INTERNATIONAL STANDARD

ISO 10303-111

First edition
2007-05-15

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

Part 111:
Integrated application resource: Elements for the procedural modelling of solid shapes

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

Partie 111: Ressources d'application intégrée: Éléments pour la modélisation procédurale des formes solides

Reference number
ISO 10303-111:2007(E)

Contents

Page

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.

The main task of technical committees is to prepare International Standards. 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 document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 10303–111 was prepared by Technical Committee ISO/TC184/SC4, *Industrial automation systems and integration*, Subcommittee SC4, *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. ISO 10303-111 is a member of the integrated application resources series. The integrated generic resources and the integrated application resources specify a single conceptual product data model.

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

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

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 integrated resources series. This part of ISO 10303 specifies the **solid_shape_element_schema**.

A set of solid modelling shape elements is defined that provide a capability for the exchange of feature-based CAD solid models, expressed in terms of the sequence of successive creation or modification operations used to build them. This kind of representation of a product shape model is referred to as a *procedural*, *history-based* or *construction history model*. The essential underlying resource for the representation of models of this type is ISO 10303-55, which provides the mechanism for capturing the sequence of operations, and defines the intended interpretation of the entities defined in this part of ISO 10303 as modelling operations. This part of ISO 10303 contains a single schema, the **solid_shape_element_schema**, which defines a set of complex geometric elements that can be incorporated into a solid shape model. The relationship of this schema to other schemas that define the integrated resources of ISO 10303 is illustrated in Figure 1 using the EXPRESS-G notation. EXPRESS-G is defined in annex D of ISO 10303-11:2004. The schemas occurring in Figure 1 are components of ISO 10303 integrated resources, and they are specified in the following resource parts:

`support_resource_schema`	ISO 10303-41
`measure_schema`	ISO 10303-41
`geometry_schema`	ISO 10303-42
`topology_schema`	ISO 10303-42
`geometric_model_schema`	ISO 10303-42
`representation_schema`	ISO 10303-43
`mathematical_functions_schema`	ISO 10303-50
`explicit_geometric_constraint_schema`	ISO 10303-108
`sketch_schema`	ISO 10303-108

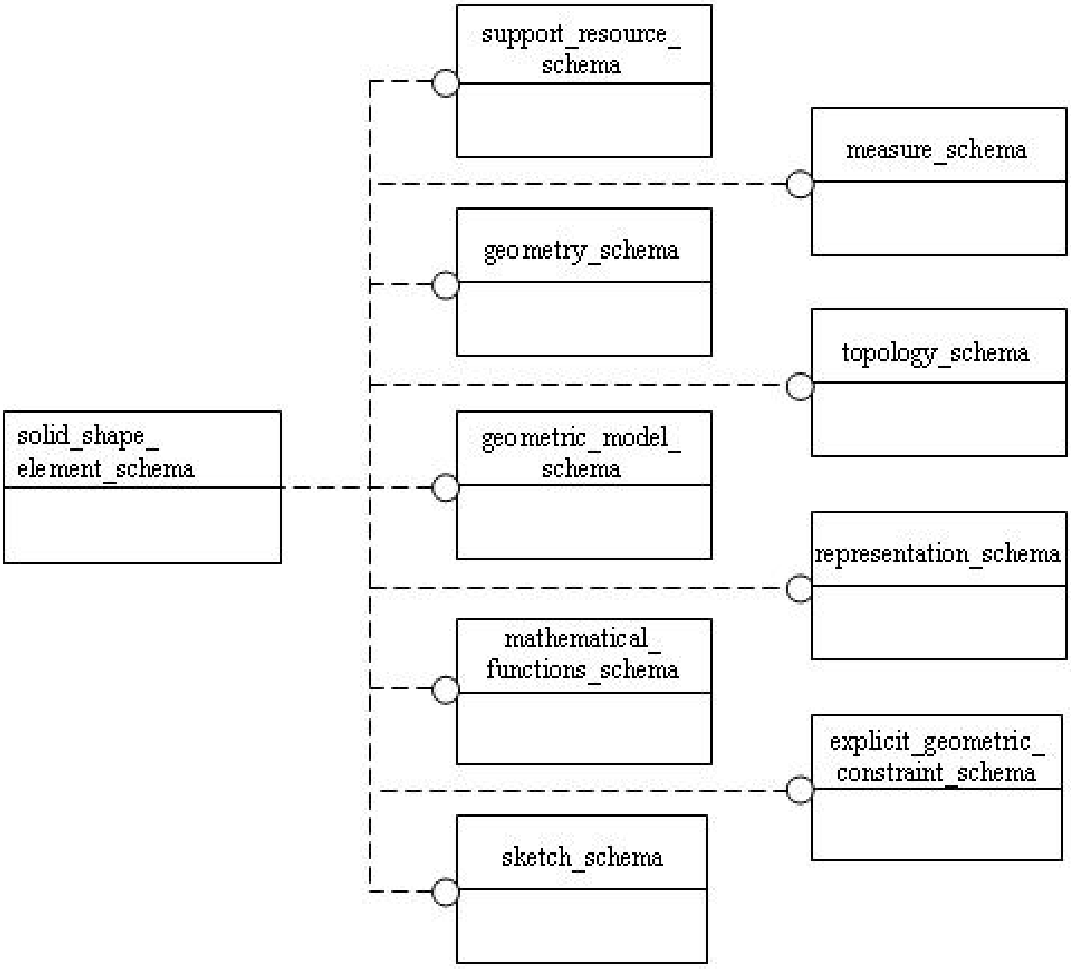


Figure 1 – Schema level diagram of relationships between the solid_-shape_element_schema of this part of ISO 10303 and other resource schemas

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

Part 111:
Integrated application resource: Elements for the procedural modelling of solid shapes

1 Scope

This part of ISO 10303 specifies resource constructs for representing the complex shape elements, sometimes known as form features, that are supported by the solid modelling capabilities of modern CAD systems. The elements are defined in such a way as to facilitate the exchange of solid models of products represented in terms of their constructional history.

NOTE 1 Procedural or constructional history models of solids can also include operations based directly on entities defined in ISO 10303-42, in particular Boolean operations and operations based on the various subtypes of **swept_face_solid** and **swept_area_solid**.

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

— the representation of solids having shape configurations resulting from blending and chamfering operations;

— the representation of solids with shape configurations resulting from offsetting, thickening, shelling and sculpturing operations;

— the representation of solids having shape configurations characteristic of certain manufacturing features, including several types of holes, pockets, slots and grooves;

— the representation of solids having circular and rectangular patterns of the types of shape configurations mentioned in the previous item of this list;

— the representation of solids that are generalizations of the solids of extrusion and revolution defined in ISO 10303-42.

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

— the representation of shape configurations on a model as *aspects* of the shape of the model, in the sense defined by ISO 10303-41;

NOTE 2 For design purposes the configurations specified in this part of ISO 10303 are regarded as shape elements in their own right, and not as local aspects of more complex shapes. These configurations can be created, subjected to modification and even subsequently deleted during the design process, so that they do not in general appear in their original form as shape aspects of the final model.

— the representation of features relating to manufacturing or other applications downstream of design.

NOTE 3 Such features have associated semantics that are absent from the shape configurations defined in this part of ISO 10303, which are purely concerned with the form of the modelled object.

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/IEC 8824-1, *Information technology — Abstract Syntax Notation One (ASN.1) - Part 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 10303-11:2004, *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual*

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 10303-50, *Industrial automation systems and integration — Product data representation and exchange — Part 50: Integrated generic resource: Mathematical constructs*

ISO 10303-55:2005, *Industrial automation systems and integration — Product data representation and exchange — Part 55: Integrated generic resource: Procedural and hybrid representation*

ISO 10303-108, *Industrial automation systems and integration — Product data representation and exchange — Part 108: Integrated application resource: Parameterization and constraints for explicit geometric product models*

3 Terms, definitions and abbreviations

3.1 Terms defined in ISO 10303-1

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

— application;

— application context;

— application protocol (AP);

— data exchange;

— exchange structure;

— implementation method;

— integrated resource (IR);

— product;

— product data.

3.2 Terms defined in ISO 10303-11

For the purposes of this document, the following terms defined in ISO 10303-11 apply.

— entity;

— entity data type;

— entity (data type) instance;

— instance;

— value.

3.3 Terms defined in ISO 10303-42

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

— boundary representation solid model (B-rep);

— constructive solid geometry (CSG);

— coordinate space;

— dimensionality;

— model space.

3.4 Terms defined in ISO 10303-43

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

— context of representation;

— element of representation;

— representation.

3.5 Terms defined in ISO 10303-55

For the purposes of this document, the following terms defined in ISO 10303-55 apply.

— design rationale.

3.6 Terms defined in ISO 10303-108

For the purposes of this document, the following terms defined in ISO 10303-108 apply.

— design intent;

— element;

— feature;

— history-based model;

— procedural model;

— sketch.

3.7 Other terms and definitions

For the purposes of this document, the following definition applies.

3.7.1
track
continuous chain of edges

NOTE The ISO 10303-42 entity **path** has a similar definition, but requires all the edges in the chain to be consistently oriented, which is not a requirement for a track.

3.8 Abbreviations

For the purposes of this document, the following abbreviations apply:

AP	application protocol (of ISO 10303)
B-rep	boundary representation
CAD	computer aided design
CSG	constructive solid geometry
IR	integrated resource (of ISO 10303)

4 Solid shape element

4.1 Introduction

The following EXPRESS declaration begins the solid shape element schema and identifies the necessary external references.

EXPRESS specification:

```
*)
SCHEMA solid_shape_element_schema;

REFERENCE FROM support_resource_schema                    -- ISO 10303-41
  (text);

REFERENCE FROM measure_schema                            -- ISO 10303-41
  (length_measure,
   plane_angle_measure,
   positive_length_measure,
   positive_plane_angle_measure);

REFERENCE FROM geometry_schema                            -- ISO 10303-42
  (axis2_placement_3d,
   bounded_curve,
   bounded_surface,
   curve_bounded_surface,
   geometric_representation_item,
   point,
   point_on_curve,
   surface);

REFERENCE FROM topology_schema                             --ISO 10303-42
  (connected_face_set,
   edge_curve,
   face_surface,
   open_shell,
   topological_representation_item,
   vertex);

REFERENCE FROM geometric_model_schema                      --ISO 10303-42
  (boolean_result,
   csg_primitive,
   extruded_face_solid,
   primitive_2d,
   revolved_face_solid,
   solid_model,
   swept_face_solid);

REFERENCE FROM representation_schema                       --ISO 10303-43
  (representation_item,
   using_items);
```

```
REFERENCE FROM mathematical_functions_schema            --ISO 10303-50
  (positive_integer);

REFERENCE FROM explicit_geometric_constraint_schema     --ISO 10303-108
  (non_negative_length_measure);

REFERENCE FROM sketch_schema                            --ISO 10303-108
  (positioned_sketch);
(*
```

NOTE 1 The schemas referenced above can be found in the following parts of ISO 10303:

support_resource_schema	ISO 10303-41
measure_schema	ISO 10303-41
geometry_schema	ISO 10303-42
topology_schema	ISO 10303-42
geometric_model_schema	ISO 10303-42
representation_schema	ISO 10303-43
mathematical_functions_schema	ISO 10303-50
explicit_geometric_constraint_schema	ISO 10303-108
sketch_schema	ISO 10303-108

NOTE 2 See annex D for a graphical presentation of this schema using the EXPRESS-G notation.

NOTE 3 A listing of the complete EXPRESS schema specified in this part of ISO 10303, without comments or other explanatory text, is available from the Internet – see annex C.

4.2 Fundamental concepts and assumptions

The fundamental types of operation available to the user of a modern CAD system are

a) Extrusion and revolution operations based on 2D sketches;

b) Boolean operations, for the combination of simpler shapes into more complex ones;

c) 'Local operations' that create local modifications on a pre-existing solid.

The use of ISO 10303-55, which provides the mechanisms needed for the capture and transfer of procedural models, allows operations of the types (a) and (b) to be represented by means of entities defined in ISO 10303-42, although the range of extrusion possibilities provided there is rather limited.

This part of ISO 10303 is mainly concerned with operations of type (c). It provides representations for complex geometric elements that can be used to create local modifications of a solid shape model during the course of a design procedure. Again, the essential underlying resource is ISO 10303-55, whose primary mechanism is based on the interpretation of an instance of a geometric element, if it occurs in the specialized context of a **procedural_representation_sequence** (as defined in ISO 10303-55), as an instruction to the receiving system to create such an element.

A type (c) operation assumes the pre-existence of a solid that will be modified by the operation. For that reason, many of the entities defined in the **solid_shape_element_schema** define a solid that has a new shape configuration of a specific type created upon it. The shape configurations available are described as *features* in the documentation of many CAD systems, but in a strict sense they are not features because they have no associated application semantics. The system user doubtless has some intended design functionality in mind when creating such a configuration on the model, but with current systems that intended functionality is not recorded. All that is captured by the system is the modified shape, and so it is better to think of the operations as shape creation operations rather than feature creation operations.

For the above reason, and also for more technical reasons spelled out in annex E, the use of the word 'feature' is henceforth avoided as far as possible in this part of ISO 10303.

NOTE 1 Some of the shape configurations defined in this schema have names that reflect methods of manufacture that might be used to produce them when the modelled part is actually made. This is because such names are typically employed in the user interfaces of CAD systems to denote those configurations; their use in this schema is not intended to imply any manufacturing-related semantics.

EXAMPLE Shape configurations such as slots, grooves, countersunk and counterbored holes provide illustrations of commonly occurring shape configurations named in terms of manufacturing processes.

Other operations defined in the **solid_shape_element_schema** are type (a) operations that generalize the basic extrusion and revolution operations specified in ISO 10303-42.

Some of the entities defined in this schema specify operations which, in the originating system, required the user to select one or more elements from the model as displayed on the screen of that system. In all such cases, the selection operations are captured and transferred by requiring the originating system to transmit explicit representations of the selected entities that enable the receiving system to reconstruct the selection procedure. The mechanism underlying this process is fully described in ISO 10303-55, a knowledge of which is fundamental to the understanding of this part of ISO 10303.

NOTE 2 Proof-of-concept transfers have confirmed that all major CAD systems can support the ISO 10303-55 approach for the capture and transfer of user-selected elements in exchanges of procedurally defined models.

The initial application context for the shape configurations defined in this part of ISO 10303 is the detail design of mechanical engineering parts without complex sculptured geometry. The use of ISO 10303-55 allows the procedural modelling of that type of geometry, if required, through the direct invocation of B-spline and related types of geometric entities defined in ISO 10303-42. That approach is possibly not optimal, and more convenient procedures for the creation and modification of free-form surface geometry may be included in a future edition of this document.

4.3 Solid shape element type definitions

4.3.1 base_solid_select

The **base_solid_select** type specifies those types of entity which may be used as the base solid for the creation of shape elements defined in this schema. The selection **solid_model** includes solids of extrusion and revolution generated by sweeping 2D sketches.

The entity **csg_primitive** in the SELECT list is itself a SELECT type, allowing the entity **primitive_2d** as a choice. This two-dimensional entity is out of scope for the modelling of solid shapes, and a WHERE rule is therefore imposed to disallow **primitive_2d** as a value of **base_solid_select**.

EXPRESS specification:

```
*)
TYPE base_solid_select = SELECT
  (solid_model,
   csg_primitive,
   boolean_result);
WHERE
   WR1: NOT('GEOMETRIC_MODEL_SCHEMA.PRIMITIVE_2D' IN TYPEOF(SELF));
END_TYPE;
(*
```

Formal propositions:

WR1: The entity **primitive_2d** shall not be permitted as a choice of **base_solid_select**.

4.3.2 blend_end_condition_select

The **blend_end_condition_select** enumerates possibilities for the termination conditions on a blend of a continuous track of edges on a solid. It is used by the entity **track_blended_solid_with_end_conditions** (see clause 4.4.4, where the interpretations of the three possibilities are detailed).

EXPRESS specification:

```
*)
TYPE blend_end_condition_select = SELECT
  (point_on_curve,
   edge_curve,
   vertex);
END_TYPE;
(*
```

4.3.3 generalized_surface_select

The **generalized_surface_select** type specifies surfaces and other surface-based geometric elements suitable for use as the sculpturing element in a **sculptured_solid** (see clause 4.4.12) or as the base entity for a **thickened_face_solid** (see clause 4.4.47).

EXPRESS specification:

```
*)
TYPE generalized_surface_select = SELECT
  (surface,
   face_surface,
   surfaced_open_shell);
END_TYPE;
(*
```

4.3.4 trim_condition_select

The **trim_condition_select** type allows a choice between different types of trimming conditions in the creation of a swept configuration. It is used in the definitions of **extruded_face_solid_with_trim_conditions** (see clause 4.4.48) and **revolved_face_solid_with_trim_conditions** (see clause 4.4.51).

EXPRESS specification:

```
*)
TYPE trim_condition_select = SELECT
  (length_measure,
   plane_angle_measure,
   generalized_surface_select,
   solid_model);
END_TYPE;
(*
```

4.3.5 blend_radius_variation_type

The **blend_radius_variation_type** enumerates possibilities for the variation of the radius of a blend along an edge, or a sequence of edges, between points at which specific values are given.

EXPRESS specification:

```
*)
TYPE blend_radius_variation_type = ENUMERATION OF
  (linear,
   cubic,
   unspecified);
END_TYPE;
(*
```

Enumerated item definitions:

linear: the blend radius varies linearly between radius definition points.

cubic: the blend radius varies as a cubic between radius definition points.

unspecified: the blend radius variation is not specified.

NOTE The interpretation of these three possibilities is detailed in clause 4.4.6, in the specification of the entity **solid_with_variable_radius_edge_blend**.

4.3.6 trim_intent

The type **trim_intent** enumerates descriptive specifications of termination conditions on solids of linear extrusion. These supplement the explicitly specified termination elements specified in the entity **extruded_face_solid_with_trim_conditions** (see clause 4.4.48).

EXPRESS specification:

```
*)
TYPE trim_intent = ENUMERATION OF
   (blind,
    offset,
    through_all,
    unspecified,
    up_to_next);
END_TYPE;
(*
```

Enumerated item definitions:

blind: The extrusion terminates at an extent determined by the value of a trim condition of type **length_-measure**, before it meets any other geometric element of the model under construction;

offset: The extrusion terminates at a given distance from a specified surface in the model;

through_all: The extrusion extends through the entire model, possibly making multiple entries to and exits from the part material;

unspecified: No descriptive specification is given.

up_to_next: The extrusion terminates when it first meets a surface in the model.

4.4 Solid shape element entity definitions

4.4.1 modified_solid

A **modified_solid** is a type of **solid_model** (as defined in ISO 10303-42) resulting from the creation of a new constructional element on an existing solid. Its attribute **base_solid** specifies the original solid upon which the modification is created. An attribute **rationale** is provided for the capture of textual information, if available, explaining the reasons for the use of the particular constructional operation concerned in the design sequence where it occurs.

EXPRESS specification:

```
*)
ENTITY modified_solid
  ABSTRACT SUPERTYPE OF (ONEOF
                         (edge_blended_solid,
                          sculptured_solid,
                          shelled_solid,
                          offset_face_solid,
                          modified_solid_with_placed_configuration))
  SUBTYPE OF (solid_model);
  rationale  : text;
  base_solid : base_solid_select;
END_ENTITY;
(*
```

Attribute definitions:

rationale: explanatory text detailing the reasons for the use of an instance of this entity.

base_solid: the initial solid before the new constructional element is created on it.

NOTE In common with the present entity, **procedural_representation_sequence** as defined in ISO 10303-55 also has an attribute **rationale** of type **text**. For **modified_solid** this attribute allows for the capture of the rationale underlying the designer's creation of an instance of a particular entity data type in the construction process. In the case of **procedural_representation_sequence** the corresponding attribute allows the capture of the rationale associated with a design procedure involving a sequence of such operations, where the designer's choice of a particular sequence may have significance.

4.4.2 edge_blended_solid

An **edge_blended_solid** is a type of **modified_solid** representing the result of blending one or more edges of the original base solid. The blend may take the form of a chamfer, in which case the result will not be a smooth transition between the faces adjoining along the edges concerned. Alternatively, it may take the form of a blend that gives a smooth (tangent-continuous) transition between the surfaces of those faces.

NOTE 1 Strictly speaking, a chamfer is not a blend, since its surface is not tangent-continuous with the faces on either side of the modified edge. However, blends and chamfers otherwise have very similar semantics, and for that reason they have been grouped together in this entity.

NOTE 2 Various types of blends may be defined between two or more solids, or between two or more surfaces not belonging to a solid model, but such blends are currently excluded from this schema.

In manifold solid modelling, no more than two faces adjoin at an edge. The blending modification causes that edge to disappear from the modified model; it is replaced by an interpolated face that blends the two faces previously separated by the edge concerned.

The exchange of a procedural representation of the blended model requires the exchange file to contain explicit representations of the original unblended edges, as defined in the sending system, to enable the corresponding edges to be identified in the receiving system. The identification mechanism is detailed in the definition of **user_selected_elements** in ISO 10303-55. An attribute **blended_edges** is provided for the capture and transfer of the explicit edges concerned.

NOTE 3 The base solid may in fact be of type **solid_model**, **csg_primitive** or **boolean_result**. The only case for which explicitly defined edges exist in the base solid is that where its type is **manifold_solid_brep**, a specific subtype of **solid_model**. The use of this entity therefore requires that the sending system, if it uses some form of representation other than **manifold_solid_brep** as its primary representation for the base solid, can generate a secondary representation of that form from which the necessary explicit edge information may be derived.

NOTE 4 This part of ISO 10303 provides no specifications for *vertex blends*, smooth transitions in regions where blend surfaces for adjoining edges of the base solid meet. The geometry generated by CAD systems for vertex blends is highly system-dependent. It is assumed that the receiving system will treat vertex blends in some default manner; they may subsequently be reconfigured using the system's native capabilities.

NOTE 5 This part of ISO 10303 provides no specifications for *blend caps*, small bounded surface regions created by CAD systems to complete a closed boundary shell after creation of a blend. Figures 2 and 3 in clause 4.4.5 provide an example. Simple replacement of the blended edge by the blend surface, and then trimming back the horizontal and vertical faces affected, would leave a small triangular gap on either side of the L-block. In this case the gaps are filled by modifying the boundaries of the side faces of the block to cover them, but in more complex cases less simple system-dependent strategies are used. It is assumed that the receiving system will treat blend caps in some default manner; they may subsequently be reconfigured using the system's native capabilities.

EXPRESS specification:

```
*)
ENTITY edge_blended_solid
  ABSTRACT SUPERTYPE OF (track_blended_solid ANDOR (ONEOF
                         (solid_with_constant_radius_edge_blend,
                          solid_with_variable_radius_edge_blend,
                          solid_with_chamfered_edges)))
  SUBTYPE OF (modified_solid);
  blended_edges : LIST [1:?] OF UNIQUE edge_curve;
END_ENTITY;
(*
```

Attribute definitions:

blended_edges: a list of distinct **edge_curve** instances specifying unblended edges of a manifold boundary representation of the original base solid.

4.4.3 track_blended_solid

A **track_blended_solid** is a type of **edge_blended_solid** in which the edges to be blended form a continuous open or closed track. A WHERE rule is imposed to ensure this condition.

NOTE 1 A track is related to a **path** as defined in ISO 10303-42 in that it consists of a list of distinct edges connected end to end so that it is possible to traverse all the edges in the list continuously from an initial vertex to a final vertex. The initial and final vertices may be the same in the case of a closed track. The difference is that a **path** consists of instances of **oriented_edge**, so that there is a sense associated with the path as a whole. A track, by contrast, is made up of unoriented edges.

NOTE 2 This ABSTRACT entity may only be instantiated as a complex instance with one of the instantiable subtypes of **edge_blended_solid**.

NOTE 3 Figure 4 in clause 4.4.5 shows an example of a **track_blended_solid**, in which the blended track is a sequence of five edges of the volume created by extruding a rectangle with two rounded corners. Since the blend is a constant radius blend, this solid can be represented by a complex instance of **track_blended_solid** and **solid_with_constant_radius_edge_blend**.

EXPRESS specification:

```
*)
ENTITY track_blended_solid
  ABSTRACT SUPERTYPE OF (track_blended_solid_with_end_conditions)
  SUBTYPE OF (edge_blended_solid);
WHERE
  WR1: check_continuous_edges(SELF\edge_blended_solid.blended_edges);
END_ENTITY;
(*
```

Formal propositions:

WR1: One vertex of each member of the list of **edge_curve** instances (except the last instance, in the case of an open track) shall be identical with one vertex of the following member.

4.4.4 track_blended_solid_with_end_conditions

A **track_blended_solid_with_end_conditions** is a type of **track_blended_solid** in which the blend does not both start and terminate at the end points of the specified track of edges. For this type of blended solid, the extremities of the blend are defined by the SELECT type **blend_end_condition_select** (see clause 4.3.2). This allows the three following choices:

a) a point lying on the initial or final edge in the track;

b) a planar **edge_curve** at the beginning or end of a pre-existing blend surface on the same solid, whose plane intersects the initial or final edge in the track between the end points of that edge;

c) the initial vertex of the first edge in the track, or the final vertex of the last edge.

When a point-based or edge-based end condition is imposed at only one end of the track, the other end of the blend will terminate at an end vertex of the track, as in the case of the supertype **track_blended_solid**. In that case the selected condition at the end where no special condition is imposed shall be of the vertex-based type. The vertex concerned is required to be instance identical with the initial vertex of the first edge in the track or the final vertex of the last edge.

In the second case listed above, the the newly created blend is required to join the pre-existing blend smoothly, with continuity of surface normal across the referenced **edge_curve** that forms their junction.

The creation of a **track_blended_solid_with_end_conditions** will involve, in the sending system, user selection of the terminal elements of the blend, if they are of the **point** or **edge_curve** types. The exchange file will need to contain explicit representations of those elements, as defined in the sending system, to enable the corresponding elements to be identified in the receiving system. The identification mechanism is detailed in the definition of **user_selected_elements** in ISO 10303-55. An attribute **end_conditions** is provided for the capture and transfer of the explicit elements concerned. The occurrence of an end condition of the **vertex** type will inform the receiving system that no special condition has been imposed at the blend boundary concerned. In that case no selection will be necessary. The vertex to be instanced will be available explicitly, because it belongs to one of the explicitly transferred **edge_curve** instances composing the track.

NOTE 1 Figure 6 in clause 4.4.6 shows an example of a **track_blended_solid_with_end_conditions**, in which the blended track is a sequence of five edges of the volume created by extruding a rectangle with two rounded corners. Since the blend is a variable radius blend, this solid can be represented by a complex instance of **track_blended_solid_with_end_conditions** and **solid_with_variable_radius_edge_blend**. Either or both of the small arcs that terminate the track blend at its two ends can be selected as termination conditions for subsequently defined blends.

EXPRESS specification:

```
*)
ENTITY track_blended_solid_with_end_conditions
  SUBTYPE OF (track_blended_solid);
  end_conditions : LIST [2:2] OF blend_end_condition_select;
WHERE
  WR1: SIZEOF(TYPEOF(SELF) *
     ['SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_CONSTANT_RADIUS_EDGE_BLEND',
      'SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_VARIABLE_RADIUS_EDGE_BLEND',
      'SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_CHAMFERED_EDGES']) = 1;
  WR2: NOT (('GEOMETRY_SCHEMA.VERTEX' IN TYPEOF(end_conditions[1]))
       AND ('GEOMETRY_SCHEMA.VERTEX' IN TYPEOF(end_conditions[2])));
  WR3: NOT (('GEOMETRY_SCHEMA.VERTEX' IN TYPEOF(end_conditions[1]))
       AND (NOT ((end_conditions[1]
            :=: SELF\edge_blended_solid.blended_edges[1].edge_start)
            XOR (end_conditions[1]
            :=: SELF\edge_blended_solid.blended_edges[1].edge_end))));
  WR4: NOT (('GEOMETRY_SCHEMA.VERTEX' IN TYPEOF(end_conditions[2]))
       AND (NOT ((end_conditions[2]
            :=: SELF\edge_blended_solid.blended_edges[HIINDEX(
                SELF\edge_blended_solid.blended_edges)].edge_start)
            XOR (end_conditions[2]
            :=: SELF\edge_blended_solid.blended_edges[HIINDEX(
                SELF\edge_blended_solid.blended_edges)].edge_end))));
END_ENTITY;
(*
```

Attribute definitions:

end_conditions: the **point**, **edge_curve** or **vertex** instances used to define the extremities of the blend.

Formal propositions:

WR1: This entity shall only be instanced as a complex instance with one of the entities **solid_with_constant_radius_edge_blend**, **solid_with_variable_radius_edge_blend** or **solid_with_chamfered_edges**.

WR2: The end conditions on the blend shall not both be defined by terminal vertices of the specified track of edges.

WR3: If the initial end condition is specified in terms of a vertex, that vertex shall be a vertex of the first edge in the track.

WR4: If the final end condition is specified in terms of a vertex, that vertex shall be a vertex of the last edge in the track.

NOTE 2 **WR2** prevents this entity from being employed in cases where the use of the supertype **track_blended_solid** is appropriate. **WR3** and **WR4** ensure that the relevant end vertex of the track belongs to either the first or the last edge of the track, in cases where only one of the more specialized blend end conditions is specified.

Informal propositions:

IP1: The selected terminating elements of a blend of this type shall have appropriate geometric relationships with the edges being blended so that they are valid end boundaries for the type of blend surface being created. In particular, if the initial or final boundary condition is of type **point_on_curve** then the specified **point_on_curve** and the corresponding **edge_curve** instance on which the point lies shall share the same basis curve.

IP2: If the initial or final boundary condition is of type **vertex** then the vertex referenced shall be either the initial or the final vertex of the track.

NOTE 3 Because the edges in a track are not oriented, **WR3** and **WR4** are necessary but not sufficient to ensure satisfaction of **IP2**.

4.4.5 solid_with_constant_radius_edge_blend

A **solid_with_constant_radius_edge_blend** is a type of **edge_blended_solid** in which a single constant blend radius value is associated with one or more edges of the original base solid.

In the current edition of this part of ISO 10303 the geometry of the transferred blend is restricted to the form most commonly implemented in CAD systems, known as the *rolling ball blend*. For this case, the surface of the blend is defined as the envelope of a constant radius sphere moving in contact with the surfaces of the two faces adjoining at the edge being blended. The curves of contact of the sphere with those two surfaces are the trim curves of the blend surface.

NOTE 1 There are other closely related interpretations of the rolling ball blend that result in slightly different geometry, especially when the blend radius is variable. The geometry of a blend surface is in any case usually represented approximately rather than exactly, and this may result in slight differences in the transferred results, even if the sending and receiving systems both use the same nominal interpretation. Further different types of blend have been implemented in various CAD systems for handling specialized blending problems. The transfer of a blend in the most commonly implemented rolling ball form may give a satisfactory result in the receiving system, but in cases where it does not it will be necessary to use the native capabilities of that system to edit the resulting geometry.

For a **solid_with_constant_radius_edge_blend** the **edge_curve** instances specified are not necessarily connected, and the order in which they are specified may be immaterial, because all edges are modified in the same way. In such a case it is not necessary to create a complex instance with **track_blended_-solid**. However, if a continuous track of edges is specified, and particularly if specialized end conditions are imposed on the blend then a complex instance shall be created with **track_blended_solid** or **track_-blended_solid_with_end_conditions**, as appropriate.

NOTE 2 An L-shaped block is shown in Figure 2. The creation of a constant radius blend on the single concave edge of this block results in the blended solid depicted in Figure 3.

NOTE 3 Figure 4 shows an example of a **track_blended_solid** as defined in clause 4.4.3, in which the blended track is a sequence of five edges of a volume created by extruding a rectangle with two rounded corners. The blend is a constant radius blend. This solid can be represented by a complex instance of **solid_with_constant_radius_-edge_blend** and **track_blended_solid**.

EXPRESS specification:

```
*)
ENTITY solid_with_constant_radius_edge_blend
  SUBTYPE OF (edge_blended_solid);
  radius : positive_length_measure;
END_ENTITY;
(*
```

Attribute definitions:

radius: the value of the constant blend radius.

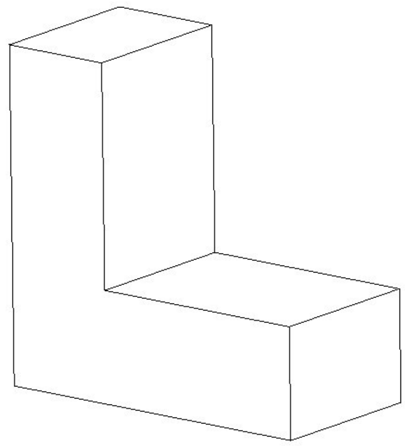


Figure 2 – L-section block

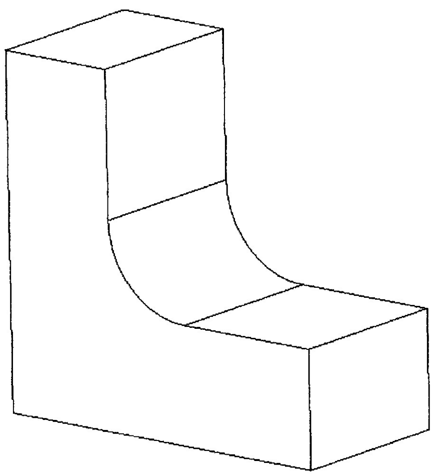


Figure 3 – L-section block of Figure 2 with constant radius edge blend

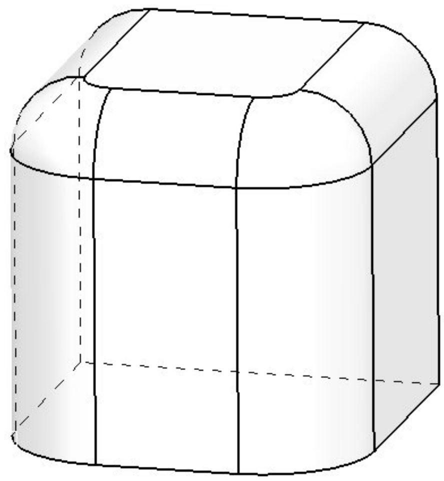


Figure 4 – A track blended solid with a constant radius edge blend

4.4.6 solid_with_variable_radius_edge_blend

A **solid_with_variable_radius_edge_blend** is a type of **edge_blended_solid** in which different radius values are specified at selected points of each of the edges concerned, and a specified interpolation method is used to compute blend radius values at intermediate points.

As stated in clause 4.4.5, the geometry of the transferred blend is assumed to be of the rolling ball type, though in this case the radius of the ball varies during its motion.

NOTE 1 The note in clause 4.4.5 concerning differences in the geometric interpretation of rolling ball edge blends applies for this entity also.

In an instance of this entity it is required that the edges blended shall be joined end-to-end in an open or closed track. This is necessary to ensure the correct correspondence of the specified radius-defining points with the edges concerned. This entity shall therefore only be instantiated as a complex instance with **track_blended_solid** or, if specialized end conditions are imposed on the blend, **track_blended_solid_with_end_conditions**. If any edge is required to have a constant radius blend, even though the track blend as a whole has a variable radius, then that edge shall be treated as an edge with linear variation of blend radius, with the same radius value being specified at each of its end points.

NOTE 2 Figure 6 shows a variable radius track blend with one constant radius blend segment.

In general, the positions of terminal vertices of edges in the list may or may not be used as radius-defining points. If **track_blended_solid_with_end_conditions** is not instantiated, the positions of the start vertex of the first edge of the list and the end vertex of the last edge of the list shall always be defined as selected points.

NOTE 3 A single connected track of edges can be blended by an instance of this entity. The variable radius blending of more general networks of edges will require the use of multiple instances.

In any interval whose radius-defining function is **cubic**, the actual function is determined using Hermite interpolation, in terms of the radii at each end point of the interval and values of the first derivatives of the radius variation function at those end points.

NOTE 4 Appropriate values of the required derivatives can be determined from the radius values at the end-points using standard methods for the computation of natural cubic spline functions [5]. The details differ slightly for the cases of open and closed edge tracks.

NOTE 5 In any interval for which the value of the radius-defining function is **unspecified** it is recommended that linear interpretation is used initially in the receiving system but that the user is warned that some other native blending capability of that system may be more appropriate.

NOTE 6 Figure 5 shows the L-shaped block of Figure 2 after creation of a variable radius blend on its concave edge. In this example the blend radius varies linearly along the length of the edge concerned.

NOTE 7 Figure 6 shows an example of a **track_blended_solid_with_end_conditions** as defined in clause 4.4.4, in which the blended track is a sequence of five edges of a volume created by extruding a rectangle with two rounded corners. The blend is a variable radius blend. This solid can be represented by a complex instance of **solid_with_variable_radius_edge_blend** and **track_blended_solid_with_end_conditions**.

EXPRESS specification:

```
*)
ENTITY solid_with_variable_radius_edge_blend
  SUBTYPE OF (edge_blended_solid, track_blended_solid);
  point_list         : LIST[2:?] OF point;
  radius_list        : LIST[2:?] OF positive_length_measure;
  edge_function_list : LIST[1:?] OF blend_radius_variation_type;
WHERE
  WR1: SIZEOF(point_list) = SIZEOF(radius_list);
  WR2: SIZEOF(edge_function_list) = SIZEOF(radius_list) - 1;
  WR3: NOT((point_list[1] = point_list[HIINDEX(point_list)]) AND NOT
         (radius_list[1] = radius_list[HIINDEX(radius_list)]));
END_ENTITY;
(*
```

Attribute definitions:

point_list: the list of points at which the radius is specified.

radius_list: the list of given radius values at specified points on the edge(s).

edge_function_list: the list of **blend_radius_variation_type** values defining the required variation of radius between consecutive points.

Formal propositions:

WR1: The number of radius-defining points shall be the same as the number of radius values.

WR2: The number of radius-defining functions shall be one less than the number of radius values.

WR3: If the first and last radius-defining points are the same, their associated radius values shall be equal.

Informal propositions:

IP1: The lists of points, radius values and edge functions shall be linked in the sense that the nth radius value is associated with the nth point, and the nth edge function is associated with the interval between the nth and $(n+1)$th points.

IP2: The specified radius-defining points shall be ordered in terms of increasing distance along the track from the initial point of the blend.

IP3: If any member of **point_list** is of type **point_on_curve** then its basis curve shall be identical with the basis curve of the **edge_curve** on which that point lies.

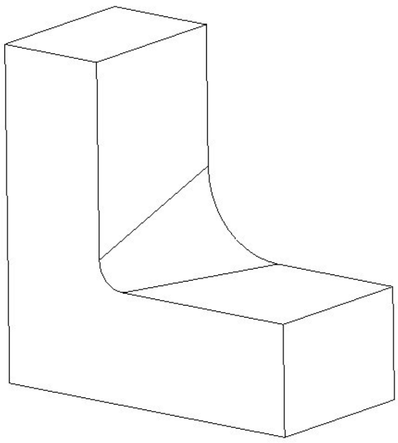


Figure 5 – L-section block of Figure 2 with variable radius edge blend

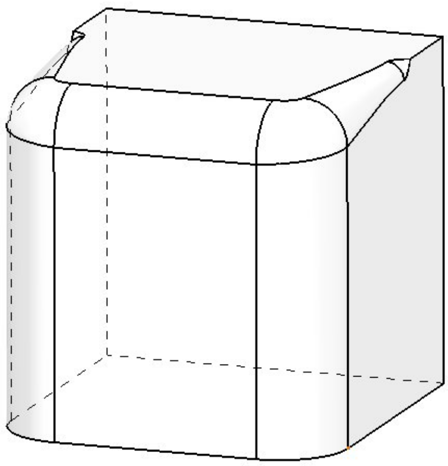


Figure 6 – A track blended solid with end conditions for which the blend radius is variable

4.4.7 solid_with_chamfered_edges

A **solid_with_chamfered_edges** is a type of **edge_blended_solid** in which the members of a list of edges are blended by the creation of bevelled faces on the resulting solid. In this case the blending surfaces are not tangent-continuous with the surfaces being blended. This is an ABSTRACT entity, with three instantiable subtypes.

For a **solid_with_chamfered_edges** the **edge_curve** instances specified are not necessarily connected, and the order in which they are specified may be immaterial, because all edges are modified in a consistent manner. In such a case it is not necessary to create a complex instance with **track_blended_solid**. However, if a continuous track of edges is specified, and particularly if specialized end conditions are imposed on the chamfer, then a complex instance shall be created from one of the instantiable subtypes of this entity and **track_blended_solid** or **track_blended_solid_with_end_conditions**, as appropriate.

NOTE 1 For two primary reasons, it is not appropriate to define the shape of a chamfer surface precisely in this part of ISO 10303:

a) Chamfer surfaces are ruled surfaces, computed in terms of a pair of bounding curves in each of the three subtypes of this entity specified below. No relationship is defined here between the rulings of chamfer surfaces and the parameterization of the bounding curves that are used in constructing them, because different CAD systems may differ in how they address this detail;

b) In principle, offset distances are measured along geodesics (curves of shortest distance) on the surfaces concerned [6]. On a planar surface, geodesics are straight lines. In other cases it is almost certain that CAD systems differ in the way they approximate the distances concerned.

Despite these uncertainties, the differences in the geometry generated by different CAD systems are not likely to be significant for most practical engineering purposes.

EXPRESS specification:

```
*)
ENTITY solid_with_chamfered_edges
  ABSTRACT SUPERTYPE OF (ONEOF
                        (solid_with_single_offset_chamfer,
                         solid_with_double_offset_chamfer,
                         solid_with_angle_based_chamfer))
  SUBTYPE OF (edge_blended_solid);
END_ENTITY;
(*
```

Informal propositions:

IP1: If the type list of an instance of this entity contains **track_blended_solid_with_end_conditions** and either or both end conditions are specified as being of type **edge_curve**, then the curve underlying any such referenced instance of **edge_curve** shall be a line.

NOTE 2 The **edge_curve** instances mentioned in **IP1** are the edges that terminate the blend surface at one or both extreme ends of the track for the case described.

4.4.8 solid_with_single_offset_chamfer

A **solid_with_single_offset_chamfer** is a type of **solid_with_chamfered_edges** in which the chamfers are created as follows:

— offset curves are defined on the two faces adjoining at each edge concerned, at equal specified distances from the edge;

— a ruled surface is constructed between the offset curves, and is used as the underlying surface of the new chamfer face.

EXPRESS specification:

```
*)
ENTITY solid_with_single_offset_chamfer
  SUBTYPE OF (solid_with_chamfered_edges);
  offset_distance : positive_length_measure;
END_ENTITY;
(*
```

Attribute definitions:

offset_distance: the offset distance, from the original edges, of the curves used in defining the chamfer surface.

4.4.9 solid_with_double_offset_chamfer

A **solid_with_double_offset_chamfer** is a type of **solid_with_chamfered_edges** in which the chamfers are created as in the case of the **solid_with_single_offset_chamfer** except that the chamfer boundaries are offset by different distances from the edge being chamfered. Two offset distance attributes are therefore defined. The first applies to the offset on the face to the left of the topological edge as transmitted in the file, the second on the face to the right of that edge, as traversed in the positive sense, viewed from outside the solid.

NOTE 1 It is imperative that the **edge_curve** instances that are members of the **blended_edges** attribute of the supertype **edge_blended_solid** are given the correct senses in the ISO 10303 representation. This is necessary to ensure that the faces lying to the left and right of each edge are consistent in the sending and the receiving systems.

NOTE 2 Figure 7 illustrates the double offset chamfer.

EXPRESS specification:

```
*)
ENTITY solid_with_double_offset_chamfer
  SUBTYPE OF (solid_with_chamfered_edges);
  left_offset_distance  : positive_length_measure;
  right_offset_distance : positive_length_measure;
END_ENTITY;
(*
```

Attribute definitions:

left_offset_distance: the offset distance on the left of the topological edge being blended.

right_offset_distance: the offset distance on the right of the topological edge being blended.

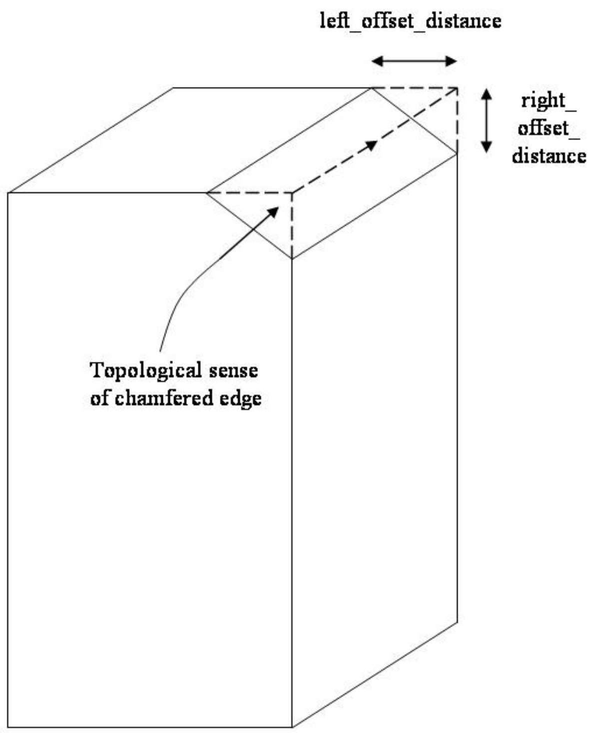


Figure 7 – Solid with double offset chamfer

4.4.10 solid_with_angle_based_chamfer

A **solid_with_angle_based_chamfer** is a type of **solid_with_chamfered_edges** in which the chamfers are defined in terms of an offset curve on one of the faces adjoining at the edge concerned and an offset angle measured with respect to the surface of that face. The BOOLEAN attribute **left_offset** is used to determine whether the offset curve is defined on the face to the left or to the right of the topological edge being chamfered. If this face is not planar, the chamfer angle is measured with respect to the tangent plane to the underlying surface at points on the offset curve.

NOTE 1 It is imperative that the **edge_curve** instances that are members of the **blended_edges** attribute of the supertype **edge_blended_solid** are given the correct senses in the ISO 10303 representation. This is necessary to ensure that the faces lying to the left and right of each edge are consistent in the sending and the receiving systems.

NOTE 2 Figure 8 illustrates the **solid_with_angle_based_chamfer**.

EXPRESS specification:

```
*)
ENTITY solid_with_angle_based_chamfer
  SUBTYPE OF (solid_with_chamfered_edges);
  offset_distance            : positive_length_measure;
  left_offset                : BOOLEAN;
  offset_angle               : positive_plane_angle_measure;
END_ENTITY;
(*
```

Attribute definitions:

offset_distance: the offset distance used in defining one boundary curve of the chamfer.

left_offset: a BOOLEAN attribute, TRUE if the offset curve is defined on the face to the left of the topological edge being chamfered, and FALSE otherwise.

offset_angle: the positive angle measured from the face on which the offset curve is defined. The angle is measured with respect to the tangent plane at points on that offset curve.

Informal propositions:

IP1: The value of **offset_angle** shall be less than $90°$ ($\pi/2$ radians).

NOTE 3 **IP1** can be made formal in any specialization of this schema in which the unit of angular measure is specified.

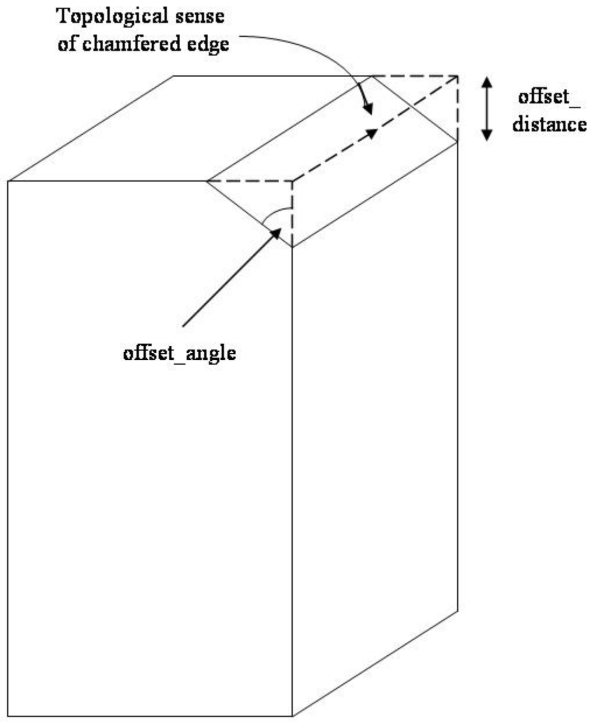


Figure 8 – Solid with angle based chamfer

4.4.11 surfaced_open_shell

A **surfaced_open_shell** is a type of **open_shell** (as defined in ISO 10303-42) with the constraint that all faces in the underlying **connected_face_set** shall be of type **face_surface**.

NOTE 1 The faces of an ISO 10303-42 **open_shell** are not required to have associated geometry.

NOTE 2 This entity is used in the definitions of **generalized_surface_select** as specified in clause 4.3.3 and **sculptured_solid** as specified in clause 4.4.12.

EXPRESS specification:

```
*)
ENTITY surfaced_open_shell
  SUBTYPE OF (open_shell);
WHERE
  WR1: SIZEOF(QUERY(q <* SELF\connected_face_set.cfs_faces |
         NOT ('TOPOLOGY_SCHEMA.FACE_SURFACE' IN TYPEOF(q)))) = 0;
END_ENTITY;
(*
```

Attribute definitions:

SELF\connected_face_set.cfs_faces: the set of faces composing the open shell.

Formal propositions:

WR1: All faces participating in a **surfaced_open_shell** instance shall be of type **face_surface**.

4.4.12 sculptured_solid

A **sculptured_solid** is a type of **modified_solid** that results from using a **surface**, **face_surface**, or **surfaced_open_shell** to partition the base solid into two or more solids, of which only one is retained. If only two solids result from the partitioning process, one of them will lie on each side of the sculpturing element. They are therefore easily distinguished by reference to the normal direction of that element, and the use of **sculptured_solid** as defined in this clause is appropriate.

The definition of the positive side of the sculpturing element depends on the nature of that element. For a **surface**, it is the side containing the positive surface normal vector. For a **face_surface** or a **surfaced_open_shell**, it is the side containing the topological normal to the face or shell respectively.

If more than two solids result from the partition by the sculpturing element then it may be necessary to select the solid to be retained. When that is the case the subtype **sculptured_solid_with_selection**, as defined in clause 4.4.13, shall be used.

NOTE 1 This entity provides the means for creating a solid, part of whose boundary lies on a free-form or sculptured surface, or a combination of such surfaces.

NOTE 2 Figures 9 and 10 show an example of the construction of a sculptured solid. The first figure shows the original solid and the generalized surface, and the second shows the solid that results after the partitioning operation.

EXPRESS specification:

```
*)
ENTITY sculptured_solid
  SUBTYPE OF (modified_solid);
  sculpturing_element : generalized_surface_select;
  positive_side       : BOOLEAN;
END_ENTITY;
(*
```

Attribute definitions:

sculpturing_element: the surface based element used to partition the solid.

positive_side: an indicator of the side of the sculpturing element on which the result of the partitioning operation is to be retained. If its value is TRUE the solid on the positive side of that element is retained, and vice versa.

Informal propositions:

IP1: The specified **sculpturing_element** shall partition the original base solid into two or more portions.

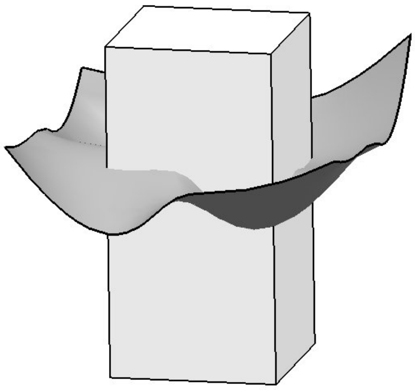


Figure 9 – Solid and generalized surface defining a sculptured solid

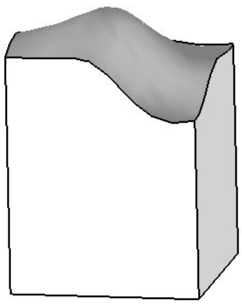


Figure 10 – The sculptured solid after the partitioning operation

4.4.13 sculptured_solid_with_selection

A **sculptured_solid_with_selection** is a type of **sculptured_solid** for use when a sculpturing operation as defined in clause 4.4.12 does not give a unique result. In this case the retained solid is distinguished from other generated solids by selection of the closed outer **shell** (as defined in ISO 10303-42) of an equivalent manifold boundary representation of that solid.

It is assumed that, in the sending system, the outer shell of the resulting retained solid is selected from the screen by the system user. The selection mechanism defined in ISO 10303-55 shall be used to capture this process.

NOTE Typically, the solid to be retained will be identified by user selection of a vertex, edge or face belonging to the boundary shell of the equivalent B-rep model. The ISO 10303-55 entities **user_selected_shape_elements** and

indirectly_selected_shape_elements provide the mechanisms for capturing and transferring the direct selection of the entire shell, or for its indirect selection in terms of one or more of its component elements.

EXPRESS specification:

```
*)
ENTITY sculptured_solid_with_selection
  SUBTYPE OF (sculptured_solid);
  retained_solid : topological_representation_item;
END_ENTITY;
(*
```

Attribute definitions:

retained_solid: an instance of **topological_representation_item** that uniquely identifies the closed shell of the selected solid to be retained.

Informal propositions:

IP1: the selected closed shell shall lie on that side of the sculpturing element indicated by the **positive_side** attribute of the supertype **sculptured_solid**.

4.4.14 offset_face_solid

An **offset_face_solid** is a type of **modified_solid** that results from selecting one or more faces on an existing solid and adding (or subtracting) a specified thickness of material to (or from) each of those faces. A positive value of the relevant member of the list **offset_distances** results in the addition of material in the positive sense of the topological normal of the corresponding face, and a negative value leads to the subtraction of material. The modified faces lie on parallel offsets of the original surfaces of those faces.

The exchange of an offset face solid requires the exchange file to contain explicit representations of the original faces, as defined in the sending system, to enable the corresponding faces to be identified in the receiving system. The identification mechanism is detailed in the definition of **user_selected_elements** in ISO 10303-55. An attribute **offset_faces** provides for the capture and transfer of the explicit **face_surface** instances concerned.

NOTE 1 The base solid may in fact be of type **solid_model**, **csg_primitive** or **boolean_result**. The only case for which explicitly defined faces exist in the base solid is that where its type is **manifold_solid_brep**, a specific subtype of **solid_model**. The use of this entity therefore requires that the sending system, if it uses some form of representation other than **manifold_solid_brep** as its primary representation for the base solid, generates a secondary representation of that form from which the necessary explicit face information may be derived.

NOTE 2 An **offset_face_solid** is shown in Figure 11. In this example the modified horizontal has been given a positive thickness value and the modified vertical face has been given a negative thickness value.

EXPRESS specification:

```
*)
ENTITY offset_face_solid
  SUBTYPE OF (modified_solid);
  offset_faces     : LIST [1:?] OF SET[1:?] OF face_surface;
  offset_distances : LIST [1:?] OF length_measure;
WHERE
  WR1: SIZEOF(offset_faces) = SIZEOF(offset_distances);
END_ENTITY;
(*
```

Attribute definitions:

offset_faces: a list of sets of instances of **face_surface**, the members of each set to be offset by a specified distance.

offset_distances: the list of offset distance values for the sets of **face_surface** instances.

Formal propositions:

WR1: The number of sets of faces to be thickened shall equal the number of thickness values.

Informal propositions:

IP1: The lists of offset face sets and offset distances shall be linked in the sense that the nth offset value is associated with the nth set of faces.

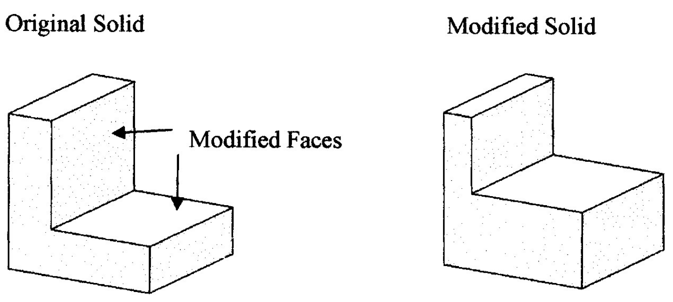


Figure 11 – Offset face solid

4.4.15 shelled_solid

A **shelled_solid** is a type of **modified_solid** created by the application of an operation called *shelling* or *hollowing* on the original base solid. The modified solid is generated by the deletion of certain specified faces of the base solid and the thickening of all the remaining faces. It is assumed that the faces to be deleted are selected by the user of the sending system, and that they are identified in the exchange file by the use of the ISO 10303-55 entity **user_selected_elements**. An attribute **deleted_face_set** is provided for the capture and transfer of the explicit **face_surface** instances concerned.

NOTE 1 The base solid may in fact be of type **solid_model**, **csg_primitive** or **boolean_result**. The only case for which explicitly defined faces exist in the base solid is that where its type is **manifold_solid_brep**, a specific subtype of **solid_model**. The use of this entity therefore requires that the sending system, if it uses some form of representation other than **manifold_solid_brep** as its primary representation for the base solid, generates a secondary representation of that form from which the necessary explicit face information may be derived.

NOTE 2 Figure 12 illustrates the **shelled_solid** entity.

EXPRESS specification:

```
*)
ENTITY shelled_solid
  SUPERTYPE OF (ONEOF
               (double_offset_shelled_solid,
                complex_shelled_solid))
  SUBTYPE OF (modified_solid);
  deleted_face_set : SET [1:?] OF face_surface;
  thickness        : length_measure;
WHERE
  WR1: thickness <> 0;
END_ENTITY;
(*
```

Attribute definitions:

deleted_face_set: the set of **face_surface** instances of the base solid whose members are to be deleted.

thickness: the thickness of material to be added to the retained faces of the base solid. If its value is positive, material is added to the exterior side of those faces, and if it is negative, material is added on the interior side.

Formal propositions:

WR1: The value of the attribute **thickness** shall not be zero.

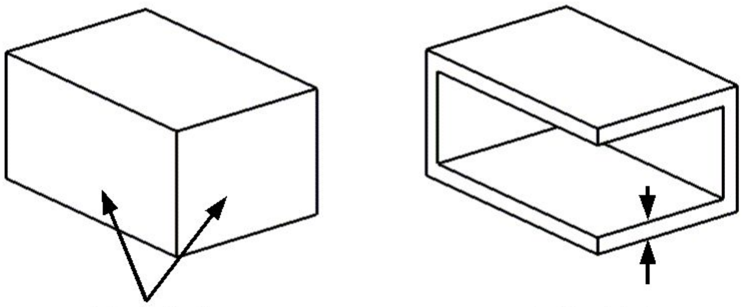


Figure 12 – Shelled solid

4.4.16 double_offset_shelled_solid

A **double_offset_shelled_solid** is a type of **shelled_solid** in which the retained faces of the original base solid have two associated thickness attributes. Both may be either positive or negative, and positive and negative thicknesses correspond to the addition of material in the direction of the outwards and inwards normals of the retained faces of the base solid, respectively. In this case the actual thickness of material is determined by the difference between the two specified (partial) thickness values.

EXPRESS specification:

```
*)
ENTITY double_offset_shelled_solid
  SUBTYPE OF (shelled_solid);
  thickness2 : length_measure;
WHERE
  WR1: thickness2 <> 0;
  WR2: SELF\shelled_solid.thickness <> thickness2;
END_ENTITY;
(*
```

Attribute definitions:

thickness2: the second thickness value to be applied to the retained faces of the base solid.

SELF\shelled_solid.thickness: the first thickness value to be applied to the retained faces of the base solid.

Formal propositions:

WR1: The value of the attribute **thickness2** shall not be zero.

WR2: The value of the attribute **thickness** of the supertype **shelled_solid** shall not be equal to the value of **thickness2** (their difference gives the actual thickness of material of the shelled solid).

4.4.17 complex_shelled_solid

A **complex_shelled_solid** is a type of **shelled_solid** in which selected faces have specified thicknesses differing from the default thickness defined at the supertype level in **shelled_solid** (see clause 4.4.15). As there, a positive thickness value results in the addition of material to the exterior side of the retained faces of the original solid, while a negative value results in the addition of material on the interior side of those faces.

EXPRESS specification:

```
*)
ENTITY complex_shelled_solid
  SUBTYPE OF (shelled_solid);
  thickened_face_list :  LIST [1:?] OF SET[1:?] OF face_surface;
  thickness_list      :  LIST [1:?] OF length_measure;
WHERE
  WR1: SIZEOF(thickened_face_list) = SIZEOF(thickness_list);
  WR2: SIZEOF(QUERY(q <* thickness_list | (q = 0))) = 0;
END_ENTITY;
(*
```

Attribute definitions:

thickened_face_list: the list of sets of **face_surface** instances whose associated thicknesses differ from the default value inherited from the supertype.

thickness_list: the list of thickness values to be applied to the sets of selected faces.

Formal propositions:

WR1: The numbers of members of **thickened_face_list** and **thickness_list** shall be equal.

WR2: No member of the thickness list shall have the value zero.

Informal propositions:

IP1: The lists **thickened_face_list** and **thickness_list** shall be linked in the sense that the nth thickness value is associated with the nth face set.

4.4.18 modified_solid_with_placed_configuration

A **modified_solid_with_placed_configuration** is a type of **modified_solid** in which a local shape configuration is positioned and oriented by means of an **axis_placement** rather than (as in the case of earlier entities in this schema) with respect to elements of the base solid.

EXPRESS specification:

```
*)
ENTITY modified_solid_with_placed_configuration
  ABSTRACT SUPERTYPE OF (ONEOF
                        (solid_with_depression,
                         solid_with_protrusion,
                         solid_with_shape_element_pattern))
  SUBTYPE OF (modified_solid);
  placing : axis2_placement_3d;
END_ENTITY;
(*
```

Attribute definitions:

placing: the **axis2_placement_3d** instance used to position and orient the placed configuration.

4.4.19 solid_with_depression

A **solid_with_depression** is a type of **modified_solid_with_placed_configuration** in which the base solid has been modified by the removal of material. This ABSTRACT entity is the supertype of four solids exhibiting specific types of depression configurations, all of them characterized by a **depth** attribute.

Three of these four types of depression may be either 'blind' or 'through'. A blind depression terminates within the part material, and a through depression penetrates right through it. An instance of a through depression shall be a complex instance of one of these three basic depression configurations with the additional subtype **solid_with_through_depression** (see clause 4.4.20). The fourth type of depression is the groove (see clause 4.4.37), for which only the 'blind' case is permitted.

NOTE The names given to some of the configurations represented by subtypes of **solid_with_depression** do not precisely define their shapes. In customary usage, 'hole' is used in general to denote a depression whose depth is greater than its other dimensions, 'pocket' to denote a depression whose depth is less than its other dimensions, 'slot' to denote a depression whose length is greater than either its width or its depth and which has a constant cross-section except (possibly) near its ends, and 'groove' to denote a circumferential depression of a similar type that exists on a surface with rotational symmetry. In the ensuing clauses, figures are provided that illustrate all these configurations. However, the descriptions given here are strictly informal. They are for guidance only, and do not impose any restrictions on the actual use of the entity types defined in this part of ISO 10303.

EXPRESS specification:

```
*)
ENTITY solid_with_depression
  ABSTRACT SUPERTYPE OF (solid_with_through_depression
    ANDOR ONEOF (solid_with_hole,
                 solid_with_pocket,
                 solid_with_slot,
                 solid_with_groove))
  SUBTYPE OF (modified_solid_with_placed_configuration);
  depth : positive_length_measure;
END_ENTITY;
(*
```

Attribute definitions:

depth: the depth of the depression.

4.4.20 solid_with_through_depression

A **solid_with_through_depression** is a type of **solid_with_depression** in which the depression penetrates completely through the part material. An attribute **exit_faces** is provided for the capture and exchange of the faces through which a through depression finally emerges from the part material. This allows for cases where a depression may penetrate through several layers of material, having multiple entries and exits. Specification of the face(s) through which the final exit occurs provides a precise means of delimiting the depth of the depression. This should be consistent with the value of the **depth** attribute of the supertype **solid_with_depression**. The referenced faces shall be explicit elements of a boundary representation model in the sending system, and the corresponding faces of the model under reconstruction in the receiving system shall be identified by a matching process.

In cases where the receiving system does not implement the concept of a through depression the **depth** value alone may be used, though errors may then arise if the model is subsequently modified. However, if the receiving system does provide the concept of a through depression the exit face information shall take precedence and the **depth** value shall be disregarded.

EXPRESS specification:

```
*)
ENTITY solid_with_through_depression
  SUBTYPE OF (solid_with_depression);
  exit_faces : SET[1:?] OF face_surface;
WHERE
  WR1: SIZEOF(TYPEOF(SELF) *
         ['SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_HOLE',
          'SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_POCKET',
          'SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_SLOT']) = 1;
END_ENTITY;
(*
```

Attribute definitions:

exit_faces: the set of **face_surface** instances through which a 'through' depression finally emerges from the part material.

Formal propositions:

WR1: The entity **solid_with_through_depression** may only be instanced in combination with one of the entities **solid_with_hole**, **solid_with_pocket** or **solid_with_slot**.

NOTE A depression having the form of a groove is not permitted to be of the 'through' type because such a groove would cut the base solid into two portions.

4.4.21 solid_with_hole

A **solid_with_hole** is a type of **solid_with_depression** in which the base solid has been modified by the creation of a hole. This ABSTRACT entity acts as the supertype of solids exhibiting all types of hole configurations. At present, the only subtype provided in this document is the stepped cylindrical hole.

NOTE This ABSTRACT entity allows for the future definition of additional hole configuration subtypes.

EXPRESS specification:

```
*)
ENTITY solid_with_hole
  ABSTRACT SUPERTYPE OF (solid_with_stepped_round_hole)
  SUBTYPE OF (solid_with_depression);
END_ENTITY;
(*
```

4.4.22 solid_with_stepped_round_hole

A **solid_with_stepped_round_hole** is a type of **solid_with_hole** where the hole is made up of a sequence of coaxial cylindrical segments, each segment after the first having a different radius from its predecessor in the sequence. A simple round hole may be represented using this entity with just one segment. A two-segment stepped round hole whose first segment has the larger of the two radii may be used to represent what is known as a *counterbored hole*.

NOTE 1 No requirement is imposed for segment diameters to decrease with increasing depth of the hole. Such a requirement may be appropriate for certain manufacturing methods, but not for others.

For holes of this type, the sum of the values of the members of the **depths** list is required to be equal to the value of the **depth** attribute of the supertype **solid_with_depression**.

NOTE 2 For a blind hole of this type, a conical or spherical bottom condition results in an extension beyond the defined depth of the hole. For a flat-bottomed hole, the position of the flat bottom defines the depth, and any associated fillet surface occurs *within* that defined depth (see clauses 4.4.25, 4.4.26, 4.4.27 and Figure 15).

The axis placement of the hole in a **solid_with_stepped_round_hole** has its origin at the entrance to the hole, where the hole axis intersects with the boundary shell of the base solid on which the hole is to be created. Its positive z-axis direction points out of the hole entrance along the axis of the hole; in the case of a blind hole the z-axis therefore points outwards from the part material.

NOTE 3 Figure 13 illustrates an instance of **solid_with_stepped_round_hole** having three segments.

EXPRESS specification:

```
*)
ENTITY solid_with_stepped_round_hole
SUPERTYPE OF (solid_with_stepped_round_hole_and_conical_transitions
  ANDOR ONEOF (solid_with_flat_bottom_round_hole,
               solid_with_conical_bottom_round_hole,
               solid_with_spherical_bottom_round_hole))
SUBTYPE OF (solid_with_hole);
```

```
  segments            : positive_integer;
  segment_radii       : LIST[1:segments] OF positive_length_measure;
  segment_depths      : LIST[1:segments] OF positive_length_measure;
DERIVE
  SELF\solid_with_depression.depth
                      : positive_length_measure := compute_total_depth(SELF);
WHERE
  WR1: NOT (('SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_THROUGH_DEPRESSION'
       IN TYPEOF(SELF)) AND (SIZEOF(TYPEOF(SELF) *
       ['SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_FLAT_BOTTOM_ROUND_HOLE',
       'SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_SPHERICAL_BOTTOM_ROUND_HOLE',
       'SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_FLAT_BOTTOM_ROUND_HOLE'])
       <> 0));
END_ENTITY;
(*
```

Attribute definitions:

segments: the number of cylindrical segments of the hole.

segment_radii: the list of radius values of the individual cylindrical segments of the stepped hole.

segment_depths: the list of lengths of the individual cylindrical segments of the stepped hole.

Formal propositions:

WR1: If the hole has **solid_with_through_depression** in its type list, that list shall not include any of the subtypes that define blind bottom conditions for holes of this type.

Informal propositions:

IP1: The lists of radii and lengths shall be linked in the sense that the nth item on the radius list is associated with the nth cylindrical segment of the stepped hole.

4.4.23 conical_stepped_hole_transition

A **conical_stepped_hole_transition** is a conical face interposed at a junction between two cylindrical segments of a stepped round hole. The cone is characterized in terms of its apex angle and its base radius. The apex of the cone always lies on the hole axis, on that side of the junction for which the hole radius is smaller, and the base radius of the cone is defined on the planar surface normal to that axis and intersecting it at the point where the radius change occurs. The apex angle is positive when the transition is from a larger to a smaller hole radius, in the sense of increasing hole depth, and negative if it is from a smaller to a larger radius.

NOTE 1 Such a conical transition is often referred to by the manufacturing-oriented name of *countersink*. It has the effect of bevelling the sharp convex edge that would otherwise occur at the hole entry or exit, or where the hole radius changes.

NOTE 2 Two occurrences of **conical_stepped_hole_transition** are shown in Figure 14.

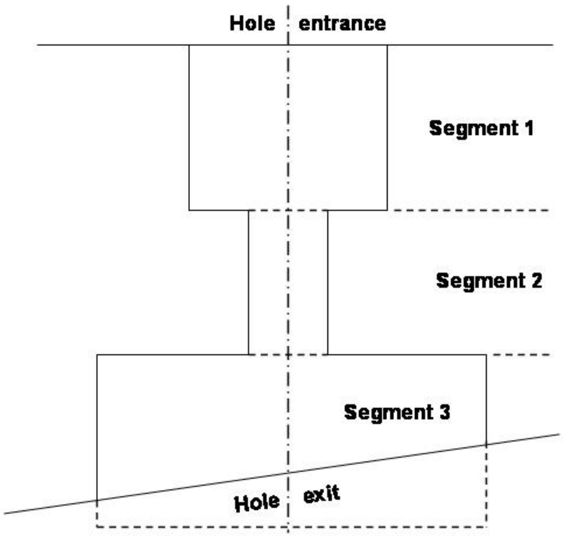


Figure 13 – Cross-section of solid with stepped round hole

EXPRESS specification:

```
*)
ENTITY conical_stepped_hole_transition
  SUBTYPE OF (geometric_representation_item);
  transition_number   : positive_integer;
  cone_apex_angle     : plane_angle_measure;
  cone_base_radius    : positive_length_measure;
END_ENTITY;
(*
```

Attribute definitions:

transition_number: the number of the segment junction to which the conical transition applies, counting the entrance to the hole as 1.

cone_apex_angle: the apex angle of the conical transition surface.

cone_base_radius: the base radius of the conical transition surface, as specified in the descriptive text.

Informal propositions:

IP1: The cone apex angle shall not have the value $180°$ (2π radians).

NOTE 3 A cone degenerates to a plane when its apex angle is 180°, and in this case there is no conical transition. Additionally, the value of **cone_base_radius** would be infinite. For these reasons the specified value of the apex angle is excluded. This proposition may be made formal in any environment where units of angular measure are defined.

4.4.24 solid_with_stepped_round_hole_and_conical_transitions

A **solid_with_stepped_round_hole_and_conical_transitions** is a type of **solid_with_stepped_round_-hole** in which one or more of the inter-segment junctions has been modified by the presence of a **conical_stepped_hole_transition** as described in clause 4.4.23. The entry to the hole is counted as the first junction, and the definition also permits the exit of a through hole to be modified in a similar manner. If such a conical transition is present at a junction between segments it modifies the adjacent cylindrical segment that has the smaller radius. If the entrance to the hole is through a face that is not both planar and normal to the hole axis, the radius specified for an initial conical transition shall be defined on a plane perpendicular to the hole axis and containing the origin of the associated axis placement. For a conical transition applied at the exit of a through hole, the radius shall be specified on a plane perpendicular to the hole axis and passing through the point at which that axis intersects the surface of the base solid in the vicinity of the hole exit.

NOTE 1 Figure 14 shows an instance of **solid_with_stepped_round_hole_and_conical_transitions** with three segments and two countersinks.

EXPRESS specification:

```
*)
ENTITY solid_with_stepped_round_hole_and_conical_transitions
  SUBTYPE OF (solid_with_stepped_round_hole);
  conical_transitions : SET[1:?] OF conical_stepped_hole_transition;
WHERE
  WR1: SIZEOF (conical_transitions) <=
         (SELF\solid_with_stepped_round_hole.segments + 1);
  WR2: ('SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_THROUGH_DEPRESSION'
         IN TYPEOF(SELF)) XOR (SIZEOF(conical_transitions) <=
         SELF\solid_with_stepped_round_hole.segments);
  WR3: validate_countersink_radii(SELF);
END_ENTITY;
(*
```

Attribute definitions:

conical_transitions: the set of conical transitions applying to the stepped hole.

Formal propositions:

WR1: The number of conical transitions applying to a stepped round hole shall in no case exceed one more than the number of cylidrical segments composing the hole.

WR2: Either the hole is a though hole or the number of conical transitions applying to it shall not exceed the number of cylindrical segments composing the hole.

WR3: For any conical transition other than at the hole entrance and (in the case of a through hole) the hole exit, the value of the base radius of the transition surface shall lie between the values of the radii of the preceding and succeeding cylindrical hole segments at the junction concerned. Also, any conical transition surface at the entrance to the hole shall have a positive apex angle, and any conical transition surface at the exit from the hole shall have a negative apex angle. For intermediate transitions the apex angle shall be positive for a transition from a larger to a smaller radius, and negative for a transition from a smaller to a larger radius.

NOTE 2 WR2 further restricts the number of transitions in the case of a blind hole because the bottom of a blind hole is modelled by a bottom condition rather than a transition.

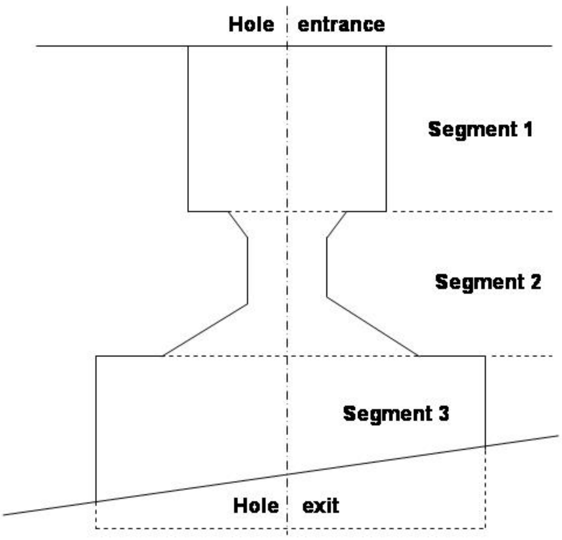


Figure 14 – Cross-section of solid with stepped round hole and conical transitions

4.4.25 solid_with_flat_bottom_round_hole

A **solid_with_flat_bottom_round_hole** is a type of **solid_with_stepped_round_hole** in which the bottom of the created blind hole lies on a planar surface normal to the hole axis. A fillet radius may be specified between this planar surface and the cylindrical shaft of the hole.

NOTE The bottom configuration of the hole in this entity is illustrated in Figure 15 in clause 4.4.27.

EXPRESS specification:

```
*)
ENTITY solid_with_flat_bottom_round_hole
  SUBTYPE OF (solid_with_stepped_round_hole);
  fillet_radius : non_negative_length_measure;
WHERE
  WR1 : fillet_radius <
          SELF\solid_with_stepped_round_hole.segment_radii[segments];
END_ENTITY;
(*
```

Attribute definitions:

fillet_radius: the radius of the fillet between the flat bottom of the hole and its cylindrical shaft.

Formal propositions:

WR1: The fillet radius shall be less than the radius of the last cylindrical segment of the hole.

4.4.26 solid_with_spherical_bottom_round_hole

A **solid_with_spherical_bottom_round_hole** is a type of **solid_with_stepped_round_hole** in which the bottom of the created blind hole lies on a spherical surface centred on the hole axis.

Since the **depth** attribute of the hole is defined as the sum of the lengths of the cylindrical segments making up the stepped hole, its value does not take into account the presence of the spherical bottom configuration.

NOTE The bottom configuration of the hole in this entity is illustrated in Figure 15 in clause 4.4.27.

EXPRESS specification:

```
*)
ENTITY solid_with_spherical_bottom_round_hole
  SUBTYPE OF (solid_with_stepped_round_hole);
  sphere_radius : positive_length_measure;
WHERE
  WR1 : sphere_radius >=
          SELF\solid_with_stepped_round_hole.segment_radii[segments];
END_ENTITY;
(*
```

Attribute definitions:

sphere_radius: the radius of the spherical surface defining the bottom of the hole.

Formal propositions:

WR1: The sphere radius shall be greater than or equal to the radius of the last cylindrical segment of the hole.

4.4.27 solid_with_conical_bottom_round_hole

A **solid_with_conical_bottom_round_hole** is a type of **solid_with_stepped_round_hole** in which the bottom of the created blind hole lies on a conical surface coaxial with the hole. The **semi_apex_angle** attribute specifies half of the included angle at the tip of the cone. The apex of the cone may be smoothly rounded with a specified radius by a spherical surface having its centre on the axis of the hole. This radius may be zero in the case where there is no tip rounding.

Since the **depth** attribute of the hole is defined as the sum of the lengths of the cylindrical segments making up the stepped hole, its value does not take into account the presence of the conical bottom configuration.

NOTE 1 The bottom configuration of the hole in this entity is illustrated in Figure 15.

EXPRESS specification:

```
*)
ENTITY solid_with_conical_bottom_round_hole
  SUBTYPE OF (solid_with_stepped_round_hole);
  semi_apex_angle : positive_plane_angle_measure;
  tip_radius      : non_negative_length_measure;
WHERE
  WR1 : tip_radius <
          SELF\solid_with_stepped_round_hole.segment_radii[segments];
END_ENTITY;
(*
```

Attribute definitions:

semi_apex_angle: half of the included angle at the tip of the cone.

tip_radius: the radius of the spherical surface rounding off the tip of the cone (may be zero).

Formal propositions:

WR1: The tip rounding radius shall be less than the radius of the last cylindrical segment of the hole.

Informal propositions:

IP1: The semi-apex angle of the conical hole bottom shall be less than 90° ($\pi/2$ radians).

NOTE 2 **IP1** can be made formal in any specialization of this schema in which the unit of angular measure is specified.

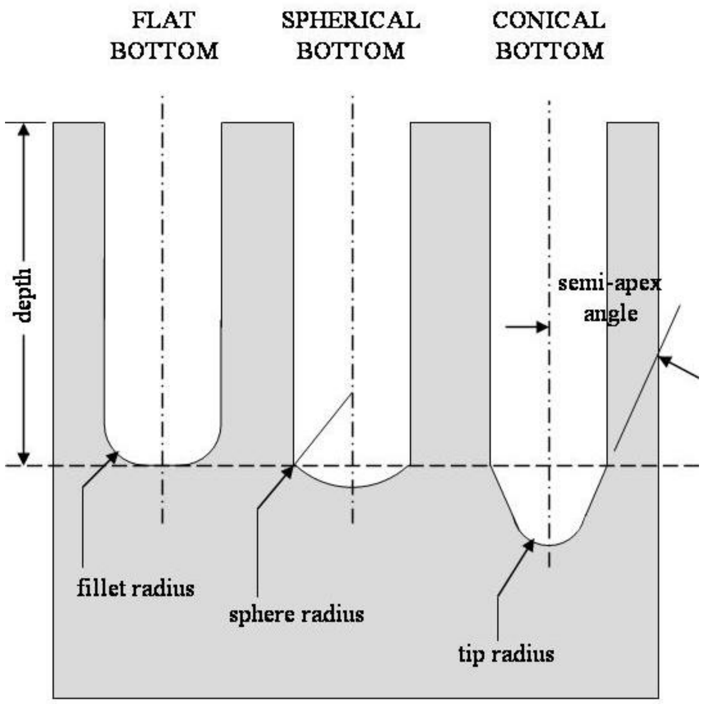


Figure 15 – Bottom conditions for blind round holes

4.4.28 solid_with_pocket

A **solid_with_pocket** is a type of **solid_with_depression** that results when a 2D closed profile is linearly extruded to generate a solid, which is then subtracted from the base solid to create a depression in it. This is an ABSTRACT entity, having instantiable subtypes with rectangular, circular and general profile shapes (see clauses 4.4.29 – 4.4.31).

NOTE 1 The first two of the instantiable subtypes listed above are not defined as subtypes of the pocket with general profile shape because of the difficulty of specializing attributes from the general case to handle diverse specific cases.

The axis placement of the pocket is oriented with its z-axis normal to the plane of the defining profile, and pointing out of the entrance to the pocket; in the case of a blind pocket the z-axis therefore points outwards from the part material. The axis placement is positioned so that its origin coincides with a reference point that is specified in the description of each particular profile form. This reference point is required to lie on the boundary shell of the base solid on which the pocket is to be created.

For a **solid_with_pocket** the **depth** attribute inherited from the supertype **solid_with_depression** is the distance between the origin of the axis placement and the planar floor of the pocket, measured in the negative z-direction. The attribute **floor_blend_radius** allows the specification of a blend or fillet radius between the floor and the walls of the pocket. The attribute **draft_angle** allows a possible deviation of the wall faces from perpendicularity with respect to the pocket floor. A positive value gives a form that tapers inwards with increasing depth, while a negative value gives an outward taper – in this case the area of the pocket floor is larger than that of the entrance to the pocket.

EXPRESS specification:

```
*)
ENTITY solid_with_pocket
ABSTRACT SUPERTYPE OF (ONEOF
                      (solid_with_rectangular_pocket,
                       solid_with_circular_pocket,
                       solid_with_general_pocket))
SUBTYPE OF (solid_with_depression);
  floor_blend_radius : non_negative_length_measure;
  draft_angle        : plane_angle_measure;
END_ENTITY;
(*
```

Attribute definitions:

floor_blend_radius: the radius of the blend or fillet between the floor and the pocket walls.

draft_angle: the angle between the pocket walls and the normal to the planar pocket floor. The pocket tapers inwards or outwards with increasing depth when this value is positive or negative, respectively. A zero value gives walls that are perpendicular to the pocket floor.

NOTE 2 If the pocket is a through pocket there is no pocket floor, and the value of the floor blend radius is immaterial. In such cases it may be set to zero.

Informal propositions:

IP1: The combination of pocket depth, draft angle and floor blend radius shall be such that the floor fillet does not intersect any edge in the loop of edges forming the boundary of the pocket configuration.

NOTE 3 Violation of **IP1** due to too large a choice of fillet radius would have the effect of modifying the specified profile that is extruded to create the pocket volume.

4.4.29 solid_with_rectangular_pocket

A **solid_with_rectangular_pocket** is a type of **solid_with_pocket** for which the defining pocket profile is a rectangle, which may have rounded corners. The attributes defined specify the length and width of the rectangle and the blending radius at its corners. The x- and y-axes of the associated axis placement are aligned with the pocket length and the pocket width, respectively. The origin of the axis placement is the centre point of the rectangular profile, and the pocket is positioned by requiring this point to lie on the surface of the base solid as it existed before creation of the pocket.

NOTE 1 If the specified corner radius is equal to the floor fillet radius defined at the level of the supertype **solid_with_pocket**, the vertex blends at the bottom corners of the pocket will lie on spherical surfaces. If the values of these two radii are different, however, this part of ISO 10303 does not specify the nature of those vertex blends, which may be generated in different system-dependent ways in the sending and receiving systems. Any such differences are not likely to have great significance in practical engineering terms.

NOTE 2 Figure 16 illustrates the form of a **solid_with_rectangular_pocket**.

EXPRESS specification:

```
*)
ENTITY solid_with_rectangular_pocket
  SUBTYPE OF (solid_with_pocket);
  pocket_length : positive_length_measure;
  pocket_width  : positive_length_measure;
  corner_radius : non_negative_length_measure;
WHERE
  WR1: (corner_radius < pocket_width/2)
         AND (corner_radius < pocket_length/2);
END_ENTITY;
(*
```

Attribute definitions:

pocket_length: the length of the rectangular pocket.

pocket_width: the width of the rectangular pocket.

corner_radius: the radius of the corner blends of the rectangle that is extruded to form the pocket volume. Its value may be zero.

NOTE 3 If the pocket has a non-zero draft angle the corner radius will vary with increasing pocket depth.

Formal propositions:

WR1: The specified corner radius shall be less than half the length of the shorter side of the rectangular profile.

NOTE 4 If the corner radius were allowed to be exactly half the length of the shorter side of the profile, the form of this configuration would reduce to that of a straight slot (see clauses 4.4.32 and 4.4.33).

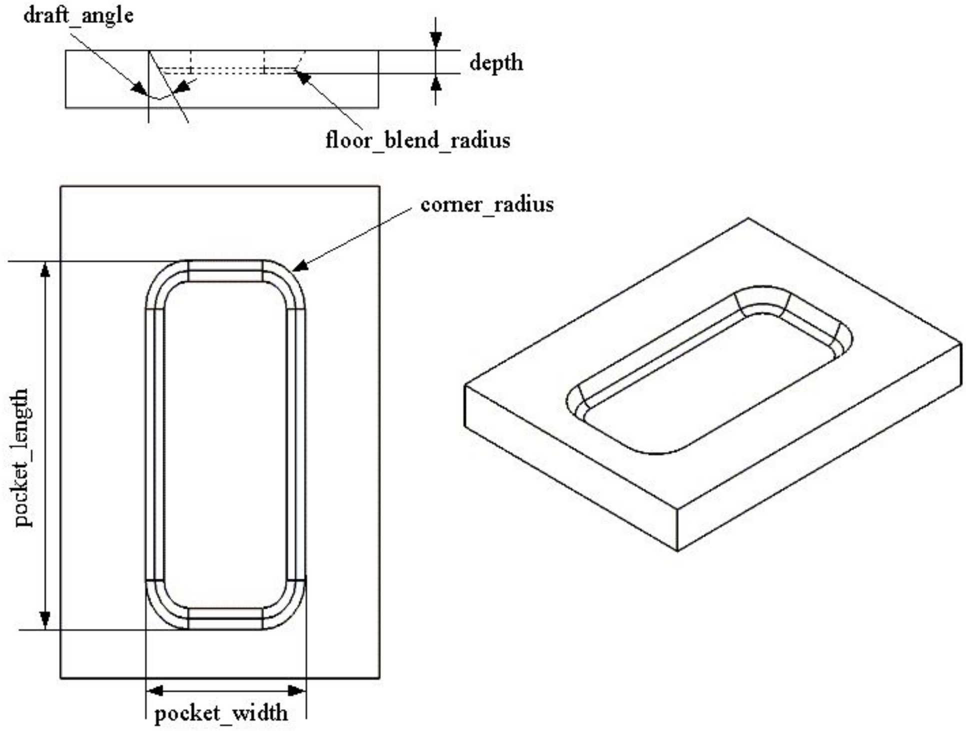


Figure 16 – Solid with rectangular pocket

4.4.30 solid_with_circular_pocket

A **solid_with_circular_pocket** is a type of **solid_with_pocket** for which the defining pocket profile is a circle. The single attribute specifies the radius of the circle. The origin of the associated axis placement is the centre point of the circle, and the pocket is positioned by requiring this point to lie on the surface of the base solid as it existed before creation of the pocket.

NOTE In the case where the **draft_angle** attribute of the supertype **solid_with_pocket** has the value zero, an instance of this entity could equally well be represented as an instance of **solid_with_flat_bottom_round_hole**. Informally, the distinction may be made that a circular pocket has a diameter exceeding its depth, while a circular hole has a depth exceeding its diameter. When the draft angle is non-zero, however, there is a clear distinction between the semantics of the pocket and hole cases – the types of hole currently defined in this schema do not have a draft angle.

EXPRESS specification:

```
*)
ENTITY solid_with_circular_pocket
  SUBTYPE OF (solid_with_pocket);
  pocket_radius: positive_length_measure;
WHERE
  WR1: SELF\solid_with_pocket.floor_blend_radius <= pocket_radius;
END_ENTITY;
(*
```

Attribute definitions:

pocket_radius: the radius of the circular hole.

SELF\solid_with_pocket.floor_blend_radius: the radius of the edge blend between the floor and the wall(s) of the pocket.

Formal propositions:

WR1: The value of **floor_blend_radius** in the supertype **solid_with_pocket** shall not exceed the radius of the circular pocket.

4.4.31 solid_with_general_pocket

A **solid_with_general_pocket** is a type of **solid_with_pocket** for which the defining pocket profile has a general closed non-self-intersecting configuration. The attribute **profile** specifies the planar pocket profile as a **positioned_sketch** as defined in ISO 10303-108. The attribute **reference_point** specifies a point in the plane of the **positioned_sketch** that is used to position the pocket on the base solid. It is used as the origin of the associated axis placement, and the pocket is positioned by requiring this point also to lie on the surface of the base solid as it existed before creation of the pocket. The x- and y-axes of the axis placement are those of the plane underlying the **positioned_sketch** defining the pocket profile.

EXPRESS specification:

```
*)
ENTITY solid_with_general_pocket
  SUBTYPE OF (solid_with_pocket);
  profile         : positioned_sketch;
  reference_point : point;
WHERE
  WR1: SIZEOF(['GEOMETRY_SCHEMA.CURVE_BOUNDED_SURFACE',
    'TOPOLOGY_SCHEMA.FACE_SURFACE'] * TYPEOF(profile.sketch_basis)) = 1;
  WR2: profile IN using_items(reference_point,[]);
END_ENTITY;
(*
```

Attribute definitions:

profile: the **positioned_sketch** defining the profile used to create the pocket.

reference_point: the point used as a reference for positioning the pocket on the base solid.

Formal propositions:

WR1: The instance of **positioned_sketch** used to define the pocket profile shall be based on one of the types **curve_bounded_surface** or **face_surface**, either of which defines a closed area.

WR2: The specified reference point shall be a constituent element of the cross-sectional profile.

4.4.32 solid_with_slot

A **solid_with_slot** is a type of **solid_with_depression** on which a slot has been created. A slot is a depression whose shape is generated by sweeping a constant symmetrical cross-sectional profile along a linear or curved centreline, the profile always being perpendicular to the tangent of that centreline. A slot may have one or two rounded ends generated by rotating half the slot cross-sectional profile about its axis of symmetry. Alternatively, one or both ends of the slot may emerge from the part material.

NOTE 1 In a traditional machining context, such a configuration is generated by motion through the part material of a shaped cutting tool, whose rotation automatically generates the rounded ends of the slot, if it has closed ends. As defined in this schema, however, the slot is a pure shape configuration and carries no manufacturing semantics.

Two specific forms of slot profile are provided in this schema. The first is the trapezoidal section, which includes as special cases the simple rectangular section, sections with draft angles and sections having the dovetail form. A floor fillet radius may be defined for the bottom corners of this type of section. The second profile form is that of the T-slot. The type list of every instance of **solid_with_slot** shall include one of the subtypes **solid_with_straight_slot** or **solid_with_curved_slot**, together with one of the subtypes **solid_with_trapezoidal_section_slot** or **solid_with_tee_section_slot**. These are defined, respectively, in clauses 4.4.33, 4.4.34, 4.4.35 and 4.4.36.

The axis placement of the slot is oriented with its z-axis pointing out of the entrance to the slot; in the case of a blind slot the z-axis therefore points outwards from the part material. For a straight slot its origin is positioned at the mid-point of the slot and its x-axis is oriented along the length of the slot. For a curved slot the axis placement of the slot is that used to define the bounded curve acting as the slot centreline. In either case, the origin is required to lie on the surface of the base solid as it existed before the creation of the slot.

If a slot has one or more open ends, the faces or sets of faces through which it emerges from the part material at its extremities are recorded. The slot may have multiple entries and exits along its length, but only the first entry and the last exit are recorded. In the linear case, these faces are ordered in the sense of the positive x-axis of the axis placement used to position and orient the slot. For a curved slot, the ordering is in the positive sense of the curve defining the centreline of the slot.

NOTE 2 An example of a slot with two entries and exits is shown in Figure 18. For this case the attribute **end_exit_faces** will record the outer faces of the rectangular block, as defined in the sending system.

EXPRESS specification:

```
*)
ENTITY solid_with_slot
  ABSTRACT SUPERTYPE OF ((ONEOF (solid_with_straight_slot,
                                 solid_with_curved_slot))
                     AND (ONEOF (solid_with_trapezoidal_section_slot,
                                 solid_with_tee_section_slot)))
  SUBTYPE OF (solid_with_depression);
  slot_width     : positive_length_measure;
  closed_ends    : LIST[2:2] OF LOGICAL;
  end_exit_faces : LIST[2:2] OF SET[0:?] OF face_surface;
```

```
WHERE
  WR1: NOT(('SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_THROUGH_DEPRESSION' IN
         TYPEOF(SELF)) AND (closed_ends = [FALSE,FALSE]));
  WR2: NOT(((closed_ends[1] = TRUE) AND (SIZEOF(end_exit_faces[1]) <> 0))
         OR ((closed_ends[2] = TRUE) AND (SIZEOF(end_exit_faces[2]) <> 0)));
END_ENTITY;
(*
```

Attribute definitions:

slot_width: the width of the slot, in senses that are defined precisely for the two subtypes **solid_with_-trapezoidal_section_slot** (see clause 4.4.35) and **solid_with_tee_section_slot** (see clause 4.4.36).

closed_ends: a set of two LOGICAL values, TRUE if the first and second extremities of the slot, respectively, are closed and FALSE if they are open. The UNKNOWN value is available for cases where a system does not support the distinction between slots with open and closed ends.

end_exit_faces: a list of two sets of **face_surface** instances, defining the entry and exit faces of an open-ended slot at its extremities. The sets of faces shall be ordered in the positive sense of the slot as defined above. If either extremity of the slot is closed, the corresponding set of faces shall be empty.

Formal propositions:

WR1: An instance of **solid_with_slot** shall not be open at both ends and also 'through' in the sense of having no bottom face (such a configuration would have the effect of cutting the base solid into two or more portions).

WR2: If either end of the slot is closed, the set of end exit faces corresponding to that end of the slot shall be empty.

4.4.33 solid_with_straight_slot

A **solid_with_straight_slot** is a type of **solid_with_slot** for which the slot has a straight centreline. The centreline is aligned with the x-axis of the associated axis placement, whose origin lies at its mid-point. This origin is required to lie on the surface of the base solid as it existed before creation of the slot. In the case of a slot with one or more open ends the specified length of the slot shall be consistent with the geometry of the exchanged object, but the exit face information shall take priority over the length information.

NOTE Figure 17 shows a straight slot with rectangular cross-section and two closed ends, while Figure 18 illustrates an open-ended straight slot with multiple entries and exits from the part material.

EXPRESS specification:

```
*)
ENTITY solid_with_straight_slot
  SUBTYPE OF (solid_with_slot);
  slot_length : positive_length_measure;
END_ENTITY;
(*
```

Attribute definitions:

slot_length: the length of that portion of the slot having parallel sides, excluding any rounded closed ends that may be present.

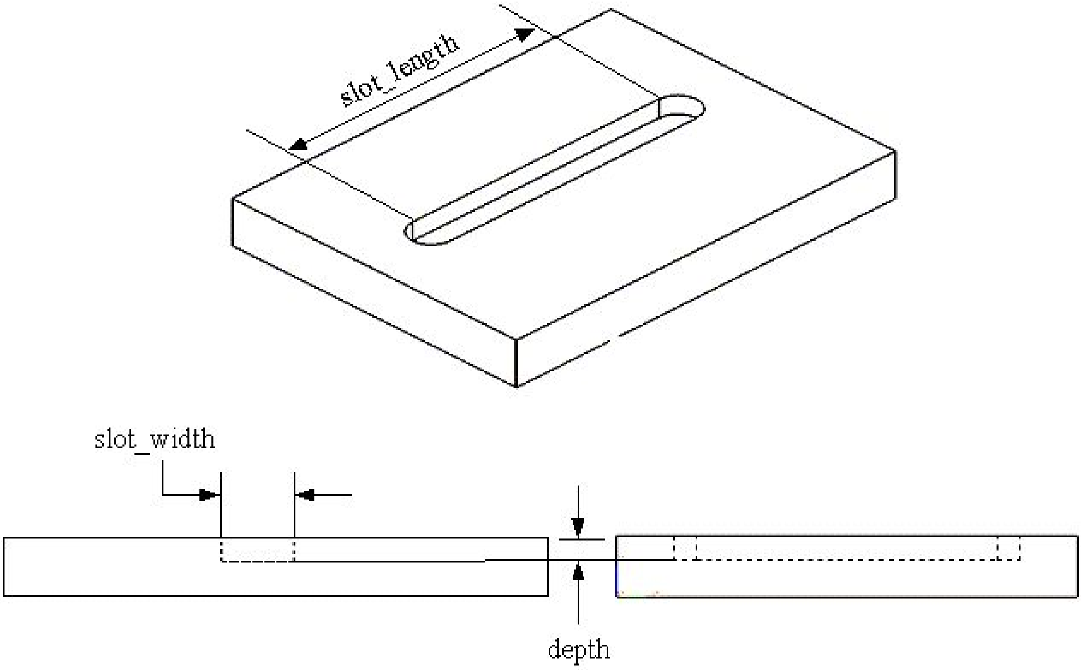


Figure 17 – Solid with straight slot, having two closed ends

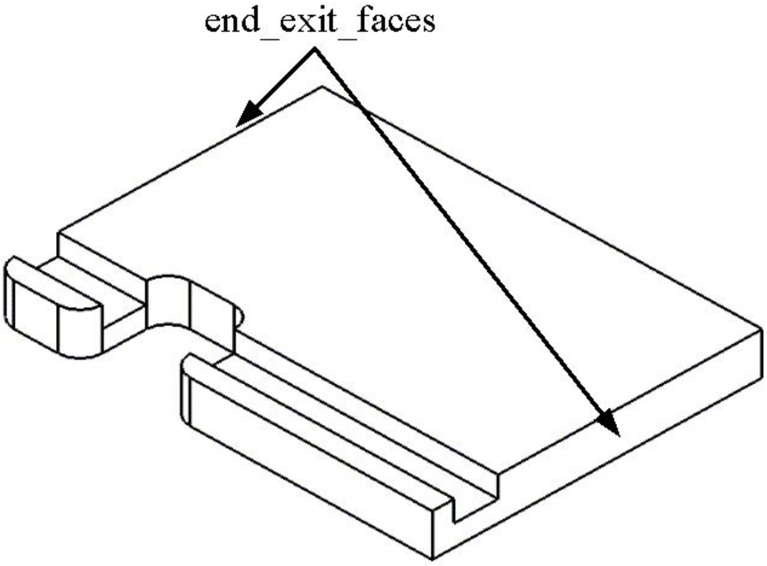


Figure 18 – Straight slot with multiple entries and exits from part material

4.4.34 solid_with_curved_slot

A **solid_with_curved_slot** is a type of **solid_with_slot** that has a curved centreline, defined by an instance of **bounded_curve**. The resulting slot geometry is that which would result from the motion of the centre of an appropriately shaped cutting tool along that curve between its two end points.

NOTE This description does not imply any specific manufacturing semantics for the curved slot.

The slot is positioned by requiring the origin of the axis placement used to specify the bounded curve to lie on the surface of the base solid as it existed before creation of the slot. In the case of a slot with one or more open ends the bounded curve used to specify its centreline shall be consistent with the geometry of the exchanged object, but the exit face information shall take priority over the curve information.

EXPRESS specification:

```
*)
ENTITY solid_with_curved_slot
  SUBTYPE OF (solid_with_slot);
  slot_centreline : bounded_curve;
END_ENTITY;
(*
```

Attribute definitions:

slot_centreline: the bounded curve used to specify that portion of the slot having parallel sides, excluding any rounded closed ends that may be present.

Informal propositions:

IP1: The bounded curve representing the slot centreline shall not be a straight line.

4.4.35 solid_with_trapezoidal_section_slot

A **solid_with_trapezoidal_section_slot** is a type of **solid_with_slot** in which the slot has the first of two possible cross-sections defined for slots in this part of ISO 10303. This section has the form of a symmetrical trapezium, whose depth (measured in the direction of the axis of symmetry) is defined by the **depth** attribute of the supertype **solid_with_depression** (see clause 4.4.19). The width is defined by the **slot_width** attribute of the supertype **solid_with_slot** (see clause 4.4.32). For this type of profile, the slot width is interpreted as the width at the entrance to the slot, corresponding to the width of the top of the trapezoidal section, assuming that its bottom lies on the floor of the slot.

NOTE 1 In its simplest form a trapezoidal-section slot has a flat bottom. However, the use of a zero draft angle with an appropriate choice of floor fillet radius will give rise to a U-section slot in which the section has a semicircular bottom.

NOTE 2 Figure 19 in clause 4.4.36 illustrates two cases of the trapezoidal-section slot. The first has a draft angle and a floor fillet, while the second is of dovetail form.

EXPRESS specification:

```
*)
ENTITY solid_with_trapezoidal_section_slot
SUBTYPE OF (solid_with_slot);
  draft_angle         : plane_angle_measure;
  floor_fillet_radius : non_negative_length_measure;
END_ENTITY;
(*
```

Attribute definitions:

draft_angle: the draft angle, measured with respect to the axis of symmetry of the slot section. For a positive draft angle the slot walls slope inwards with increasing depth. A negative draft angle gives rise to a slot of dovetail form.

floor_fillet_radius: the radius of the fillet between the floor and the walls of the trapezium on which the slot section is based. It may be zero.

NOTE 3 If a slot with this type of section is a through slot there is no slot floor, and the value of the floor fillet radius will normally be immaterial. In such cases it may be set to zero.

Informal propositions:

IP1: The draft angle shall lie in the open interval $(-90°, 90°)$ or in terms of radians $(-\pi/2, \pi/2)$.

IP2: The draft angle shall not have such a large positive value that the walls of the profile intersect at a point lying above the floor of the profile (in such a case the depth of the slot section would be incompatible with the value of the depth attribute of the supertype **solid_with_depression**).

IP3: The floor fillet radius shall be sufficiently small that at least three points of the underlying straight-sided trapezium remain, namely the mid-point of the profile floor and the top points of the profile walls.

NOTE 4 These propositions can be made formal in any application of this schema where units of angular measure are specified.

NOTE 5 If the limiting condition implied by **IP2** holds, the profile will take the form of a V section.

4.4.36 solid_with_tee_section_slot

A **solid_with_tee_section_slot** is a type of **solid_with_slot** in which the slot has the second of two possible cross-sections defined for slots in this part of ISO 10303. This section has the form of an inverted letter T. Its overall depth (measured in the direction of the axis of symmetry) is defined by the **depth** attribute of the supertype **solid_with_depression** as specified in clause 4.4.19. The width of the leg of the T (measured in the direction perpendicular to the depth) is defined by the **slot_width** attribute of the supertype **solid_with_slot** as specified in clause 4.4.32.

NOTE Figure 19 illustrates the cross-section of a T-section slot.

EXPRESS specification:

```
*)
ENTITY solid_with_tee_section_slot
  SUBTYPE OF (solid_with_slot);
  tee_section_width  : positive_length_measure;
  collar_depth       : positive_length_measure;
WHERE
  WR1: collar_depth < SELF\solid_with_depression.depth;
  WR2: tee_section_width > SELF\solid_with_slot.slot_width;
END_ENTITY;
(*
```

Attribute definitions:

tee_section_width: the width of the head (wider part) of the T section.

collar_depth: the length of the stem (narrower part) of the T section.

SELF\solid_with_depression.depth: the overall depth of the slot.

SELF\solid_with_slot.slot_width: the width of the stem of the T section.

Formal propositions:

WR1: The collar depth shall be less than the overall depth of the T section.

WR2: The tee-section width shall be greater than the width of the stem of the T section.

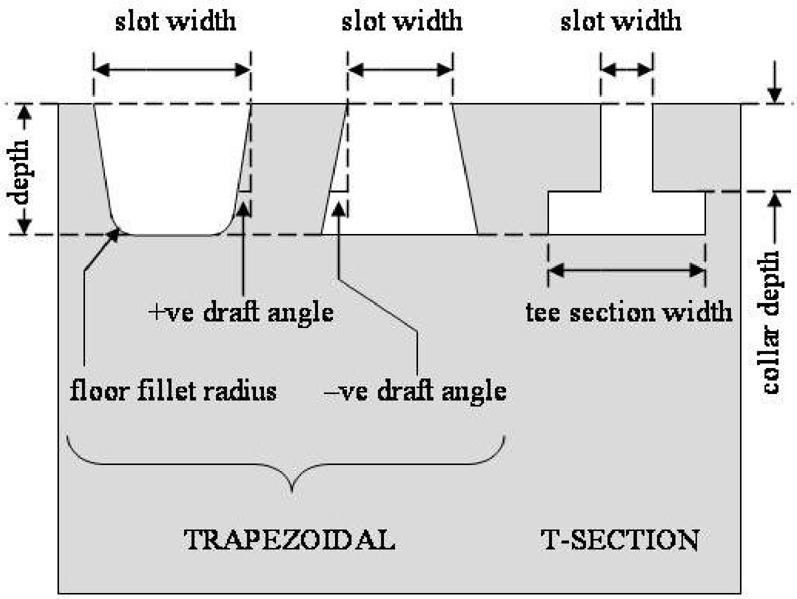


Figure 19 – Trapezoidal and T slot sections

4.4.37 solid_with_groove

A **solid_with_groove** is a type of **solid_with_depression** that results from the creation of a depression extending around a surface of revolution in such a way that its cross-sections are constant in any plane containing the axis of that surface. The cross-section is of symmetric trapezoidal form, which includes as special cases rectangular sections, sections with draft angles, U- and V-sections. For this entity, the **depth** attribute of the supertype **solid_with_depression** measures the depth of the trapezoidal section of the groove. Its value will not in general represent the physical depth of the groove, which has no well-defined value when the configuration is created on anything other than a circular cylindrical surface. For that particular case the physical depth of the groove is determined by the difference between the radius of the cylindrical surface and the value of the attribute **groove_radius**, which defines the position of the floor of the groove with respect to the axis of symmetry of the cylindrical surface.

The attribute **external_groove** is used to distinguish between external and internal grooves (examples are grooves defined on the outer face of a cylinder and on the cylindrical surface of a hole, respectively). The normal vector of the surface concerned has a component pointing away from its axis of revolution in the external case and towards it in the internal case.

The z-axis of the axis placement associated with a groove coincides with the axis of the surface on which the groove is created. Its origin lies at the intersection of that axis with the mid-plane of the groove.

NOTE 1 An example of **solid_with_groove** is illustrated in Figure 20, for the case of an external groove.

EXPRESS specification:

```
*)
ENTITY solid_with_groove
  SUBTYPE OF (solid_with_depression);
  groove_radius       : positive_length_measure;
  groove_width        : positive_length_measure;
  draft_angle         : plane_angle_measure;
  floor_fillet_radius : non_negative_length_measure;
  external_groove     : BOOLEAN;
WHERE
  WR1: NOT ('SOLID_SHAPE_ELEMENT_SCHEMA.SOLID_WITH_THROUGH_DEPRESSION'
         IN TYPEOF(SELF));
END_ENTITY;
(*
```

Attribute definitions:

groove_radius: the radius of the cylindrical floor surface of the groove.

groove_width: the width of the top of the groove cross-section.

draft_angle: the angle between the walls of the groove and its plane of symmetry. If its value is positive the walls slope inwards with increasing depth, and vice versa.

floor_fillet_radius: the fillet radius between the floor and the walls of the groove.

external_groove: a Boolean attribute whose value is TRUE if the groove is defined on an external surface of rotation and FALSE if it is defined on an internal surface of rotation.

Formal propositions:

WR1: A groove shall be a blind configuration (a through groove would cut the base solid into two parts).

Informal propositions:

IP1: The groove configuration shall not be defined on any surface that does not have axial symmetry.

IP2: The depth of the groove shall not be greater than the local radius of the axisymmetric surface on which the groove is defined.

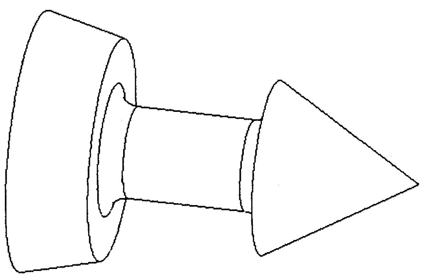
IP3: The draft angle shall lie in the open interval $(-90°, 90°)$, or in terms of radians $(-\pi/2, \pi/2)$.

IP4: The draft angle shall not have such a large positive value that the walls of the profile intersect at a point lying above the floor of the profile (in such a case the depth of the groove section would be incompatible with the value of the depth attribute of the supertype **solid_with_depression**).

IP5: The floor fillet radius shall be sufficiently small that at least three points of the underlying straight-sided trapezium remain, namely the mid-point of the profile floor and the two top points of the profile wall.

NOTE 2 Propositions **IP3**, **IP4** and **IP5** can be made formal in any application of this schema that specifies units of angular measure.

NOTE 3 When the draft angle is zero and the floor fillet radius is equal to half the slot width, a U-section results. If the limiting condition described in **IP4** holds, the profile will take the form of a V-section.



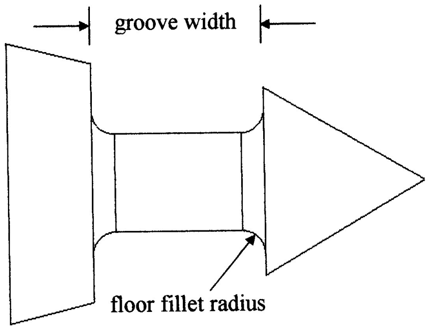


Figure 20 – Solid with groove

4.4.38 solid_with_protrusion

A **solid_with_protrusion** is a type of **modified_solid_with_placed_configuration** in which the base solid has been modified by the addition of material. This ABSTRACT entity is the supertype of solids exhibiting three specific types of protrusion, each defined by extrusion of a planar area in a direction normal to its plane of definition. The created protrusions are characterized by attributes **protrusion_height** and **protrusion_draft_angle**. The height of the protrusion is measured from the origin of its associated axis placement, in the direction of its positive z-axis, which is in the direction of extrusion. A positive draft angle implies that the cross-sectional area of the extrusion will decrease with increasing height, and vice versa.

EXPRESS specification:

```
*)
ENTITY solid_with_protrusion
  ABSTRACT SUPERTYPE OF (ONEOF
                        (solid_with_circular_protrusion,
                         solid_with_rectangular_protrusion,
                         solid_with_general_protrusion))
  SUBTYPE OF (modified_solid_with_placed_configuration);
  protrusion_height      : positive_length_measure;
  protrusion_draft_angle : plane_angle_measure;
END_ENTITY;
(*
```

Attribute definitions:

protrusion_height: the height of the protrusion.

protrusion_draft_angle: the draft or taper angle of the protrusion, measured with respect to the direction of extrusion.

Informal propositions:

IP1: The draft angle shall lie in the open interval $(-90°, 90°)$ or in terms of radians $(-\pi/2, \pi/2)$.

IP2: The draft angle shall not be so large as to reduce the protrusion height from its specified value.

NOTE **IP1** can be made formal in any in any application of this schema where units of angular measure are specified.

4.4.39 solid_with_circular_protrusion

A **solid_with_circular_protrusion** is a type of **solid_with_protrusion** in which the cross-section of the added volume of material is circular. The axis placement of a **solid_with_circular_protrusion** has its origin at the intersection of the boundary shell of the base solid with the axis of the protrusion to be created on it, with the positive z-axis pointing outwards from the part material of the base solid.

EXPRESS specification:

```
*)
ENTITY solid_with_circular_protrusion
  SUBTYPE OF (solid_with_protrusion);
  protrusion_radius : positive_length_measure;
END_ENTITY;
(*
```

Attribute definitions:

protrusion_radius: the radius of the cylindrical protrusion.

4.4.40 solid_with_rectangular_protrusion

A **solid_with_rectangular_protrusion** is a type of **solid_with_protrusion** in which the cross-section of the added extruded volume of material is a rectangle, possibly with rounded corners. The axis placement of a **solid_with_rectangular_protrusion** has its origin at the intersection of the boundary shell of the base solid with a line through the centre of the rectangle and normal to its plane. The positive z-axis direction points outwards from the part material of the base solid and coincides with that line.

EXPRESS specification:

```
*)
ENTITY solid_with_rectangular_protrusion
  SUBTYPE OF (solid_with_protrusion);
  protrusion_length        : positive_length_measure;
  protrusion_width         : positive_length_measure;
  protrusion_corner_radius : non_negative_length_measure;
WHERE
  WR1: (protrusion_corner_radius <= protrusion_width/2)
         AND (protrusion_corner_radius <= protrusion_length/2);
END_ENTITY;
(*
```

Attribute definitions:

protrusion_length: the length of the protrusion, measured in the x-direction of its axis placement.

protrusion_width: the width of the protrusion, measured in the y-direction of its axis placement.

protrusion_corner_radius: the corner radius of the rectangular protrusion cross-section. Its value may be zero.

NOTE The corner radius is that defined for the original rectangle. It will decrease with increasing protrusion height if a positive draft angle is imposed, and increase with increasing protrusion height for a negative draft angle.

Formal propositions:

WR1: The corner radius shall not exceed half the length of the shorter side of the rectangular profile.

4.4.41 solid_with_general_protrusion

A **solid_with_general_protrusion** is a type of **solid_with_protrusion** in which the cross-section of the added extruded volume of material is a general planar area, represented by an instance of **positioned_sketch** as defined in ISO 10303-108. The axis placement of a **solid_with_general_protrusion** has its origin at the intersection of the boundary shell of the base solid with a line through a reference point on the defining sketch and normal to the sketch plane. The positive z-axis direction points outwards from the part material of the base solid and coincides with this line.

EXPRESS specification:

```
*)
ENTITY solid_with_general_protrusion
  SUBTYPE OF (solid_with_protrusion);
  profile         : positioned_sketch;
  reference_point : point;
WHERE
  WR1: SIZEOF(['GEOMETRY_SCHEMA.CURVE_BOUNDED_SURFACE',
    'TOPOLOGY_SCHEMA.FACE_SURFACE'] * TYPEOF(profile.sketch_basis)) = 1;
  WR2: profile IN using_items(reference_point,[]);
END_ENTITY;
(*
```

Attribute definitions:

profile: the **positioned_sketch** defining the cross-section of the protrusion.

reference_point: the point of the sketch used to define the axis placement of the protrusion.

Formal propositions:

WR1: The instance of **positioned_sketch** used to define the protrusion cross-section shall be based on one of the types **curve_bounded_surface** or **face_surface**, either of which defines a closed planar area.

WR2: The specified reference point shall be a constituent element of the cross-sectional sketch.

4.4.42 solid_with_shape_element_pattern

A **solid_with_shape_element_pattern** is a type of **modified_solid_with_placed_configuration** generated when a set of replicas of any of the shape configurations created by a subtype of **modified_solid_with_placed_configuration** is arranged in a circular or rectangular pattern on the base solid. The original shape configuration participates in the pattern, in its original position and orientation.

The attribute **replicated_element** identifies the instance within the construction sequence where the configuration to be replicated was first created. It is important to note that it is only the local shape configuration created by the referenced operation that is replicated, not the entire resulting solid. The axis placement associated with each replicated element shall be determined from that of the original element by applying the same transformation to it as is used in replicating the original element.

The attribute **base_solid** of the supertype **modified_solid** identifies the solid on which the pattern is to be created. It may or may not be identical with the solid bearing the element to be replicated.

EXAMPLE A hole configuration may be created, a Boolean union operation performed between the resulting solid and another solid, and the hole then replicated in a pattern. In this case **replicated_element** will reference the solid on which the hole was first defined but **base_solid** will reference the result of the Boolean operation. On the other hand, if a hole is created in one operation and the immediately following operation replicates it in a pattern on the same base solid, both the **replicated_element** and the **base_solid** attributes of the pattern creation operation will reference the hole creation operation.

EXPRESS specification:

```
*)
ENTITY solid_with_shape_element_pattern
  ABSTRACT SUPERTYPE OF (ONEOF
                        (solid_with_circular_pattern,
                         solid_with_rectangular_pattern))
  SUBTYPE OF (modified_solid_with_placed_configuration);
  replicated_element : modified_solid_with_placed_configuration;
END_ENTITY;
(*
```

Attribute definitions:

replicated_element: the instance in the construction sequence representing the initial creation of the configuration to be replicated.

4.4.43 solid_with_circular_pattern

A **solid_with_circular_pattern** is a type of **solid_with_shape_element_pattern** in which multiple replicates of the specified shape configuration are arranged in a circular pattern. The axis placement associated with the pattern, specified at the level of the supertype **modified_solid_with_placed_configuration**, has its origin at the centre of the circle. The attribute **angular_spacing** specifies the angular rotation between successive replicates about the centre of the circle. Each replicate of the original configuration may retain its previous orientation in the pattern, or may be rotated about the centre of the circle, depending on the value of the BOOLEAN attribute **radial_alignment**.

NOTE 1 Figure 21 diagrammatically illustrates the two cases of **solid_with_circular_pattern**, the first where the replicates are not individually rotated about the centre of the pattern, and the second where they are individually rotated. The original shape configuration is drawn with thin lines, and the replicates with thick lines.

EXPRESS specification:

```
*)
ENTITY solid_with_circular_pattern
  SUPERTYPE OF (solid_with_incomplete_circular_pattern)
  SUBTYPE OF (solid_with_shape_element_pattern);
  replicate_count  : positive_integer;
  angular_spacing  : plane_angle_measure;
  radial_alignment : BOOLEAN;
```

```
  reference_point  : point;
END_ENTITY;
(*
```

Attribute definitions:

replicate_count: the number of replicates, not including the original configuration that is being replicated.

angular_spacing: the angular increment between successive replicates. Positive and negative increments give, respectively, anticlockwise and clockwise rotation.

radial_alignment: specifies whether elements are rotated or not. If TRUE, the replicates are individually rotated about the centre of the circular pattern; if FALSE, each point in the pattern has the same translation as the reference point, so that the orientation of the original shape configuration is maintained.

reference_point: a point associated with the original configuration to be replicated, whose projected distance from the axis of rotation determines the radius of the circular pattern. Each replicate has a corresponding transformed reference point. Successive pairs of reference points have equal angular spacing about the centre of the circular pattern.

Informal propositions:

IP1: The product of (**number_of_replicates** + 1) with the absolute value of **angular_spacing** shall not exceed 360 degrees or 2π radians.

NOTE 2 **IP1** may be made formal in any application in which units of measure are specified.

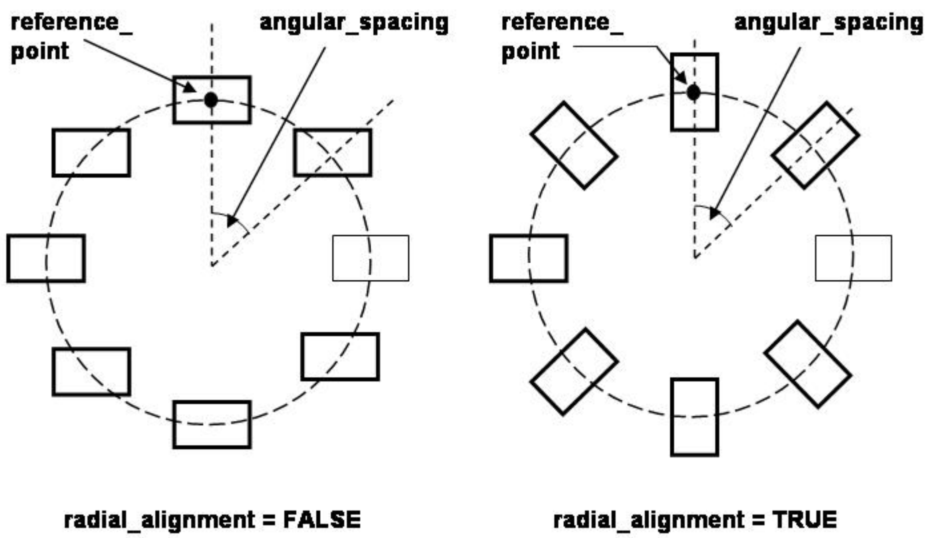


Figure 21 – The effect of radial alignment in solid_with_circular_pattern

4.4.44 solid_with_rectangular_pattern

A **solid_with_rectangular_pattern** is a type of **solid_with_shape_element_pattern** in which multiple replicates of the specified shape configuration are arranged in a rectangular pattern. Each replicate retains the orientation of the original. The numbers of rows and columns of the pattern are specified, together with the row and column separation distances. In the associated axis placement, specified at the level of the supertype **modified_solid_with_placed_configuration**, the x-axis is the row direction of the pattern and the y-axis the column direction. The origin of this axis placement coincides with the origin of the axis placement of the original shape configuration. If the value of the attribute **column_spacing** is positive, columns are added in the positive x-direction with respect to the base element, and vice versa. If the value of the attribute **row_spacing** is positive, rows are added in the positive y-direction with respect to the base element, and vice versa.

NOTE Figure 22 provides an illustration of **solid_with_rectangular_pattern**.

EXPRESS specification:

```
*)
ENTITY solid_with_rectangular_pattern
  SUPERTYPE OF (solid_with_incomplete_rectangular_pattern)
  SUBTYPE OF (solid_with_shape_element_pattern);
  row_count      : positive_integer;
  column_count   : positive_integer;
  row_spacing    : length_measure;
  column_spacing : length_measure;
WHERE
   WR1: (row_count * column_count) > 1;
END_ENTITY;
(*
```

Attribute definitions:

row_count: the number of rows in the pattern.

column_count: the number of columns in the pattern.

row_spacing: the distance between rows, positive if the columns are added in the positive y-direction and negative if in the opposite direction.

column_spacing: the distance between columns, positive if the columns are added in the positive x-direction and negative if in the opposite direction.

Formal propositions:

WR1: The product of the numbers of rows and columns shall be greater than 1, so that the pattern shall contain more than one element.

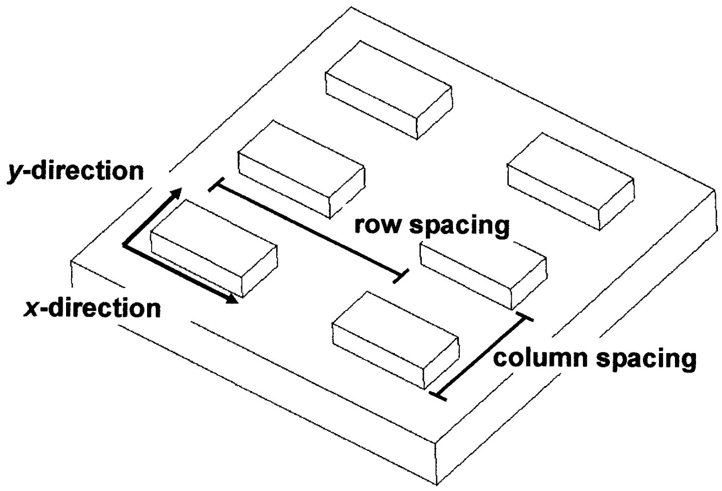


Figure 22 – Solid with rectangular pattern

4.4.45 solid_with_incomplete_circular_pattern

A **solid_with_incomplete_circular_pattern** is a type of **solid_with_circular_pattern** in which specified replicates have been omitted from the pattern.

NOTE This entity is illustrated diagrammatically in Figure 23. The angular spacing is 45 degrees. Considering the original shape configuration to be numbered 0, the replicates are numbered 1,2,. . . anticlockwise; the second replicate is therefore the one at the top of the figure, 90° from the base feature. If that instance and the one at 180° are omitted from the original pattern, the **omitted_instances** attribute will have the value [2,4].

EXPRESS specification:

```
*)
ENTITY solid_with_incomplete_circular_pattern
  SUBTYPE OF (solid_with_circular_pattern);
  omitted_instances : SET[1:?] OF positive_integer;
WHERE
  WR1: SIZEOF(omitted_instances) <
         SELF\solid_with_circular_pattern.replicate_count;
  WR2: SIZEOF(QUERY(q <* omitted_instances | q >
         SELF\solid_with_circular_pattern.replicate_count)) = 0;
END_ENTITY;
(*
```

Attribute definitions:

omitted_instances: the replicates to be omitted, counting anticlockwise from the base configuration.

Formal propositions:

WR1: The number of omitted replicates shall be less than the overall number of replicates.

WR2: No integer identifier in the set of omitted replicates shall have a value that exceeds the overall number of replicates.

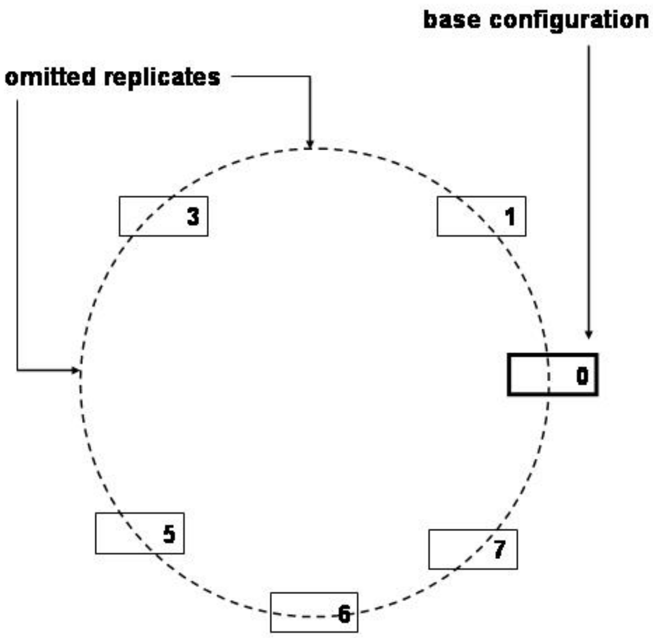


Figure 23 – Solid with incomplete circular pattern

4.4.46 solid_with_incomplete_rectangular_pattern

A **solid_with_incomplete_rectangular_pattern** is a type of **solid_with_rectangular_pattern** for which specified replicates have been omitted from the pattern. Each omitted replicate is identified by a tuple of the form $[m, n]$, where m and n denote row and column numbers respectively. The element corresponding to $[1, 1]$ is the shape configuration that has been replicated, and this element shall not be omitted.

NOTE Figure 24 diagrammatically illustrates the pattern in a **solid_with_incomplete_rectangular_pattern**. For this example the omitted instances will be represented by [[1,3],[3,1]].

EXPRESS specification:

```
*)
ENTITY solid_with_incomplete_rectangular_pattern
  SUBTYPE OF (solid_with_rectangular_pattern);
  omitted_instances : SET[1:?] OF LIST [2:2] OF positive_integer;
WHERE
  WR1: NOT([1,1] IN omitted_instances);
  WR2: SIZEOF(omitted_instances) <
        ((SELF\solid_with_rectangular_pattern.row_count *
```

```
            SELF\solid_with_rectangular_pattern.column_count) - 1);
  WR3: SIZEOF(QUERY(q <* omitted_instances |
          ((q[1] > SELF\solid_with_rectangular_pattern.row_count) OR
           (q[2] > SELF\solid_with_rectangular_pattern.column_count)))) = 0;
END_ENTITY;
(*
```

Attribute definitions:

omitted_instances: the instances to be omitted, identified in terms of row and column numbers.

Formal propositions:

WR1: The element corresponding to the tuple $[1, 1]$ shall not be omitted from the pattern.

WR2: The number of omitted replicates shall be less than the total number of replicates in the full pattern (that number does not include the original shape configuration, which is not a replicate).

WR3: The row index of any omitted replicate shall not exceed the row count of the rectangular pattern, and the column index shall not exceed the column count.

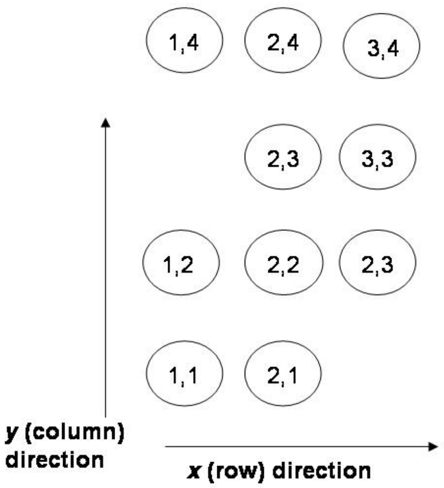


Figure 24 – Solid with incomplete rectangular pattern

4.4.47 thickened_face_solid

A **thickened_face_solid** is a type of **solid_model** created by adding thickness to a specified **face_surface**, **bounded_surface** or **surfaced_open_shell**. Material may be added on one or both sides of the original base element. The resulting solid is bounded by (i) surfaces lying on parallel offset surfaces of the base element (one of which may coincide with that element if its offset distance is zero) and (ii) ruled surfaces whose rulings are in the directions of the surface normal to the base element around its periphery.

Two offset distances are specified, **offset1** and **offset2**. Both define offset distances from the base element, positive and negative distances being measured in the directions of its positive and negative surface normals, respectively. The two offset distances shall not be equal, and one of them may be zero.

NOTE 1 Figure 25 illustrates the **thickened_face_solid** entity.

EXPRESS specification:

```
*)
ENTITY thickened_face_solid
  SUBTYPE OF (solid_model);
  base_element : generalized_surface_select;
  offset1      : length_measure;
  offset2      : length_measure;
WHERE
  WR1: NOT (('GEOMETRY_SCHEMA.SURFACE' IN TYPEOF(base_element)) AND
    (NOT ('GEOMETRY_SCHEMA.BOUNDED_SURFACE' IN TYPEOF(base_element))));
  WR2: offset1 <> offset2;
END_ENTITY;
(*
```

Attribute definitions:

base_element: the base element that is to be thickened.

offset1: the first offset distance.

offset2: the second offset distance.

Formal propositions:

WR1: If the element to be thickened is a surface, it shall be a bounded surface.

WR2: The two offset distances shall not have equal values.

Informal propositions:

IP1: The surface underlying the base element shall not be a bounded region of a single plane.

NOTE 2 The thickening of a planar base element would duplicate the capability provided by **extruded_face_-solid_with_trim_conditions**, as defined in clause 4.4.48.

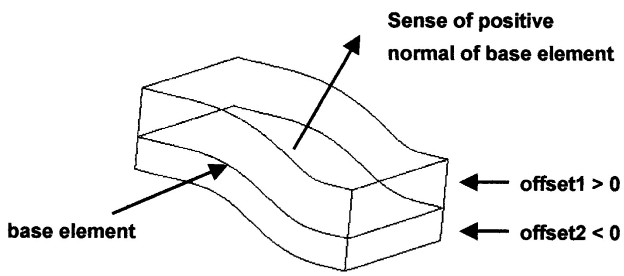


Figure 25 – Thickened face solid

4.4.48 extruded_face_solid_with_trim_conditions

An **extruded_face_solid_with_trim_conditions** is a type of **extruded_face_solid** (see ISO 10303-42), whose basic form is characterized by the distance between its two planar ends. The use of the SELECT type **trim_condition_select** allows the volume defined by an **extruded_face_solid** to be trimmed by the use of alternative end conditions. It is not required that the same type of condition is imposed at both ends of the extrusion.

The interpretations of the different specific trimming conditions are as follows:

plane_angle_measure: not allowed for this entity;

length_measure: the generated solid is trimmed at a specified distance from the defining face;

generalized_surface_select: the generated solid is trimmed where it meets, or at a specified distance before it meets, a specified **surface**, **face_surface** or connected set of **face_surface**s;

solid_model: the generated solid is trimmed where it meets, or at a specified distance before it meets, the boundary of a specified solid model.

The supertype **extruded_face_solid** has an attribute **depth**, of type **positive_length_measure**, that specifies the extent of the original extrusion, measured from the defining cross-sectional face. For an instance of **extruded_face_solid_with_trim_conditions** trimming is interpreted in a very general sense, and the trimmed volume may extend beyond the original volume defined by the supertype, in either direction. In particular, if either termination condition is of type **length_measure** its value may exceed that of the **depth** attribute of the supertype, so that the original extrusion is extended in the positive direction, or its value may be negative, in which case the original cross-sectional face is embedded in the resulting volume rather than forming one of its bounding faces. Similar extensions of the original volume may be obtained by the use of the other types of termination condition.

Some CAD systems provide additional semantics for trimming extrusions, in terms of termination conditions such as the following:

blind: the extrusion terminates at an extent determined by the value of a trim condition of type **length_measure**, before it meets any other surface in the model;

offset: the extrusion terminates at a given distance from a specified surface in the model;

up to next: the extrusion terminates when it first meets a surface in the model;

through all: the extrusion extends completely through the entire model.

Attributes **first_trim_intent** and **second_trim_intent** allow for the capture of such descriptive conditions. The explicit trim conditions specified by the attributes **first_trim_condition** and **second_trim_condition**, together with the use of the offset capability, are adequate to ensure that the model as reconstructed in the receiving system is identical with that in the sending system, subject to numerical tolerances. However, the additional transmission of trim intent may be used to ensure correct behaviour if the received model is subsequently edited. The available trim intent values, as defined by the enumerated type **trim_intent** (see clause 4.3.6), includes the value **unspecified** for use with systems not having the described capability.

When either trim intent has the value **offset**, additional attributes **first_offset** and **second_offset** allow the required offset distances to be specified. These are the distances between the planar end of the extrusion and the surface or solid boundary used to define its extent.

EXPRESS specification:

```
*)
ENTITY extruded_face_solid_with_trim_conditions
  SUPERTYPE OF (ONEOF
                 (extruded_face_solid_with_draft_angle,
                  extruded_face_solid_with_multiple_draft_angles))
  SUBTYPE OF (extruded_face_solid);
  first_trim_condition  : trim_condition_select;
  second_trim_condition : trim_condition_select;
  first_trim_intent     : trim_intent;
  second_trim_intent    : trim_intent;
  first_offset          : non_negative_length_measure;
  second_offset         : non_negative_length_measure;
WHERE
  WR1: NOT(('MEASURE_SCHEMA.PLANE_ANGLE_MEASURE'
         IN TYPEOF(first_trim_condition)) OR
         ('MEASURE_SCHEMA.PLANE_ANGLE_MEASURE'
         IN TYPEOF(second_trim_condition)));
  WR2: NOT ((('MEASURE_SCHEMA.LENGTH_MEASURE'
         IN TYPEOF(first_trim_condition)) AND
         ((first_trim_intent = offset)
         OR (first_trim_intent = up_to_next))) OR
         (('MEASURE_SCHEMA.LENGTH_MEASURE'
         IN TYPEOF(second_trim_condition)) AND
         ((second_trim_intent = offset)
         OR (second_trim_intent = up_to_next))));
  WR3: NOT (((NOT ('MEASURE_SCHEMA.LENGTH_MEASURE'
         IN TYPEOF(first_trim_condition))) AND
         ((first_trim_intent = blind)
         OR (first_trim_intent = through_all))) OR
         ((NOT('MEASURE_SCHEMA.LENGTH_MEASURE'
         IN TYPEOF(second_trim_condition))) AND
         ((second_trim_intent = blind)
         OR (second_trim_intent = through_all))));
  WR4: (((first_trim_intent = offset) AND (first_offset > 0)) XOR
         ((first_trim_intent <> offset) AND (first_offset = 0))) AND
         (((second_trim_intent = offset) AND (second_offset > 0)) XOR
         ((second_trim_intent <> offset) AND (second_offset = 0)));
  WR5: NOT((('MEASURE_SCHEMA.LENGTH_MEASURE'
         IN TYPEOF(first_trim_condition)) AND
            ('MEASURE_SCHEMA.LENGTH_MEASURE'
         IN TYPEOF(second_trim_condition))) AND
         (first_trim_condition = second_trim_condition));
END_ENTITY;
(*
```

Attribute definitions:

first_trim_condition: the first trim condition on the extrusion.

second_trim_condition: the second trim condition on the extrusion.

first_trim_intent: the intended semantics of the first trim condition.

second_trim_intent: the intended semantics of the second trim condition.

first_offset: the distance between the first planar end of the extrusion and a surface or solid boundary used to define its extent.

second_offset: the distance between the second planar end of the extrusion and a surface or solid boundary used to define its extent.

Formal propositions:

WR1: The selection **plane_angle_measure** shall not be used as a trimming condition with this entity.

WR2: If either of the trim conditions is a length measure, the corresponding trim intent shall not have the value **offset** or **up_to_next**.

WR3: If either of the trim conditions is not a length measure, the corresponding trim intent shall not have the value **blind** or **through_all**.

WR4: At either end of the extrusion, if the trim intent has the value **offset** then the corresponding offset value shall be positive; for any other value of the trim intent, the corresponding offset value shall be zero.

WR5: If both trim conditions are length measures, their values shall not be equal.

NOTE 1 A converse interpretation of **WR3** is that if the trim condition is termination by a generalized surface or the surface of a solid then the trim intent can be **up_to_next**, **offset** or **unspecified**. **WR4** then permits the specification of a positive offset distance if the trim intent has the value **offset**.

NOTE 2 Figure 26 illustrates the **extruded_face_solid_with_trim_conditions**. The circular-section extrusion extends between two pre-existing solids, the lower block and the upper sculptured volume. This figure also illustrates the following entity, **extruded_face_solid_with_draft_angle**, because the extrusion has a draft angle.

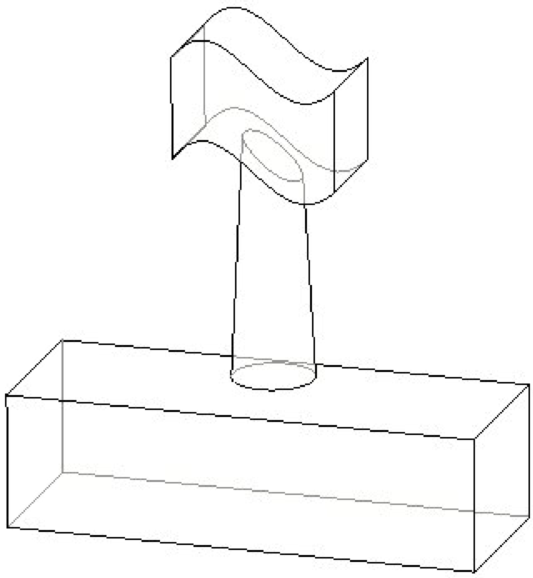


Figure 26 – Extruded face solid with draft angle

4.4.49 extruded_face_solid_with_draft_angle

An **extruded_face_solid_with_draft_angle** is a type of **extruded_face_solid_with_trim_conditions** in which a draft angle is applied along the length of the extrusion. The cross-section of the extrusion at the position of defining face is identical with the area of that face, and is unaffected by the draft angle.

At any point on the faces generated by the extrusion, the draft angle is the angle between the tangent plane to the surface and the direction of extrusion. If the angle is positive the cross sectional area of the solid will decrease in the positive direction of extrusion, and vice versa.

NOTE Figure 26 in clause 4.4.48 illustrates the **extruded_face_solid_with_draft_angle**, the extrusion being trimmed where it meets the surfaces of two solids.

EXPRESS specification:

```
*)
ENTITY extruded_face_solid_with_draft_angle
  SUBTYPE OF (extruded_face_solid_with_trim_conditions);
  draft_angle : plane_angle_measure;
WHERE
  WR1: draft_angle <> 0;
END_ENTITY;
(*
```

Attribute definitions:

draft_angle: the angle between tangent plane of the surfaces generated by the extrusion and the direction of extrusion.

Formal propositions:

WR1: The draft angle shall not have the value zero (for that case the use of the supertype **extruded_-face_solid_with_trim_conditions** would be appropriate).

Informal propositions:

IP1: The specified draft angle and trim conditions shall not give rise to any intersection between the faces generated by the extrusion, other than along those edges of the extrusion swept out by vertices of the original face.

4.4.50 extruded_face_solid_with_multiple_draft_angles

An **extruded_face_solid_with_multiple_draft_angles** is a type of **extruded_face_solid_with_trim_conditions**, in which different edges of the swept face have different associated draft angles. These angles are defined in the same manner as for the **extruded_face_solid_with_draft_angle**. If any edge of the swept face has no associated draft angle, the value of the angle for that edge shall be taken as zero.

The cross-section of the extrusion at the position of defining face is identical with the area of that face, and is unaffected by the draft angles.

EXPRESS specification:

```
*)
ENTITY extruded_face_solid_with_multiple_draft_angles
  SUBTYPE OF (extruded_face_solid_with_trim_conditions);
  drafted_edges : LIST[2:?] OF SET[1:?] OF edge_curve;
  draft_angles  : LIST[2:?] OF plane_angle_measure;
WHERE
  WR1: SIZEOF(drafted_edges) = SIZEOF(draft_angles);
  WR2: SIZEOF(QUERY(q <* draft_angles | q = 0)) = 0;
  WR3: SIZEOF(QUERY(q <* drafted_edges | (SIZEOF(QUERY(r <* q | NOT
         (SELF\swept_face_solid.swept_face IN
          using_items(r,[])))) > 0))) = 0;
END_ENTITY;
(*
```

Attribute definitions:

drafted_edges: a list of sets of edges of the extruded face, each edge in a set having the same associated draft angle.

draft_angles: the draft angles specified for each set of edges of the extruded face.

Formal propositions:

WR1: The number of sets of drafted edges shall equal the number of draft angles specified.

WR2: No member of the list **draft_angles** shall have the value zero, which is the default value applying when no other value is specified.

WR3: Every member of every set of **edge_curve** instances referenced by the attribute **drafted_edges** shall occur in the boundary of the swept face.

Informal propositions:

IP1: The lists of sets of drafted edges and draft angles shall be linked in the sense that the nth set of drafted edges is associated with the nth draft angle value.

IP2: The specified draft angles and end conditions shall not give rise to any intersection between the faces generated by the extrusion, other than along those edges of the extrusion swept out by vertices of the original face.

4.4.51 revolved_face_solid_with_trim_conditions

A **revolved_face_solid_with_trim_conditions** is a type of **revolved_face_solid** as defined in ISO 10303-42, which in its basic form is characterized by the angle between its two planar ends. The use of the SELECT type **termination_condition_select** allows the specification of alternative end conditions on the generated volume. It is not required that the same type of condition is imposed at both ends of the extrusion.

The interpretations of the different specific end conditions are as follows:

plane_angle_measure: the generated solid terminates or is trimmed at specified angular distances from the referenced face;

length_measure: not allowed for this entity;

generalized_surface_select: the generated solid is trimmed where it meets a specified **surface**, **face_surface** or connected set of **face_surface**s;

solid_model: the generated solid is trimmed where it meets the boundary of a specified solid model.

The supertype **revolved_face_solid** has an attribute **angle**, of type **plane_angle_measure**, that specifies the angular extent of the generated solid, measured from the defining cross-sectional face. For an instance of **revolved_face_solid_with_trim_conditions** trimming is interpreted in a very general sense, and the trimmed volume may extend in angular terms beyond the original volume defined by the supertype, in either direction. In particular, if either termination condition is of type **plane_angle_measure** its value may exceed that of the **angle** attribute of the supertype, so that the original extrusion is extended in the positive direction, or its value may be negative, in which case the original cross-sectional face is embedded in the resulting volume rather than forming one of its bounding faces. Similar extensions of the original volume may be obtained by the use of the other types of termination condition.

EXPRESS specification:

```
*)
ENTITY revolved_face_solid_with_trim_conditions
  SUBTYPE OF(revolved_face_solid);
  first_trim_condition  : trim_condition_select;
  second_trim_condition : trim_condition_select;
WHERE
  WR1: NOT (('MEASURE_SCHEMA.LENGTH_MEASURE'
         IN TYPEOF(first_trim_condition))
         OR ('MEASURE_SCHEMA.LENGTH_MEASURE'
         IN TYPEOF(second_trim_condition)));
  WR2: NOT((('MEASURE_SCHEMA.PLANE_ANGLE_MEASURE'
         IN TYPEOF(first_trim_condition)) AND
            ('MEASURE_SCHEMA.PLANE_ANGLE_MEASURE'
         IN TYPEOF(second_trim_condition))) AND
         (first_trim_condition = second_trim_condition));
END_ENTITY;
(*
```

Attribute definitions:

first_trim_condition: the first trim condition on the solid of revolution.

second_trim_condition: the second trim condition on the solid of revolution.

Formal propositions:

WR1: The selection **length_measure** shall not be used for either trim condition in an instance of this entity.

WR2: If both trim conditions are angle measures, their values shall not be equal.

Informal propositions:

IP1: For no instance of this entity shall the generated solid exhibit self-intersection.

NOTE Self-intersection will occur if the angle of revolution exceeds 360° (2π radians), but it is not possible to write this condition in formal terms for general trim conditions.

4.4.52 auxiliary_geometric_instance_aggregator

An **auxiliary_geometric_instance_aggregator** is a type of **geometric_representation_item** collecting instances associated with a shape creation operation that are not directly or indirectly referenced by it.

EXAMPLE In a sketch-based solid of extrusion the sketch concerned may be based on an instance of **face_surface** (see ISO 10303-108). The curves defining the edges of the face-surface may be subject to explicit constraints. In the corresponding ISO 10303 representation the instances defining the explicit constraints have direct references to the curve instances they constrain, but the **face_surface** instance, for historical reasons, has no direct reference to the constraints that apply to it. Thus in order to determine which shape element of a transferred representation a particular constraint applies to, it is necessary to search the entire model to determine the shape element or elements that contain the curves referenced by the constraint. The use of an instance of **auxiliary_geometric_instance_aggregator** will permit a direct association to be established between the extrusion and the constraints applying to its defining sketch.

NOTE 1 The use of the entity defined in this clause is not mandatory in the transfer of a procedural shape model, because the information it summarizes may be obtained by exhaustive search processes on the data in an ISO 10303 procedural model representation.

NOTE 2 By contrast with ISO 10303 representations of shape, which are global in nature, the data structures of most CAD systems are based on local shape configurations such as sketches and shape features. These local configurations provide direct references to all auxiliary elements involved in their complete specification. Thus the information content of an instance of **auxiliary_geometric_instance_aggregator** is readily available at the preprocessing stage, and its use in postprocessing will significantly reduce the amount of searching needed to establish correct relationships in the transferred model.

EXPRESS specification:

```
*)
ENTITY auxiliary_geometric_instance_aggregator
  SUBTYPE OF (geometric_representation_item);
  shape_element   : geometric_representation_item;
  auxiliary_items : SET[1:?] OF geometric_representation_item;
WHERE
  WR1: SIZEOF(QUERY(q <* auxiliary_items |
              shape_element IN using_items(q,[]))) = 0;
END_ENTITY;
(*
```

Attribute definitions:

shape_element: the creation operation for a shape element of a procedural representation.

auxiliary_items: the auxiliary items that reference the shape element but are not directly or indirectly referenced by it.

Formal propositions:

WR1: The **shape_element** shall not reference any member of the set of **auxiliary_items**, either directly or indirectly.

4.5 Solid shape element function definitions

4.5.1 check_continuous_edges

The function **check_continuous_edges** checks whether the members of a list of distinct edges, taken in order of their occurrence in the list, form a connected track.

NOTE This function is analogous to the function **path_head_to_tail** as defined in ISO 10303-42. However, the input to **path_head_to_tail** is a **path** composed of instances of **oriented_edge**, whereas the input to **check_continuous_edges** is a list of instances of **edge_curve**. The lack of consistent orientation of these edges requires the use of slightly more complex logic to establish their connectivity.

EXPRESS specification:

```
*)
FUNCTION check_continuous_edges (edges : LIST OF UNIQUE edge_curve)
                                        : BOOLEAN;

  LOCAL
    i           : INTEGER;
    next_vertex : vertex;
  END_LOCAL;

  -- first check whether there is only one edge in the list: in this
  -- case there is no connectivity to be checked.

  IF (SIZEOF(edges) = 1)
  THEN RETURN(TRUE);
  END_IF;

  -- otherwise, establish the matching vertices of edges 1 and 2 in
  -- the list, and determine the vertex of edge 2 to which edge 3,
  -- must be connected, if there are more than two edges in the list.

  IF ((edges[2].edge_start :=: edges[1].edge_end)
    XOR (edges[2].edge_start :=: edges[1].edge_start))
  THEN next_vertex := edges[2].edge_end;
  ELSE
    IF ((edges[2].edge_end :=: edges[1].edge_end)
      XOR (edges[2].edge_end :=: edges[1].edge_start))
    THEN next_vertex := edges[2].edge_start;
```

```
    ELSE RETURN(FALSE); -- no match between any vertices of edges 1 and 2
    END_IF;
  END_IF;

  -- exit if there are only two edges and they are connected

  IF (SIZEOF(edges) = 2)
  THEN RETURN(TRUE);
  END_IF;

  -- otherwise, check that any remaining edges are connected in list order.

  REPEAT i := 3 TO HIINDEX(edges);
    IF (edges[i].edge_start :=: next_vertex)
    THEN next_vertex := edges[i].edge_end;
    ELSE
      IF (edges[i].edge_end :=: next_vertex)
      THEN next_vertex := edges[i].edge_start;
      ELSE RETURN(FALSE); -- no match is found.
      END_IF;
    END_IF;
  END_REPEAT;
  RETURN(TRUE); -- all edges have now been successfully checked.

END_FUNCTION;
(*
```

Argument definitions:

edges: the list of distinct edges whose connectivity is to be checked.

4.5.2 compute_total_depth

The function **compute_total_depth** calculates the overall depth of the hole in an instance of **solid_with_-stepped_round_hole** in terms of the depths of its individual hole segments.

EXPRESS specification:

```
*)
FUNCTION compute_total_depth (swsrh : solid_with_stepped_round_hole)
                                    : positive_length_measure;

  LOCAL
    i  : INTEGER;
    td : positive_length_measure := 0;
  END_LOCAL;

  REPEAT i := 1 TO swsrh.segments;
    td := td + swsrh.segment_depths[i];
  END_REPEAT;
  RETURN(td);

END_FUNCTION;
(*
```

Argument definitions:

swsrh: the **solid_with_stepped_round_hole** for which the total depth of the hole is to be calculated.

4.5.3 validate_countersink_radii

The function **validate_countersink_radii** checks that the values of the countersink radii specified in an instance of **solid_with_stepped_round_hole_and_conical_transitions** are consistent with the radii of the hole segments, and returns the value TRUE if so. No limit is imposed on the radius of the initial countersink, at the hole entry, or (in the case of a through hole) on the radius of the last countersink at the hole exit, though in these two cases the apex angles of any countersinks defined there are required to be positive and negative, respectively. Otherwise, at all internal changes of radius between hole segments the countersink radius is required to lie between the smaller and larger radii of the two adjoining hole segments, and the signs of the apex angles of the conical transitions are required to be consistent with the increase or decrease of hole radius at each transition.

EXPRESS specification:

```
*)
FUNCTION validate_countersink_radii
  (cskhole : solid_with_stepped_round_hole_and_conical_transitions)
           : BOOLEAN;

  LOCAL
    i,j              : INTEGER;
    n                : INTEGER := 1 +
                         cskhole\solid_with_stepped_round_hole.segments;
    smaller, larger : positive_length_measure;
  END_LOCAL;

  REPEAT i := 1 TO SIZEOF(cskhole.conical_transitions);

  -- First check whether transition i applies to the entry of the hole or
  -- the exit of a blind hole - those cases only need to be checked for
  -- the sign of the cone apex angle.

  IF (((cskhole.conical_transitions[i].transition_number = 1)
       AND (cskhole.conical_transitions[i].cone_apex_angle < 0))
    XOR ((cskhole.conical_transitions[i].transition_number = n)
        AND (cskhole.conical_transitions[i].cone_apex_angle > 0)))
  THEN RETURN(FALSE);
  ELSE
    IF ((cskhole.conical_transitions[i].transition_number <> 1)
      AND (cskhole.conical_transitions[i].transition_number <> n))
    THEN

  -- For all remaining transitions, check that the cone base radius
  -- lies in the range of validity.

      BEGIN
        j := cskhole.conical_transitions[i].transition_number;
        IF cskhole\solid_with_stepped_round_hole.segment_radii[j]
          > cskhole\solid_with_stepped_round_hole.segment_radii[j-1]
        THEN
```

```
          BEGIN
            IF (cskhole.conical_transitions[i].cone_apex_angle > 0)
            THEN RETURN(FALSE);
            END_IF;
            larger
              := cskhole\solid_with_stepped_round_hole.segment_radii[j];
            smaller
              := cskhole\solid_with_stepped_round_hole.segment_radii[j-1];
          END;
        ELSE
          BEGIN
            IF (cskhole.conical_transitions[i].cone_apex_angle < 0)
            THEN RETURN(FALSE);
            END_IF;
            larger
              := cskhole\solid_with_stepped_round_hole.segment_radii[j-1];
            smaller
              := cskhole\solid_with_stepped_round_hole.segment_radii[j];
          END;
          IF ((cskhole.conical_transitions[i].cone_base_radius > larger)
            OR (cskhole.conical_transitions[i].cone_base_radius < smaller))
          THEN RETURN(FALSE);
          END_IF;
        END_IF;
      END;
    END_IF;
  END_IF;
  END_REPEAT;
  RETURN(TRUE);

END_FUNCTION;
(*
```

Argument definitions:

swsrh: the **solid_with_stepped_round_hole_and_conical_transitions** for which the validity of the countersink radii is to be checked.

EXPRESS specification:

```
*)
END_SCHEMA; -- solid_shape_element_schema
(*
```

Annex A
(normative)

Short names of entities

Table A.1 provides the short names of entities specified in this part of ISO 10303. Requirements on the use of short names are found in the implementation methods included in ISO 10303.

Table A.1 – Short names of entities

Entity data type names	Short names
auxiliary_geometric_instance_aggregator	AGIA
complex_shelled_solid	CMSHSL
conical_stepped_hole_transition	CSHT
double_offset_shelled_solid	DOSS
edge_blended_solid	EDBLSL
extruded_face_solid_with_draft_angle	EFSWDA
extruded_face_solid_with_multiple_draft_angles	EFSWMD
extruded_face_solid_with_trim_conditions	EFSWTC
modified_solid	MDFSLD
modified_solid_with_placed_configuration	MSWPC
offset_face_solid	OFFCSL
revolved_face_solid_with_trim_conditions	RFSWTC
sculptured_solid	SCLSLD
sculptured_solid_with_selection	SSWS
shelled_solid	SHLSLD
solid_with_angle_based_chamfer	SWABC
solid_with_chamfered_edges	SWCE
solid_with_circular_pattern	SWC1
solid_with_circular_pocket	SWCP
solid_with_circular_protrusion	SWC0
solid_with_conical_bottom_round_hole	SWCBRH
solid_with_constant_radius_edge_blend	SWCREB
solid_with_curved_slot	SWCS
solid_with_depression	SLWTDP
solid_with_double_offset_chamfer	SWDOC
solid_with_flat_bottom_round_hole	SWFBRH
solid_with_general_pocket	SWGP
solid_with_general_protrusion	SWG0
solid_with_groove	SLWTGR
solid_with_hole	SLWTHL
solid_with_incomplete_circular_pattern	SWICP
solid_with_incomplete_rectangular_pattern	SWIRP
solid_with_pocket	SLWTPC
solid_with_protrusion	SLWTPR
solid_with_rectangular_pattern	SWR1
solid_with_rectangular_pocket	SWRP

Table A.1 – (continued)

Entity data type names	Short names
solid_with_rectangular_protrusion	SWR0
solid_with_shape_element_pattern	SWSEP
solid_with_single_offset_chamfer	SWSOC
solid_with_slot	SLWTSL
solid_with_spherical_bottom_round_hole	SWSBRH
solid_with_stepped_round_hole	SWSRH
solid_with_stepped_round_hole_and_conical_transitions	SWSRHA
solid_with_straight_slot	SWSS
solid_with_tee_section_slot	SWT0
solid_with_through_depression	SWTD
solid_with_trapezoidal_section_slot	SWTSS
solid_with_variable_radius_edge_blend	SWVREB
surfaced_open_shell	SROPSH
thickened_face_solid	THFCSL
track_blended_solid	TRBLSL
track_blended_solid_with_end_conditions	TBSWEC

Annex B
(normative)

Information object registration

B.1 Document identification

To provide for unambiguous identification of an information object in an open system, the object identifier

```
{ iso standard 10303 part(111) 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.

B.2 Schema identification

To provide for unambiguous identification of the solid-shape-element-schema in an open information system, the object identifier

```
{ iso standard 10303 part(111) version(1) schema(1)
        solid-shape-element-schema(1) }
```

is assigned to the **solid_shape_element_schema** (see clause 4). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Annex C
(informative)

Computer interpretable listings

This annex references a listing of the EXPRESS entity data type names and corresponding short names as specified in this part of ISO 10303. It also references 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 TC184/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 D
(informative)

EXPRESS-G diagrams

The diagrams in this annex correspond to the EXPRESS schemas specified in this part of ISO 10303. The diagrams use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined in annex D of ISO 10303-11:2004.

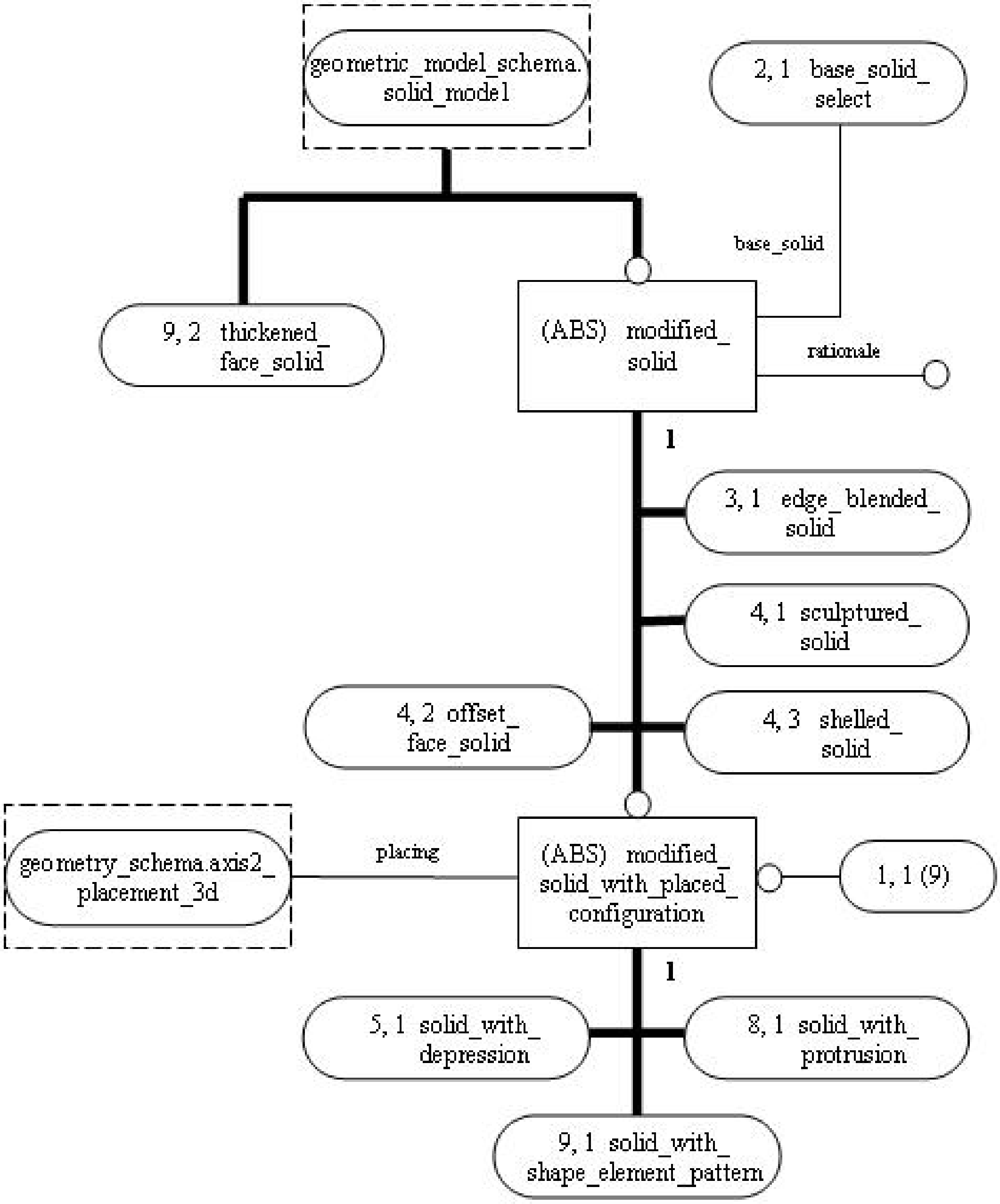


Figure D.1 – solid_shape_element_schema – EXPRESS-G diagram 1 of 11

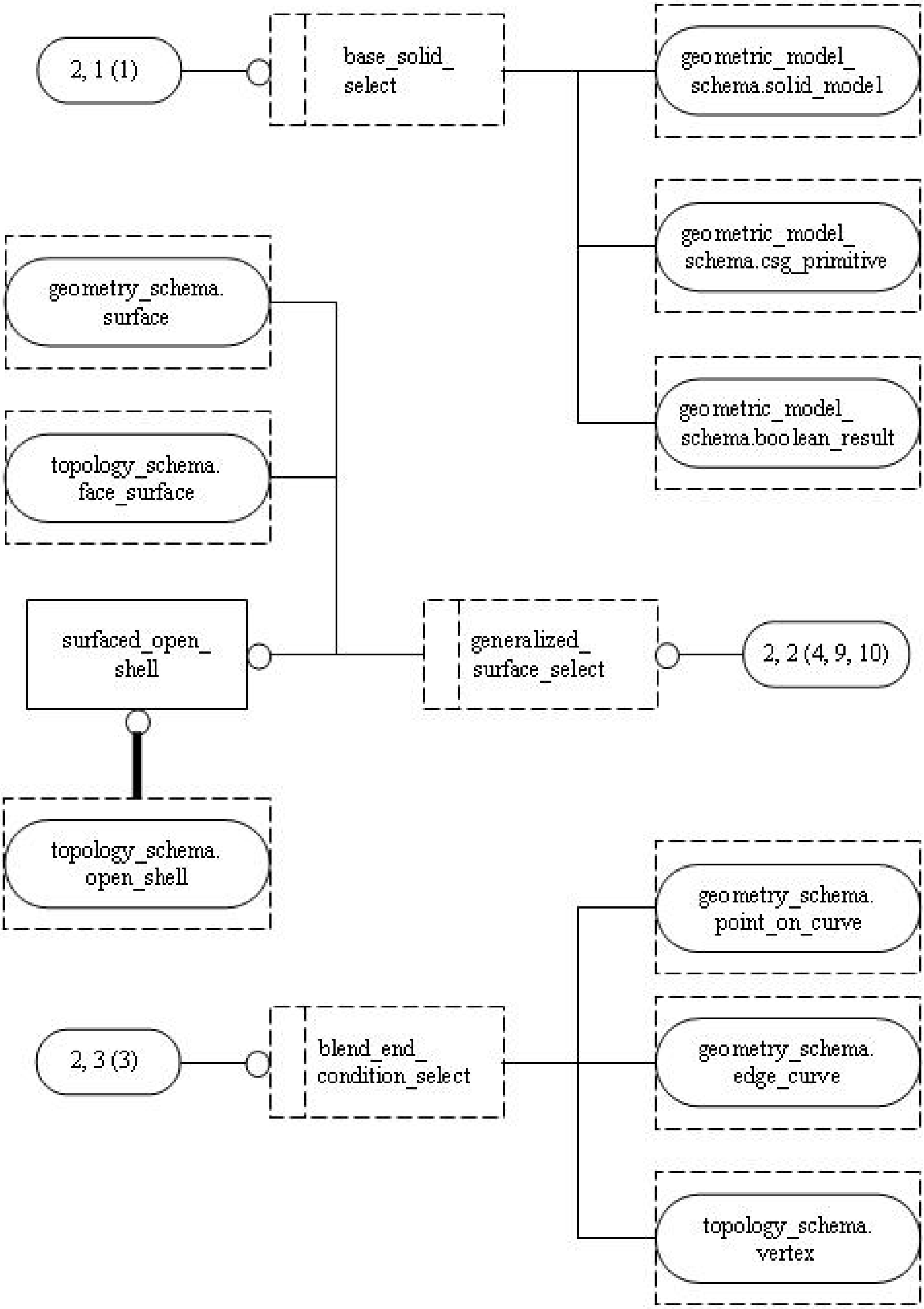


Figure D.2 – solid_shape_element_schema – EXPRESS-G diagram 2 of 11

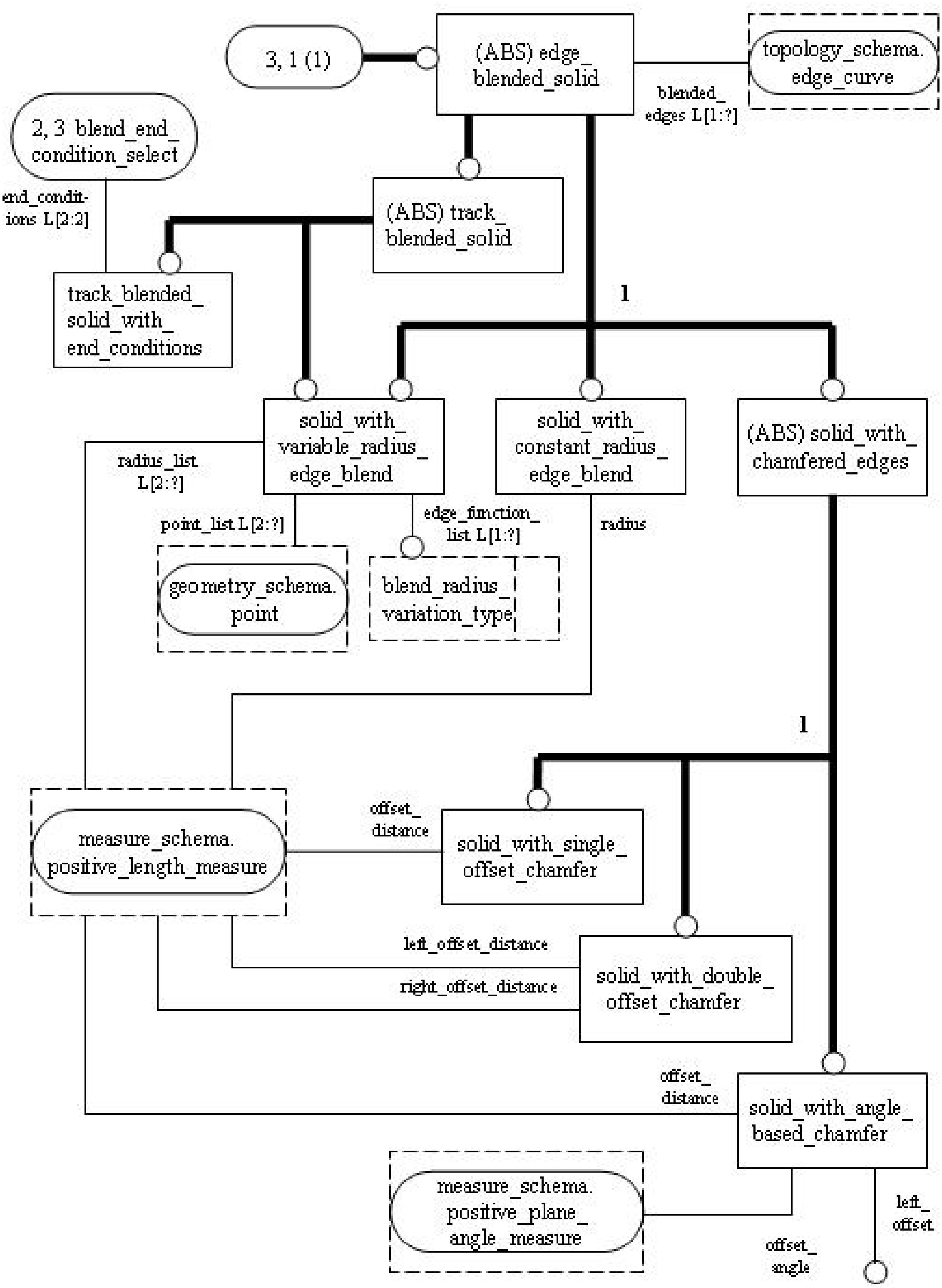


Figure D.3 – solid_shape_element_schema – EXPRESS-G diagram 3 of 11

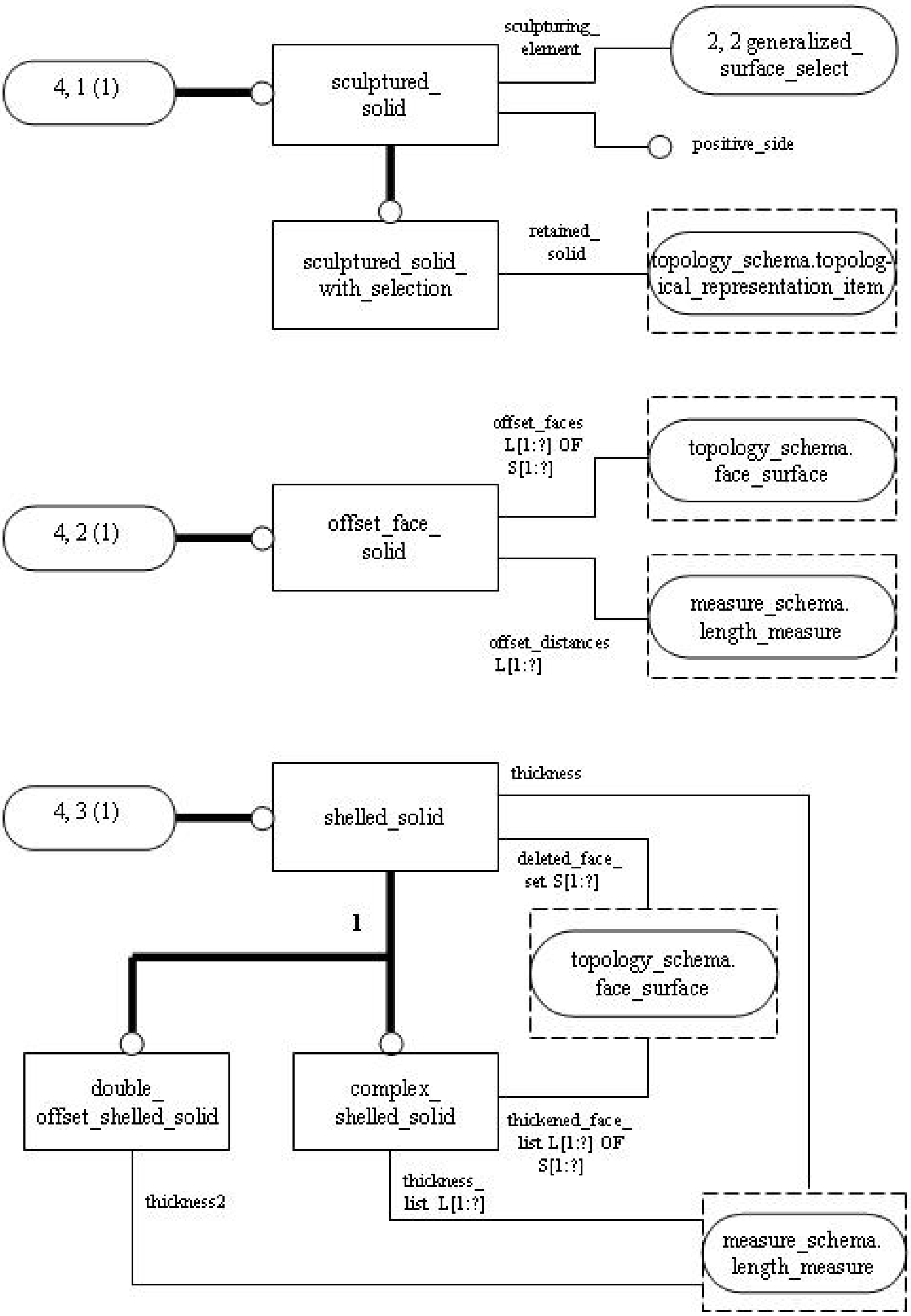


Figure D.4 – solid_shape_element_schema – EXPRESS-G diagram 4 of 11

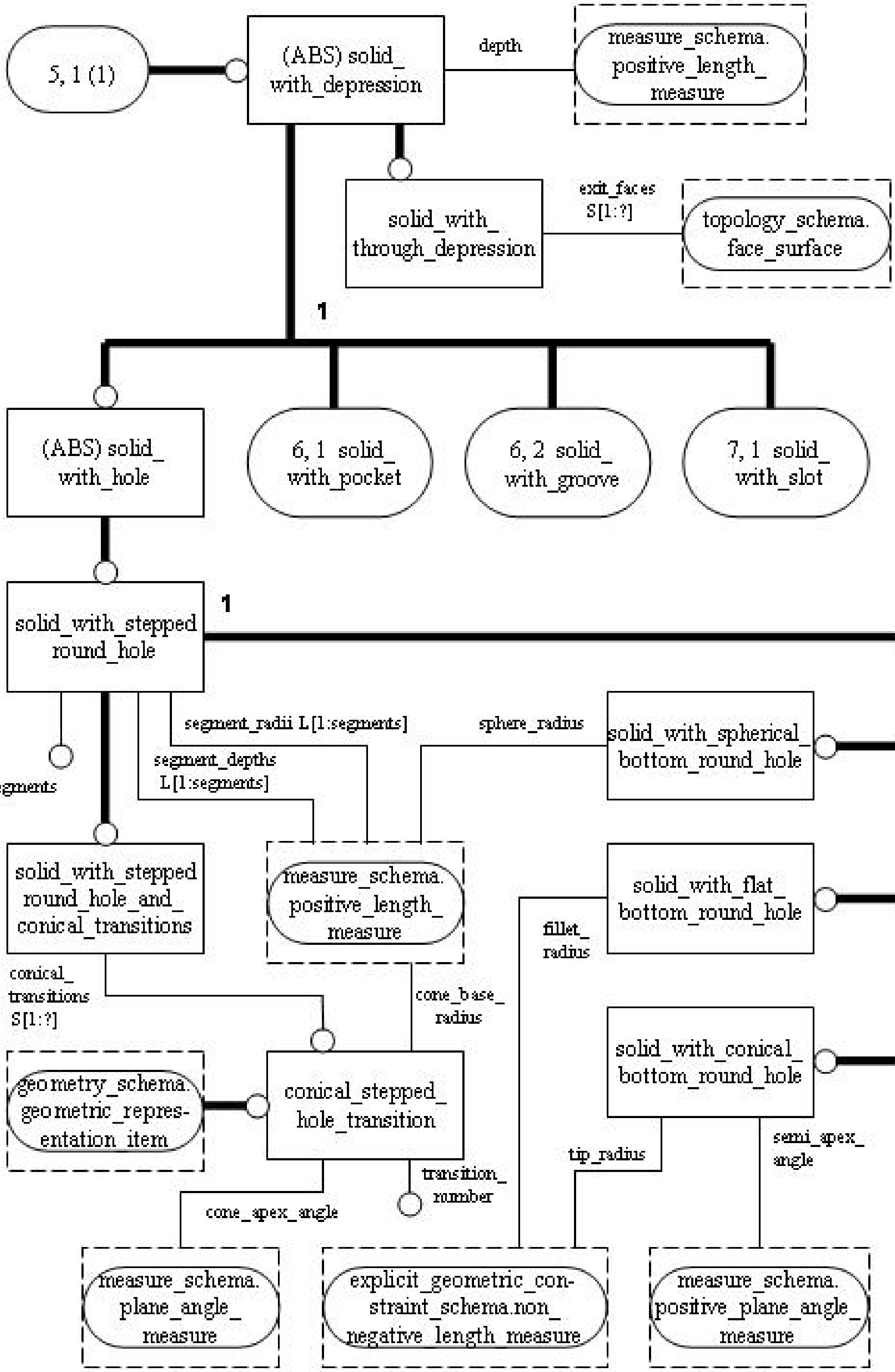


Figure D.5 – solid_shape_element_schema – EXPRESS-G diagram 5 of 11

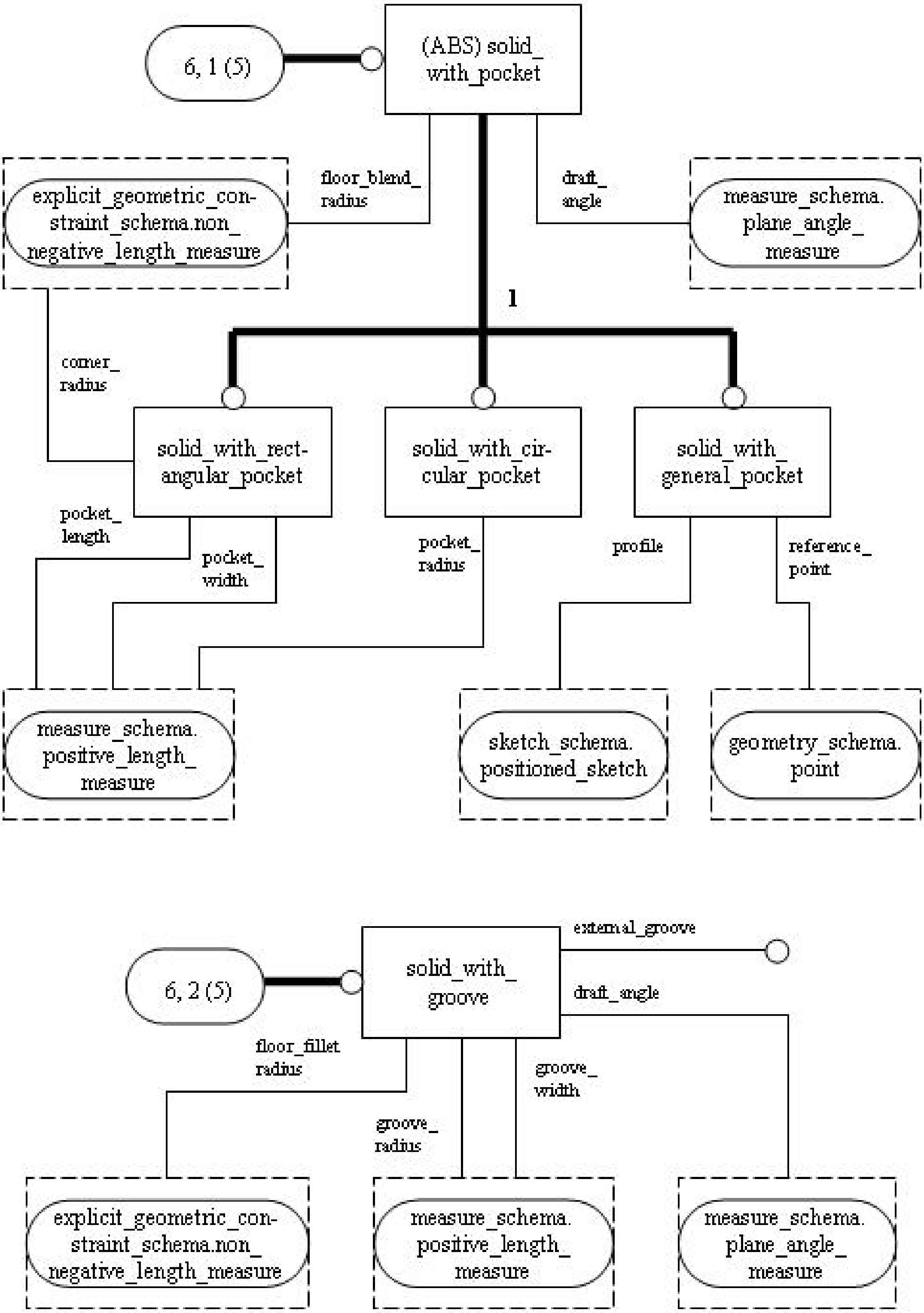


Figure D.6 – solid_shape_element_schema – EXPRESS-G diagram 6 of 11

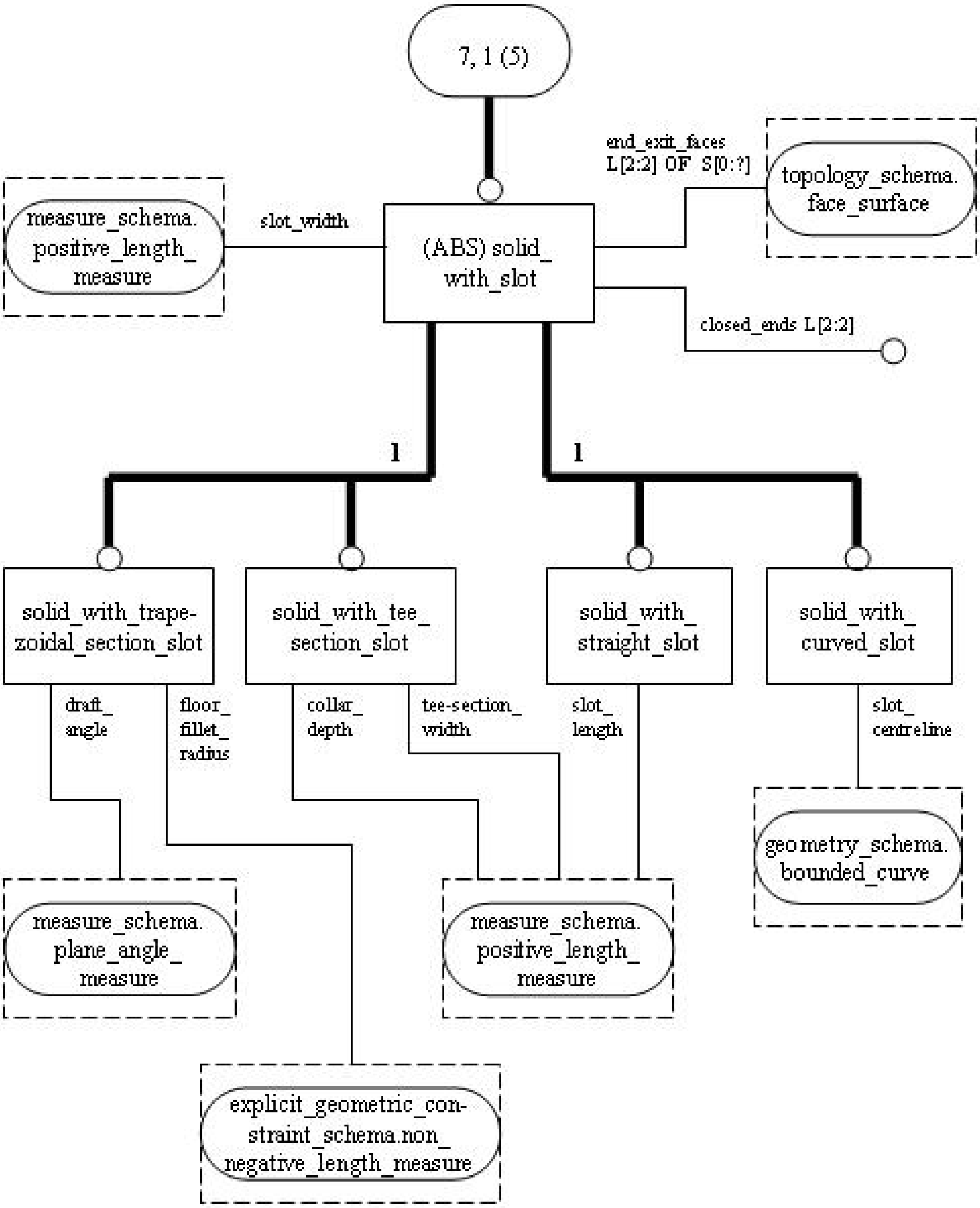


Figure D.7 – solid_shape_element_schema – EXPRESS-G diagram 7 of 11

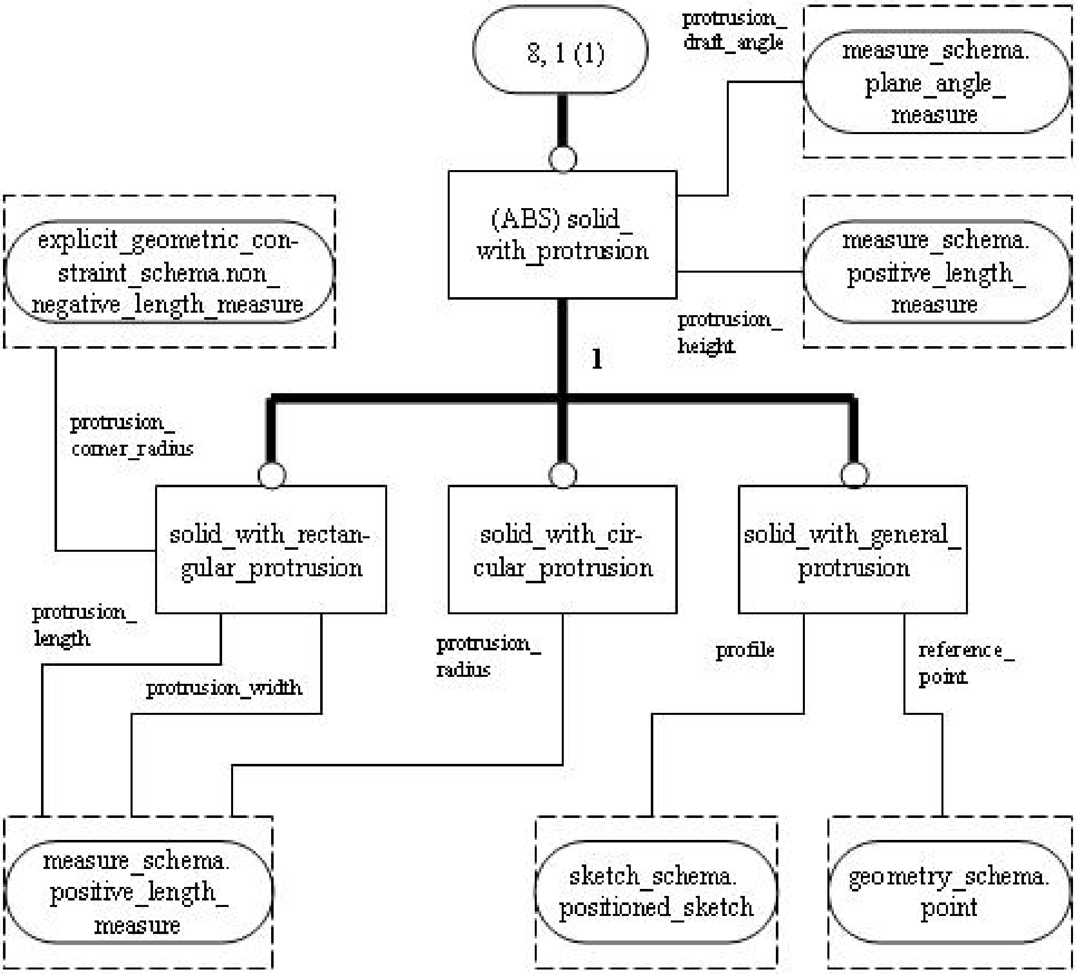


Figure D.8 – solid_shape_element_schema – EXPRESS-G diagram 8 of 11

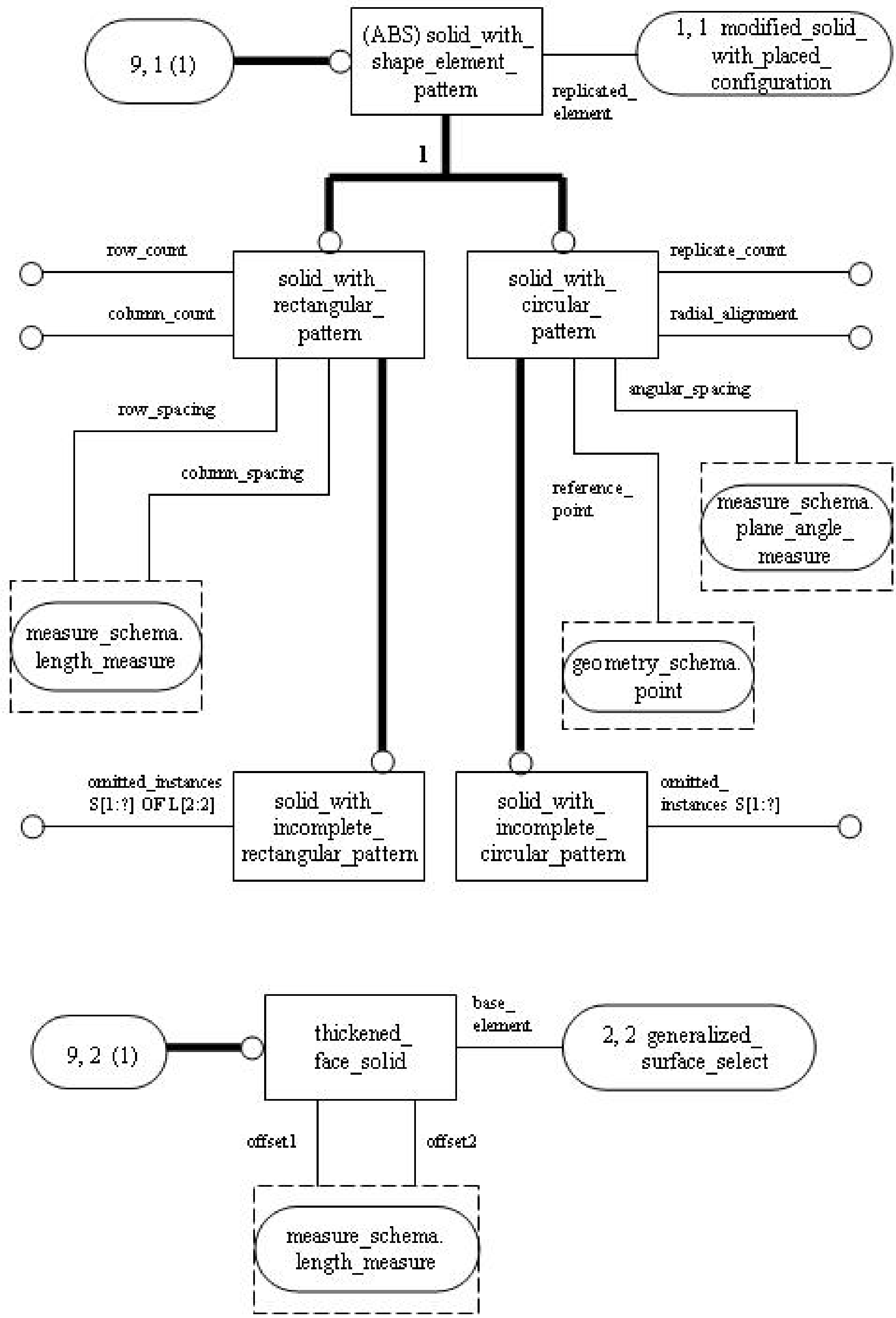


Figure D.9 – solid_shape_element_schema – EXPRESS-G diagram 9 of 11

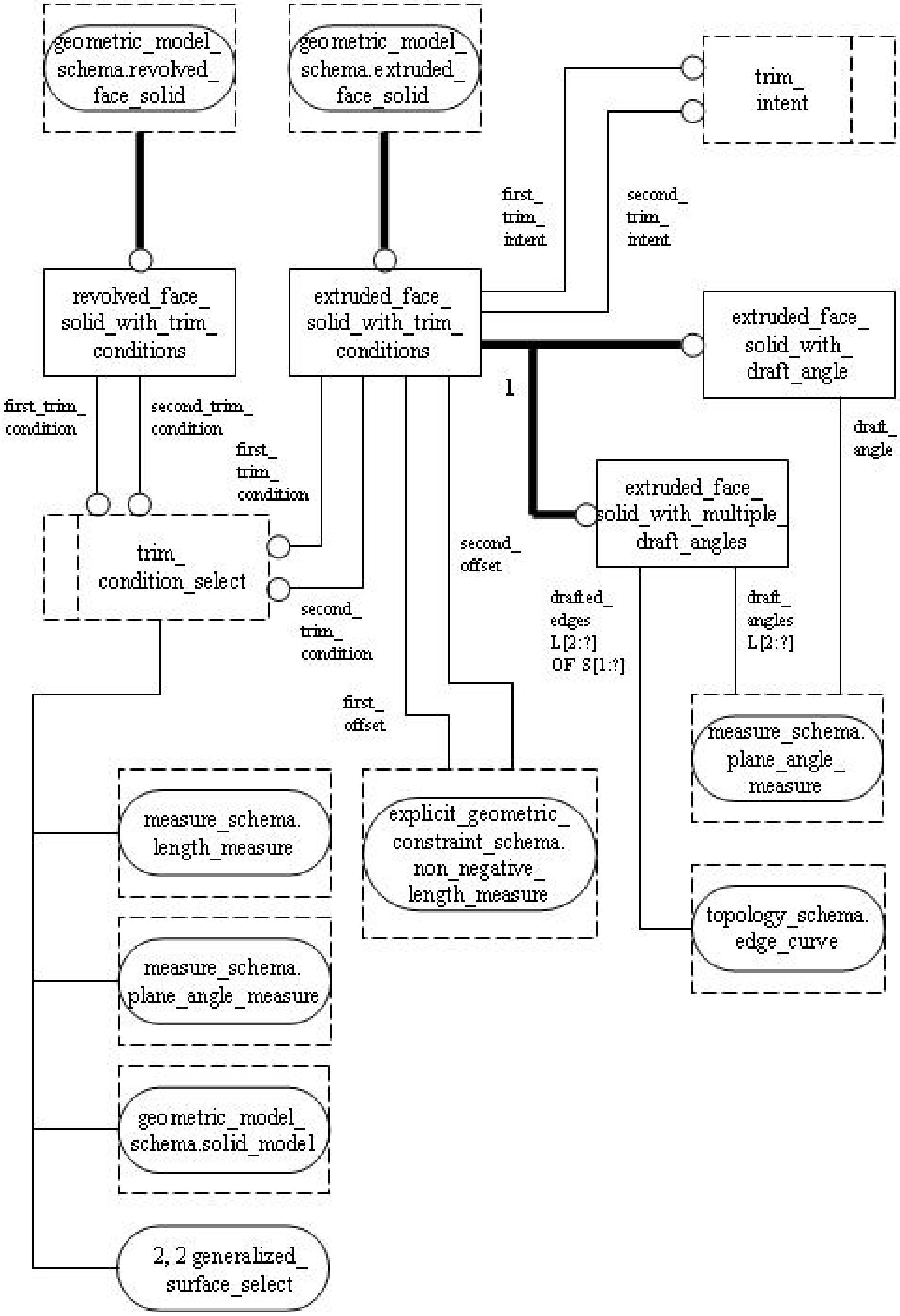


Figure D.10 – solid_shape_element_schema – EXPRESS-G diagram 10 of 11

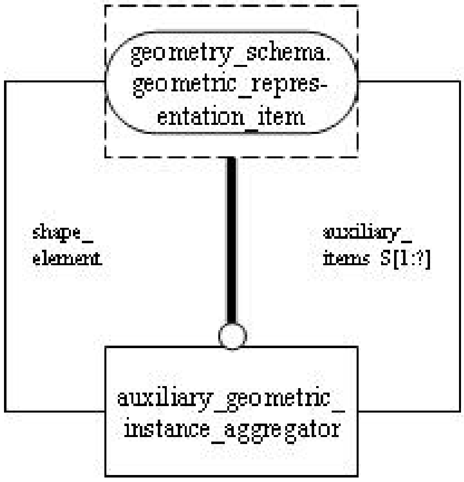


Figure D.11 – solid_shape_element_schema – EXPRESS-G diagram 11 of 11

Annex E
(informative)

Justification of the modelling approaches taken in this part of ISO 10303

E.1 Representation of constructional operations

This part of ISO 10303 provides shape elements that are primarily intended for use in the procedural representation of CAD models. A procedural representation is essentially a list of instructions for building the model, by contrast with an explicit or non-procedural representation, which specifies instances of all the elements composing the actual model as built. For a procedural model, representations are needed not of those explicit elements but of the constructional operations used to create them.

The method used in ISO 10303 for the representation of constructional operations in procedural models is defined in ISO 10303-55. The following is an extract from clause 4.2.5 of that document:

"An ISO 10303 model or **representation** is composed of instances of **representation_item** (see ISO 10303-43). The occurrence of such an instance in an exchange file or shared database was originally intended to indicate the actual presence of the item in the model being transferred. However, ISO 10303-11:2004 specifies (in clause 9.2.6) that 'When an entity is declared [in a schema], a constructor is also implicitly declared. The constructor identifier is the same as the entity identifier. . . The constructor, when invoked, shall return a partial complex entity value for that entity data type to the point of invocation. . . '. This capability was intended primarily for use in local or global rules in schemas, but it is used in this part of ISO 10303 for the representation of constructional operations for instances of the entity data types concerned *to be performed in the receiving system following a transfer*. Thus, whereas the 'point of invocation' of the constructor was originally envisaged to be during rule checking by an ISO 10303 translator, in the transfer of construction history models it will be during model regeneration in the receiving system. The treatment of attribute values as parameters passed to constructors is spelled out in detail in the cited clause of ISO 10303-11:2004."

EXAMPLE In the transfer of a non-procedural CAD model using, for instance, Edition 1 of the application protocol ISO 10303-203 [2], the presence of an instance of any geometric entity in an exchange file is an indication that such an instance participated explicitly in the model in the sending system. One way of transferring it into the receiving system is by translating the ISO 10303 representation into the corresponding representation in the native file format used by that system. If all elements in the transferred model are similarly written into that file, then the file can be read into the receiving system, which will the reconstruct the model.

A *procedural* model, on the other hand, is based on the use of the key entity type **procedural_shape_representation_sequence**, as defined in ISO 10303-55. This has an attribute **elements** that specifies a list of geometric or topological instances. If an instance of the geometric entity referred to in the last paragraph occurs in such a list, it will be interpreted as a constructor, and its effect in the receiving system will be a direct invocation of the creation operation for the entity type concerned. The distinction between the non-procedural and procedural interpretation of the same entity instance is therefore analogous to the distinction between copying the instance and creating a new instance of the same type.

The ISO 10303-55 mechanism allows a procedural representation to be generated in terms of instances of *any* subtypes of **representation_item** for which EXPRESS definitions exist. A wide selection of such entity data types is therefore immediately available for use with this part of ISO 10303, including the solids of extrusion and the boolean combinations defined in ISO 10303-42, which are routinely used in CAD design processes. However, most entities defined in ISO 10303-42 are low-level elements of boundary representation models, and defining procedural models in terms of these would be very tedious.

For that reason, this part of ISO 10303 defines a set of high-level constructional elements that capture many of the commonly used shape modelling capabilities of modern CAD systems.

The alternative approach to the capture and exchange of procedural models would have been to write new EXPRESS code, representing a constructional operation, corresponding to every existing entity definition whose instances may be constituents of an explicit CAD model. The possibilities discussed included the specification of 'operation' entities and of 'bodyless' functions or procedures whose actions were defined descriptively. The use of detailed code to define actions would effectively have required the writing of a CAD modelling system in EXPRESS, a purpose for which the language was never intended and for which it is not well suited. The dual use of entity definitions to model both constituent elements and the processes for creating them, as described earlier, avoids the need for the separate modelling of constructional operations. It is thus efficient in terms of development time and volume of documentation. Further, this approach was urged on the development team by members of the vendor community, because it enabled the re-use for the exchange of procedural models of much translator code that they had already developed for the transfer of explicit models.

Further information concerning procedural modelling in ISO 10303 may be found in ISO 10303-55.

E.2 Representation of local shape configurations

The types of local shape configurations defined in this part of ISO 10303 are referred to in the literature of CAD systems as *form features*. However, this terminology is misleading; strictly speaking, a form feature is a shape element that has engineering significance for some phase in the product life-cycle [7], but CAD systems generate only the shape with no associated engineering semantics. The functional significance of any created form element is probably present in the mind of the designer using a CAD system, but this information is not currently captured in the design process and no plausible means of doing so has yet emerged despite several years of research directed to that end. Thus the shape configurations defined in the **solid_shape_element_schema** should not be regarded as form features.

By contrast, other parts of ISO 10303, notably ISO 10303-109 [1], ISO 10303-214 [3] and ISO 10303-224 [4], define form features with associated semantics for specific application domains. The viewpoint they adopt is that the geometry of a form feature is a subregion of the boundary of a full part model, and this subregion is modelled using the entity **shape_aspect** as defined in ISO 10303-41. In the present document, however, the defined shape configurations are possibly only ephemeral; they are initially regarded as idealized shapes, which may lose their ideal form when installed on the previous version of the part model, and may then be further modified or even deleted from the model as a result of subsequent constructional operations. It is therefore not appropriate to model them as aspects of the shape of the final model. In fact this approach was tried on an experimental basis while the optimal modelling approach for this part of ISO 10303 was being decided upon, and was found to require the generation of many supporting instances in the exchanged model that had no real significance for the exchange process and were certainly no help to the receiving system in reconstructing the transferred model.

For these reasons it was decided to regard the shape elements defined in this document in a similar light to those defined in ISO 10303-42, in particular the solids of extrusion and the boolean combinations specified there. Indeed, some of the entities defined in the **solid_shape_element_schema** are directly subtyped from ISO 10303-42 entities. The decision to model in terms of new configurations created upon a pre-existing base solid was suggested by vendor participants in the development team, who in the light of experience had found this an effective way of capturing CAD constructional processes.

It is also worth mentioning that the shape elements defined in this document may be explicitly parameterized by using the capabilities of ISO 10303-108.

Bibliography

[1] ISO 10303-109, *Industrial automation systems and integration — Product data representation and exchange — Part 109: Integrated application resource: Kinematic and geometric constraints for assembly models*

[2] ISO 10303-203:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 203: Application protocol: Configuration controlled 3D designs of mechanical parts and assemblies*

[3] ISO 10303-214, *Industrial automation systems and integration — Product data representation and exchange — Part 214: Application protocol: Core data for automotive mechanical design processes*

[4] ISO 10303-224, *Industrial automation systems and integration — Product data representation and exchange — Part 224: Application protocol: Mechanical product definition for process planning using machining features*

[5] DE BOOR, C. *A Practical Guide to Splines*, New York: Springer-Verlag, 1978

[6] DO CARMO, M. P. *Differential Geometry of Curves and Surfaces*, Englewood Cliffs, New Jersey: Prentice-Hall, 1976

[7] SHAH, J. J.; MÄNTYLÄ, M. *Parametric and Feature-based CAD/CAM*, New York: Wiley, 1995

Index

ICS 25.040.40

Price based on 95 pages