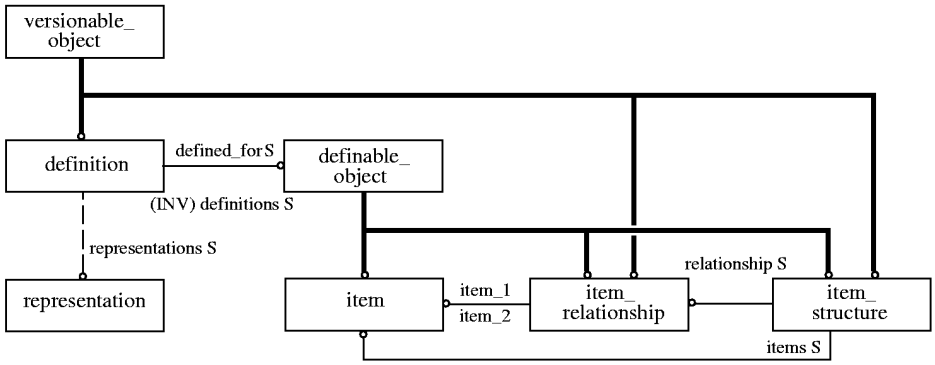


Figure L.3 — SCM framework

Fundamental to the framework is the concept of a definable_object. The definable_object is a discrete, identifiable thing used in one or more activities associated with product. It serves as the most general object from which more specialized objects are derived. A definable_object can be an item, item_-relationship or item_structure. definitions describe definable_objects and are, as a result, the descriptive information-bearing entities of the model. A definition may be further classified as a Design definition, Functional definition or Manufacturing definition, etc. A definable_object may have many different versions of definition. The definition can be changed for a definable_object during the life cycle.

Typical items in the shipbuilding APs are:

— a ship and components of the ship, such as hull, superstructure, deck, propeller, etc.;

— a part of a ship from different point of views: assembly, system, space, etc.;

— equipments, such as pump, generator, main engine, pipe, etc.;

— steel structure elements, such as plate, profile, double bottom, frame, bulkhead, etc.;

— features of steel structure elements, such holes, cutouts, endcuts, etc.;

— functional elements, such as port, logical_connections, etc.

The properties of an item are carried by the definitions. A definition must be defined for an item but an item may exist without any definition. Every property of a concept and therefore, every definition of an item, may be described in many different ways. Thus a definition can have different representations, but in some circumstances there can be a definition without representation. New representations in the shipbuilding APs can be created by subtyping them from representation. The high level relationships between the main constructs of the modelling framework can be restricted in the subtypes of each AP. This is done by re-declarations of the attributes.

L.2.2 Domain models

The domain models provide a set of templates for organizing the product being modelled along a number of different axes or views, such as product by assembly, by system or by space. However, the templates also provide a set of implicit modelling techniques for the organization of the product data. Those domain models represent generic structures which allow the modeller to organize the data of the product according to their need. The benefit of this approach is that it reduces the modelling effort, allows consistency, conformity and interoperability with the other APs that already conform to this approach.

The intention of domain models is to create a new layer of generic elements underneath the main constructs of the modelling framework, with the possibility to restrict the attribute relations using re-declarations. These new elements introduced by the domain models are subtypes of the main constructs of the modelling framework.

The following domain models belong to UoF product_structures of the Ship Common Model:

— features;

— parts;

— product_structure_by_system;

— product_structure_by_assembly;

— product_structure_by_space;

— connection_topology.

L.2.3 Common utilities

Common utilities are a group of constructs that will be required by most shipbuilding APs. The utilities differ from the framework and domain models through the fact that the majority of cases, the utilities are ready for use and do not require any further specialization for use in an ARM. Many have been created specifically for shipbuilding although some may be able to be used externally.

Common utilities are currently available for:

— ships and ship types;

— ship general characteristics;

— configuration management;

— location concepts;

— basic geometry and topology;

— materials;

— measures and units;

— external references.

L.3 Key concepts of ship structures

L.3.1 Items and definitions of part 10303-218

ISO 10303-218 is being developed to support the exchange and sharing of ship structures data. This part has been developed in conjunction with other ship product application protocols so that important shipbuilding concepts are common and interoperable within the STEP parts.

The ship structures product model provides a set of subtypes of item for describing the ship structures are listed in Figure L.4. It includes ship, parts as well as their connections, systems, assemblies and hull_cross_sections as they are the essential subtypes employed in this part.

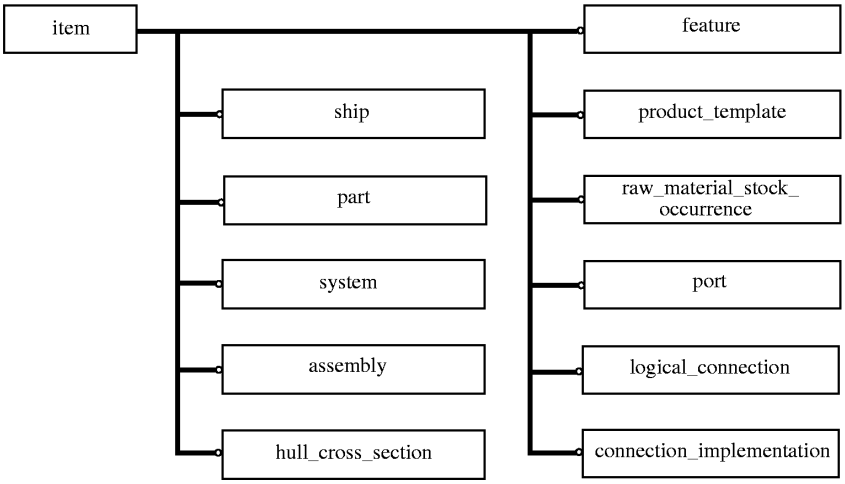


Figure L.4 — Subtypes of items for ship structures

The subtypes of Definitions for this part are listed in Figure L.5. This set of the definition sybtypes serves to describe the design and manufacturing data of ship structures, including the necessary locational and functional information.

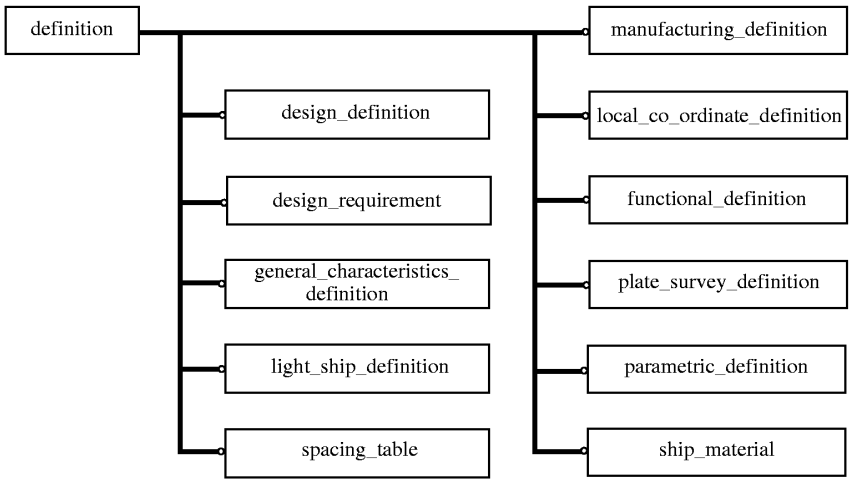


Figure L.5 — Subtypes of definitions for ship structures

The concept of item in this part is used via a hierarchical partitioning.

The Definition for the structures may be classified as a Design_definition, which is a main definition part for the preliminary and detailed design of the ship lifecycle stages. The main components of the Design_definition types, such as design definitions for structural parts, structural part features, structural part joints and structural systems, etc. employed in this part of ISO 10303 are shown in Figure L.6.

L.3.2 Structural systems

Structural_system is a specific System of this part 10303-218. It provides information and capabilities common to all types of **Structural_system**s. The **Items** carried by a **Structural_system** are restricted to be of type **Structural_system** or **Structural_part**. The same holds for the relationships.

A **Structural_system** is of a specific class, as specified by the class attribute:

— **SECTION:** an aggregate of other (sub)sections, blocks or **Structural_part**s; has both a strength **AND** a tightness function

— **PANEL_SYSTEM:** special section that consists of zero to many plates reinforced by profiles running in the same main direction; the plates lie on the same moulded surface (plane or non-plane)

— **CORRUGATED_STRUCTURE**: special section that consists of only **Corrugated_part**s and **Plate**s

— **BLOCK**: also referred to as constructional group; an aggregate of other (sub)blocks or a tightness function

— **PLATE_STRAKE**: special block that consists of only plates; has a tightness function

— **BUILT_PROFILE**: a **Structural_system** that is made of at least two **Structural_part**s (such as **Plate**s or **Profile**s) by an aggregation process; can nevertheless be used like a profile

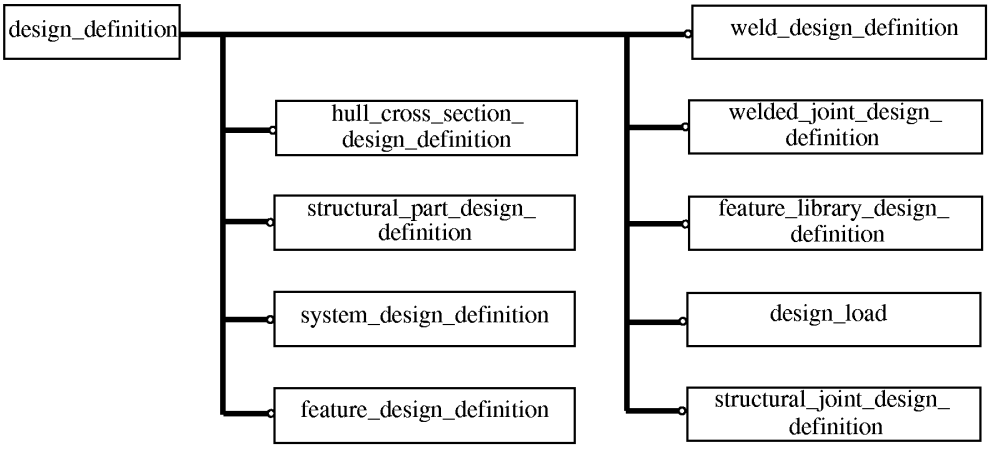


Figure L.6 — Subtypes of Design_definition for ship structures

L.3.3 Corrugated parts and systems

Corrugated structures required for this AP218 are:

— the structures have not only rectangular overall shape, such as tank tops may have rises of a floor, bulkheads may have waved plates;

— corrugations run along one axis, linearly extruded along the other;

— shape expressed using a set of parameters;

— the corrugations support for assembled purchased structures and those purchased section-wise and welded together;

— they are bounded or trimmed by other structural elements.

Figure L.7 is EXPRESS diagram of the corrugated|_part model.

Figure L.8 is EXPRESS diagram of the corrugated|_system model.

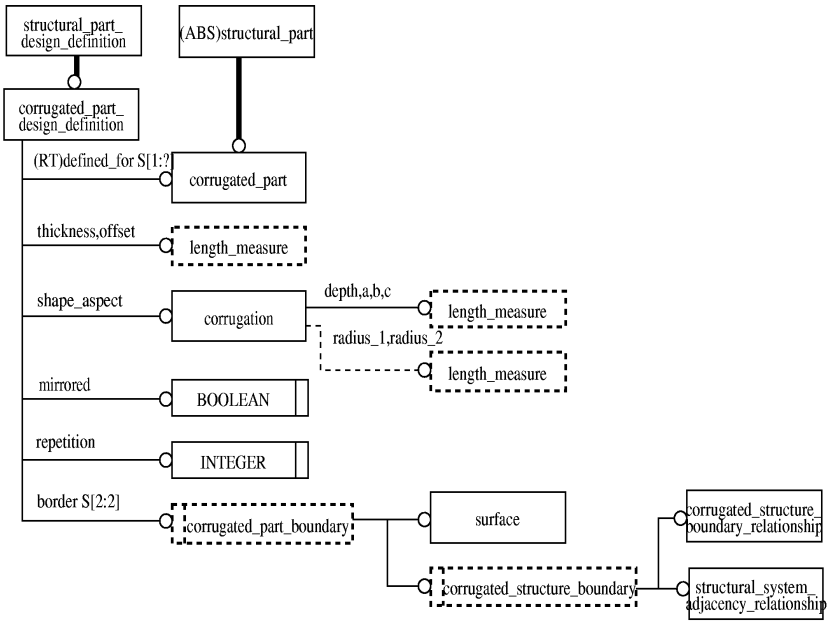


Figure L.7 — Corrugated part model

L.3.4 Structural features

Structural_features specifies the concepts for the definition of Features that are applied to Structural_-parts and Structural_systems.

The shape of a Structural_feature may be represented in one of the following ways:

— by explicitly describing the shape using Explicit_feature_design_definition (BB feature_design_-definitions);

— by referring to an entry in a features library using Feature_library_definition (BB feature_-design_definitions);

— by describing the shape parameters using one of the feature type specific Design_definitions; these are:

  — Free_form_interior_cutout_design_definition;

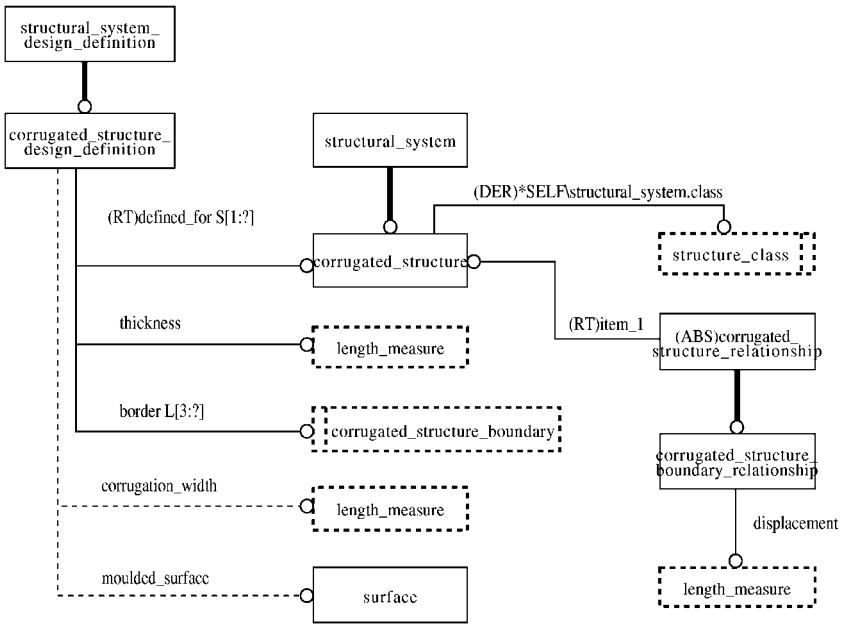


Figure L.8 — Corrugated system model

— Circular_cutout_design_definition;

— Elliptical_cutout_design_definition;

— Elongated_oval_cutout_design_definition;

— Rectangular_cutout_design_definition and its subtypes;

— Triangular_cutout_design_definition and its subtypes;

— Part_cutout_design_definition;

— Drain_hole_cutout_design_definition;

— Round_corner_design_definition;

— Inward_round_corner_design_definition;

— Outward_round_corner_design_definition;

— Beveled_corner_design_definition;

— Shear_beveled_corner_design_definition;

— Rectangular_cutback_corner_design_definition.

No reference to moulded geometry is provided, except via the parent of the Feature if this link exists (via the corresponding relationship-entities) and if the parent references underlying geometry.

A hierarchy of feature design definitions can be shown in Figure L.9.

L.3.4.1 Beveled_corner_cutout

A beveled_corner_cutout (or chamfer) denotes the surface obtained when intersecting a rectangular work item with a plane having a normal vector not parallel to any of the adjacent surfaces of the work item. The origin of the local coordinate system is placed at the intersection of the work item boundary lines at the corner that is to be cut-off. The axis of the local coordinate system coincides with the work item edges and point such that the cut-off occurs always in positive axis direction and such that offset and the angles are measured along and from the x-axis.

— Beveled_corner_cutout: The normal vector of the bevel surface is perpendicular to the normal vector of one adjacent work item surface.

— Shear_beveled_corner_cutout: The normal vector of the bevel surface is not perpendicular to any of the normal vectors of adjacent work item surfaces.

L.3.4.2 Circular_cutout

A circular_cutout is an circular opening in a work item specified by a radius "r". The mathematical description is given by: x2 + y2 = r2. The local origin coincides with the centre of the circle.

L.3.4.3 Elliptical_cutout

An elliptical_cutout is an opening in a work item described by an ellipse. The mathematical description of the ellipse is given by: (x/a)2 + (y/b)2 = 1. The local origin is placed at the centre of the circle and the coordinate axis (x-axis and y-axis) coincides with the ellipses half axis as shown below. "a" and "b" are the half axis parameter.

L.3.4.4 Elongated_oval_cutout

An elongated_oval_cutout is an opening shaped like an egg or a slot. The basic form consist of two circles, "r1" and "r2", separated by a "distance" and joined by an envelope. The default envelope consist of two straight lines joining the two circles. If the two circles have the same radius the opening is a slot. By specifying an additional circle arch (i.e. radius "r3" and arch centre coordinates "r3_x" and "r3_y") the opening obtains an egg shaped form. The local origin is placed at the centre of the left circle. The local x-axis is given by the line through the centres of the two basic circles (= symmetry axis).

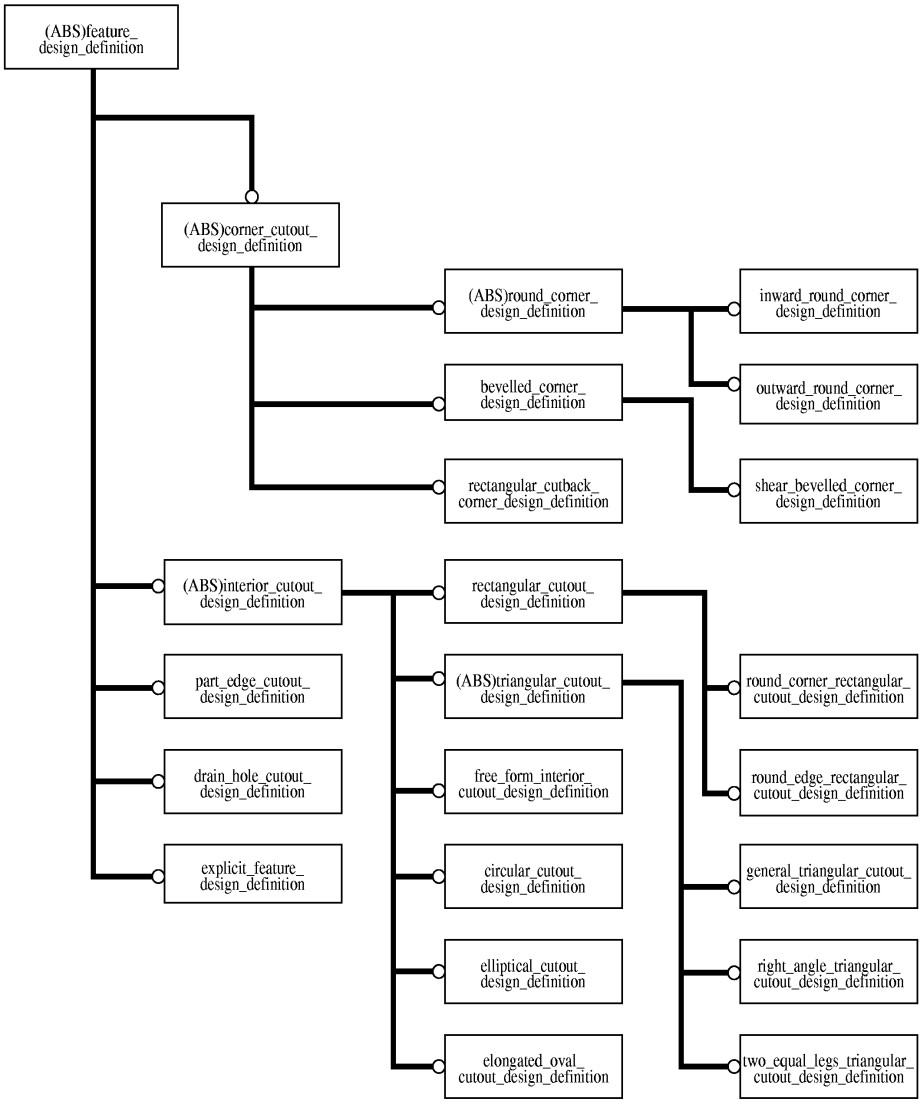


Figure L.9 — Hierarchy of feature design definitions

L.3.4.5 Rectangular_cutout

A rectangular_cutout is an opening where the opposing sides of the opening are parallel and adjacent sides are perpendicular. The interior angles of the opening are 90 degree. A "width" and "length_of" specify the rectangular opening. The local origin is placed at the intersection of the symmetry axis of the opening.

L.3.4.6 Round_corner_rectangular_cutout

A round_corner_rectangular_cutout is an rectangular_opening where the corners of the opening are rounded off. The rectangular opening is specified by a a "width", "length_of" and the corner radius "r". The local origin is placed at the intersection of the symmetry axis of the opening.

L.3.4.7 Round_edge_rectangular_cutout

A round_edge_rectangular_cutout is an rectangular_opening where two opposing sides of the opening are replaced by circle arches. A "width", "length_of" and the circle arch radius, "r", and its centre specify the rectangular opening. The local origin is placed at the intersection of the symmetry axis of the opening. The centre of the circle arches lies on the x-axis given by "distance" which can have a positive or negative value.

L.3.4.8 General_triangular_cutout

A general_triangular_cutout has no symmetry axes, no right interior angle. The origin is placed at corner point c1, and corner points c1 and c2 define the x-axis. y-axis is perpendicular to the x-axis and such that corner point c3 has a positive y-coordinate.

L.3.4.9 Right_angle_triangular_cutout

A right_angle_triangular_cutout is an opening with three corners and where one of the corners has an interior angle of 90 degrees. The local origin is placed at this corner and x- and y-axis point to the other two corners. A "height" and a "width" specify the triangular opening.

L.3.4.10 Two_equal_legs_triangular_cutout

A two_equal_legs_triangular_cutout is an opening with three corners and where two of the sides have the same length. The side being not equal to any other coincides with the specified by a "height" and a " width".

L.3.4.11 Drain_hole_cutout

A drain_hole_cutout is a symmetric oblong opening with circular ends on both sides and a break-through along the symmetry line to the exterior of the work item. The drain holes symmetry axis is perpendicular to the work item boundary. It points into the interior of the work item. The origin of the local coordinate system is placed at the intersection of the work item boundary line and the symmetry axis. All length measures (Drain hole width, depth, Drain_hole_radius, Gap and Gap_radius) are always positive. Depth is measured parallel to the symmetry axis whereas Width and Gap is perpendicular to it (see Figure 17).

L.3.4.12 Part_cutout

A part_cutout is an opening (not hole) at an edge of a work item that is caused by a part (such as profile, pipes). The shape of the opening is therefore part dependent and is not necessarily symmetric. All measures of the part_cutout are relative to the part, that is, they depend on the part ID-points. The local x_axis coincides with the work item edge whereas the direction of the y_axis is perpendicular to it. The part_cutout depth and inner shape depends on the part ID locations B and C. The width of the opening may depend on the part ID locations A, B, C, or D. The radius (R_left, R_right), x_A and x_D are positive always, whereas x_B, y_B, x_D and y_D may be positive or negative.

L.3.4.13 Rectangular_cutback_corner_cutout

A rectangular_cutback_corner_cutout is obtained by removal at a work item corner resulting in an inward-rectangular cutout which sides are parallel to the original corner sides. The local origin is placed at the corner to be cutout and the local coordinate axes are such that the work item is in positive direction. All length measures (x_depth, y_depth and z_depth) are always positive and are measured along the x-, y- and z-axis respectively . More complicated cutbacks may be achieved by combining different corner cutbacks such as beveled_corner_cutout or round_corner_cutout (see Figure 37).

L.3.4.14 Round_corner_cutout

A round_corner_cutout denotes the circular arc (cylindrical surface) between two adjacent surfaces, also called fillet. The origin of the local coordinate system is placed at the intersection of the work item boundary lines at the corner that is to be cut-off. The axis of the local coordinate system coincides with the work item boundary lines and point such that the cut-off occurs always in positive axis direction. The radius is always given in positive length measures.

— Inward_round_corner_cutout;

— Outward_round_corner_cutout .

L.3.4.14.1 Inward_round_corner_cutout

The fillet centre is located outside the finished work item, that is, the fillet describes a inward curve. The offsets (x-offset along x-axis, y-offset along y-axis) can be positive or negative. The offsets specify the distance of the fillet centre from the local origin, see Figure 27.

L.3.4.14.2 Outward_round_corner_cutout

The fillet centre is located inside the finished work items, the fillet describes a outward curve. The offsets (x-offset along x-axis, y-offset along y-axis) are always positive. The offsets specify the distance of the fillet centre from the local origin, see Figure 29.

L.4 Globally unambiguous identifier usage

Globally unambiguous identifiers (GUID) have been incorporated as part of the Ship Common Model and are used by all the Shipbuilding Application Protocols. They are used to support the following incremental exchange scenarios:

— Separate exchanges of items, definitions, and representations within an AP while preserving the relationships between these application objects. For example, the first exchange may only identify the moulded_forms, by sending only Moulded_form entities, while the second exchange may contain the specific properties and representations, by sending Moulded_form and Moulded_form_design_definition entities.

— Separate exchanges of a single ship model using AP 216 while preserving relationships between selected AP 216 application objects. For example, the first exchange may include the hull Moulded_form entity and the second include the main deck Moulded_form entity whose boundary is defined by an External_instance_reference entity containing the globally unambiguous identifier of the hull.

— Separate exchanges of a single model using several shipbuilding APs while preserving relationships between application objects in different APs. For example, the first exchange may include a deck Moulded_form entity in AP 216 and the second include the main deck Plates in AP 218 whose underlying surface is defined by an External_instance_reference entity containing the globally unambiguous identifier of the deck moulded_form in AP216.

Globally unambiguous identifiers and incremental exchanges are important for shipbuilding for the following reasons:

— Volume of data in a ship design. It is simply not practical to exchange an entire ship design in one exchange. Typical data exchanges focus on specific areas of the ship or specific systems. It is important that the result of these multiple exchanges results in a ship model which preserves the basic relationships between shipbuilding elements.

— Long design and production time. A ship design is developed over an extended period, as long as several years for military ships. Similarly, the production information is also developed over a similar period. It is important to exchange data at any point in the design or production cycle while preserving the basic relationships between shipbuilding elements.

— Instance based design. A ship design is composed of individual occurrences of parts at specific locations on the ship. It is important to uniquely identify and track each individual instance on the ship through design, production, and eventually to operation.

— Hull applicability. A class of ships may be designed similar to a single ship design, with hull-specific exceptions called out. It is composed of individual occurrences of parts at specific locations on the ships. It is important to uniquely identify and track each individual instance in the ship class through design, production, and eventually to operation.

— Concurrent design and production. Typically, production of a ship begins before the entire design is complete. Both the design and production information is assembled incrementally. Further, in order to manage changes, it is important to uniquely identify each individual instance on the ship through design and production.

— Co-design. Several design agents may collaborate on the design of a ship or ship class. This is feasible only if each company can identify the incremental changes made by the other partner.

— Co-production. Several shipbuilders may collaborate on the production of a ship or ship class. This is feasible only if each company can identify the incremental changes made by the other partner.

— Collaboration. A number of companies may collaborate in the design and production of a ship or ship class. This is feasible only if each company can identify the incremental changes made by each company.

L.4.1 Functional requirements

Since GUIDs are used to support incremental exchanges, it is not possible to fully specify the functional requirements for GUIDs in the Shipbuilding Application Protocols. The functional requirements for GUIDs are as follows:

— The GUID must be globally unique and persistent, within a ship design, across ship designs and across companies. It must point to the instance that it designates or to nothing at all (as when the instance has been deleted).

— Each software system must assign persistent GUIDs to each application object created or updated.

— Each software system must support access to the application object via GUID.

— Each software system must store the GUID of all application objects.

— Each software system must return same GUID as it received if application object was not changed.

L.4.2 Implementation constraints

A number of different implementation schemes and technologies can be used to support this notion of GUID. However, there are several implementation constraints, which should be considered:

— Must support interoperability across companies, applications/systems, and technologies. (The identifier should not be opaque but must expose the essential data elements needed for persistent identification. Opaque identifiers were specified in CORBA v1 and proved an inhibitor to interoperability across systems).

— Must support systems that use value-based keys as well as systems that use system-generated object identifiers. (Should not impose the requirement on value-based systems to store an additional attribute for system-generated identifiers.)

— Instance identifiers must contain globally unique type identifiers but also have additional requirements. Consequently, instance identifiers carry more information than type identifiers (such as PLIB BSUs).

— Identifier should be character-based and the syntax should be specified so that industry standard tools and algorithms can be used.

— Identifier should support existing Part 21 and SDAI usage as well as anticipated Part 28 XML usage.

— Identifiers should be based on unique identification of single data objects in the software application. The Ship Common Model typically uses four or more objects to describe a single data object: item, design_definition, functional_definition, and manufacturing_defintion. While it is important to uniquely identify each STEP application object, it is also important to the software application to uniquely identify the data object, which generates these STEP application objects.

L.4.3 ISE recommended GUID format

The Integrated Shipbuilding Environment (ISE) project has adopted and is implementing translators based upon the following GUID format.

The ARM attribute Global_id.id (AIM attribute applied_classification_assignment.id) is written as the string in the following format:

<XX_Key x_href=YY Id=WW x-owner=ZZ x-rev=AA />

Argument definition:

XX_Key: XX is name of related ARM entity, e.g. Moulded_form_function_definition. This argument is required. The first letter of application object is upper case, rest lower case. Suffix = "_Key".

x-href=YY: YY is the URL of SOAP server and/or database This argument is required.

Id=WW: WW is a string which identifies object. Id is required only for definable_objects. The Id is one of two types:
Value (transparent usage- a unique value which identifies this instance
GUID (opaque usage) - a system generated identifier. Must be prefixed by "guid:"

x-owner=ZZ: ZZ is a string which identifies the object. This argument is required only for Definition objects. Same string as Id of related definable_object.

x-rev=AA: AA is revision number. This argument is required only for Definition objects. Same string as Definition.version_id attribute.

The combination of (XX_key, x-href, Id) is unique for definable_objects

The combination of (XX_key, x-href, x-owner, x-rev) is unique for Definitions

EXAMPLE 1 The following examples illustrate this usage for the related moulded_form entities in AP 216:

```
<Moulded_form_Key
x-href="//joshua.ingr.com"
Id="guid:123-123-123" />

<Moulded_form_functional_definition_Key
x-href="//joshua.ingr.com"
x-owner="guid:123-123-123"
x-rev="A"/>

<Moulded_form_design_definition_Key
x-href="//joshua.ingr.com"
x-owner="guid:123-123-123"
x-rev="A"/>
```

EXAMPLE 2 A Part entity is defined in EXPRESS and is instantiated in a repository whose locator is "http://www.acme.com/stepdataserver". The following example shows how its persistent identifier would be encoded and, then, represented in a Part 21 file:

```
SCHEMA part_schema;

ENTITY Part
  Name : STRING;
  Description: STRING;
END_ENTITY;

END_SCHEMA;
```

The globally-unique persistent identifier for an instance of this type would be encoded as:

```
<Part_Key  href = "http://www.acme.com/stepdataserver"  Id = "P1"  />
```

This example illustrates the three essential data components for globally-unique persistent instance identification: a globally unique type name, a globally-unique instance locator (repository), the locally unique identifier (key) within that locator (repository).

The globally unique type name is the combination of the EXPRESS schema and type names mapped according to ISO10303-28. The EXPRESS SCHEMA name is associated with the type's namespace ("urn:iso10303:osb/part_schema"); and the EXPRESS ENTITY name is captured as the qualified element name with the string '_Key' appended (p:Part_Key). The xlink:href attribute captures the locator of the repository which manages the instance. This includes the authority (company) and establishes the global uniqueness of the identifier. The remaining elements capture the local identification attribute names and values (Id = "P1") that are guaranteed to be unique within the repository. The recommended name if there is a single local identifier is Id.

A special attribute, x-rev, may be included to expose version information about the referenced instance. The rule is that if the x-rev is present it signifies a specific relationship of this instance to other instances whose keys have the same values for all fields besides x-rev. Each such instance represents different versions of the same object; the order of the versions is application-dependent.

A system that uses system-generated object identifiers (such as guid's) would use the guid string as the value of the first key attribute.

<p:Part_Key xmlns:p = "urn:iso10303:osb/part_schema" x-id = "_10"
xlink:href = "http://www.acme.com/stepdataserver"
Id = "guid: BDA4A1BA-110C-11d0-8CC3-0080F394BA32" x-rev = "A"/>

The globally-unique persistent identifier would appear in a STEP file as:

#10 = Part('P1','Description for this part');
#20 = applied_identification_assignment(
'<p:Part_Key xmlns:p = "urn:iso10303:osb/part_schema"
xlink:href = "http://www.acme.com/stepdataserver" Id = "P1" x-re v= "A"/>',
#21,#10);
#21 = identification_role('xml_guid',' ');

L.5 AIM units for each measure type

Table 6 specifies the units to be used in the AIM corresponding to each measure type.

Table 6 — Units used in AIM

Application Object attribute reference	Application Interpreted Model measure and unit
Area_measure	value_representation_item with value_-representation_item.value_component of area_-measure and derived_unit in set of Units referenced from global_unit_assigned_context.units. The derived_unit.elements shall be area_unit. The derived_unit.elements shall consist of one derived_unit_element: 1) derived_unit_element.unit = length_unit and derived_unit_element.exponent = 2.
Length_measure	value_representation_item with value_-representation_item.value_component of length_-measure and length_unit in set of Units referenced from global_unit_assigned_context.units
Positive_length_measure	value_representation_item with value_-representation_item.value_component of positive_-length_measure and length_unit in set of Units referenced from global_unit_assigned_context.units
Luminous_intensity_measure	value_representation_item with value_-representation_item.value_component of luminous_-intensity_measure and luminous_intensity_unit in set of Units referenced from global_unit_assigned_-context.units
Mass_measure	value_representation_item with value_-representation_item.value_component of mass_-measure and mass_unit in set of Units referenced from global_unit_assigned_context.units
Plane_angle_measure	value_representation_item with value_-representation_item.value_component of plane_-angle_measure and plane_angle_unit in set of Units referenced from global_unit_assigned_context.units.
Ratio_measure	value_representation_item with value_-representation_item.value_component of ratio_-measure and ratio_unit in set of Units referenced from global_unit_assigned_context.units
Thermodynamic_temperature_measure	value_representation_item with value_-representation_item.value_component of thermodynamic_temperature_measure and thermodynamic_temperature_unit in set of Units referenced from global_unit_assigned_context.units

Table 6 — Units used in AIM (continued)

Application Object attribute reference	Application Interpreted Model measure and unit
Volume_measure	value_representation_item with value_-representation_item.value_component of volume_-measure and derived_unit in set of Units referenced from global_unit_assigned_context.units. The derived_unit.name shall be volume_unit. The derived_unit.elements shall consist of one derived_unit_element: 1) derived_unit_element.unit = length_unit and derived_unit_element.exponent = 3.
Density_measure	value_representation_item with value_-representation_item.value_component of context_-dependent_measure and derived_unit in set of Units referenced from global_unit_assigned_context.units. The derived_unit.name shall be density_unit. The derived_unit.elements shall consist of two derived_-unit_elements: 1) derived_unit_element.unit = mass_unit and derived_unit_element.exponent = 1; 2) derived_unit_element.unit = length_unit and derived_unit_element.exponent = –3.
Force_measure	value_representation_item with value_-representation_item.value_component of context_-dependent_measure and derived_unit in set of Units referenced from global_unit_assigned_context.units. The derived_unit.name shall be force_unit. The derived_unit.elements shall consist of three derived_-unit_elements: 1) derived_unit_element.unit = mass_unit and derived_unit_element.exponent = 1; 2) derived_unit_element.unit = length_unit and derived_unit_element.exponent = 1; 3) derived_unit_element.unit = time_unit and derived_unit_element.exponent = -2.
Inertia_moment_measure	value_representation_item with value_-representation_item.value_component of context_-dependent_measure and derived_unit in set of Units referenced from global_unit_assigned_context.units. The derived_unit.name shall be inertia_moment_unit. The derived_unit.elements shall consist of one derived_unit_element: 1) derived_unit_element.unit = lenth_unit and derived_unit_element.exponent = 4.

Table 6 — Units used in AIM (continued)

Application Object attribute reference	Application Interpreted Model measure and unit
Moment_measure	value_representation_item with value_-representation_item.value_component of context_-dependent_measure and derived_unit in set of Units referenced from global_unit_assigned_context.units. The derived_unit.name shall be moment_unit. The derived_unit.elements shall consist of three derived_-unit_elements: 1) derived_unit_element.unit = mass_unit and derived_unit_element.exponent = 1; 2) derived_unit_element.unit = length_unit and derived_unit_element.exponent = 2; 3) derived_unit_element.unit = time_unit and derived_unit_element.exponent = -2.
Pressure_measure	value_representation_item with value_-representation_item.value_component of context_-dependent_measure and derived_unit in set of Units referenced from global_unit_assigned_context.units. The derived_unit.name shall be pressure_unit. The derived_unit.elements shall consist of three derived_-unit_elements: 1) derived_unit_element.unit = mass_unit and derived_unit_element.exponent = 1; 2) derived_unit_element.unit = length_unit and derived_unit_element.exponent = -1; 3) derived_unit_element.unit = time_unit and derived_unit_element.exponent = -2.
Speed_measure	value_representation_item with value_-representation_item.value_component of context_-dependent_measure and derived_unit in set of Units referenced from global_unit_assigned_context.units. The derived_unit.name shall be speed_unit. The derived_unit.elements shall consist of two derived_-unit_elements: 1) derived_unit_element.unit = length_unit and derived_unit_element.exponent = 1; 2) derived_unit_element.unit = time_unit and derived_unit_element.exponent = -1.

Table 6 — Units used in AIM (concluded)

Application Object attribute reference	Application Interpreted Model measure and unit
Weight_per_length_measure	value_representation_item with value_-representation_item.value_component of context_-dependent_measure and derived_unit in set of Units referenced from global_unit_assigned_context.units. The derived_unit.name shall be weight_unit. The derived_unit.elements shall consist of two derived_-unit_elements: 1) derived_unit_element.unit = mass_unit and derived_unit_element.exponent = 1; 2) derived_unit_element.unit = time_unit and derived_unit_element.exponent = -2.

Bibliography

[1] IDEF0 Federal Information Processing Standards Publication 183, Integration Definition for Function Modeling (IDEF0), FIPS PUB 183, National Institute of Standards and Technology, 1993-12.

[2] AP Development Guidelines for Shipbuilding, ISO TC184/SC4/WG3 N701, 1997-11-21.

[3] AP218 Test Case Definition Rev D, ISO TC184/SC4/WG3 N1341, ISE-2-ESTEP-TEAM-0016, 2003-01-31.

[4] ISO/IEC *Directives, Part 2 (Fourth edition, 2000), Rules for the structure and drafting of International Standards.*

[5] ISO 657-14 Hot-rolled steel sections – Part 14: Hot-finished structural hollow sections – Dimensions and sectional properties.

[6] ISO 31 Quantities and Units.

[7] ISO 690 Documentation – Bibliographic references –- content, form, and structure.

[8] ISO 1000 SI units and recommendations for use of their multiples and of certain other units.

[9] ISO 10303-44 Industrial automation systems and integration – Product data representation and exchange – Part 44: Integrated generic resource: Product structure configuration.

Index

ICS 25.040.40; 47.040

Price based on 878 pages