INTERNATIONAL STANDARD ISO 10303-108:2005
TECHNICAL CORRIGENDUM 1

Published 2008-12-15

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION

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

Part 108: Integrated application resource: Parameterization and constraints for explicit geometric product models

TECHNICAL CORRIGENDUM 1

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

Partie 108: Ressources d'application intégrées: Paramétrage et contraintes pour les modèles de produits géométriques explicites

RECTIFICATIF TECHNIQUE 1

Technical Corrigendum 1 to ISO 10303-108:2005 was prepared by Technical Committee ISO/TC 184, *Automation systems and integration*, Subcommittee SC 4, *Industrial data.*

Introduction

The modifications made to ISO 10303-108:2005 have three purposes:

a) to remove an entity name clash with ISO 10303-210:2001 (published earlier than ISO 10303-108:2005 and therefore having prior claim on the name) concerning **model_parameter**. *This Technical Corrigendum provides for its replacement throughout ISO 10303-108 with* **variational_parameter***;*

b) to remove the definition of **non_negative_length_measure***, which has been moved to ISO 10303-41, and to replace it by a reference to that resource;*

c) to correct minor errors in EXPRESS code.

The opportunity has also been taken to update the normative reference to ISO 10303-55 (now published) and to correct a few minor editorial errors mainly concerning the numbering of notes and examples.

ICS 25.040.40 **Ref. No. ISO 10303-108:2005/Cor.1:2008(E)**

Published in Switzerland

Modifications to the text of ISO 10303-108:2005

Table of Contents, p. iii ff.

The term **model_parameter** *is being systematically replaced by* **variational_parameter**. *The necessary entity name change requires replacement of the titles of several subclauses.*
Make the following replacements:

Subclause:	*Previous title:*	*New title:*
4.2.1	Model parameters	Variational parameters
4.4.1	model_parameter	variational_parameter
4.4.2	bound_model_parameter	bound_variational_parameter
4.4.3	unbound_model_parameter	unbound_variational_parameter
4.4.7	unbound_model_parameter_semantics	unbound_variational_parameter_semantics
5.2.5	Roles of model parameters...	Roles of variational parameters...

Delete the entry 7.3.10 non_negative_length_measure *from the Table of Contents*

Clause 1, p. 1

Replace model parameters *by* variational parameters *in line 1 of the first paragraph, and lines 1 and 3 of the second paragraph.*

Clause 1.1, p. 3

Replace model parameters *by* variational parameters *in the third bulleted item of this subclause.*

Clause 2, p. 5

ISO 10303-55 has now been published. After ISO 10303-55 *delete :—, and also delete the footnote.*

Clause 3.7.24, p. 11

Replace **model parameter** *by* **variational parameter**, *and reposition this definition in the list to follow the definition of* **variational** *(previously clause 3.7.34). The wording of the definition is unchanged, but the text of the notes and the example needs to be changed.*
Replace NOTE 1, EXAMPLE and NOTE 2 as follows:

NOTE 1 Assignment of different values to variational parameters generates different members of a family of models. Variational parameters therefore express design freedom in a model, according to the parameterization scheme imposed by its creator. Limitations may be defined on the allowable ranges of variational parameters.

EXAMPLE The dimensions of a generic block may be represented by variational parameters L (length), W (width) and H (height). Individual members of the family of blocks are specified by assigning numerical values to the three parameters independently. Alternatively, relationships may be defined between the variational parameters, such as $L = 2W, H = 0.5W$, to restrict the size of the family and define it in terms of the single independent variational parameter W.

NOTE 2 Distinction must be made between the use of the word *parameter* in this part of ISO 10303, in ISO 10303-11, in ISO 10303-42 and in ISO 10303-50. In ISO 10303-11 a parameter is used for the formal representation of an input to, or output from, a function or procedure defined in an EXPRESS schema. In ISO 10303-42 a parameter is a variable used to identify the position of a point on a curve or a surface, so that the parameter may be thought of as an input to a function whose output is a coordinate value. In ISO 10303-50 a parameter is defined as 'a free variable in an expression'. In this part of ISO 10303 the term **variational parameter** is used for a variable that controls dimensions or other gross characteristics of a model, for example the overall shape of a product model. A variational parameter may be thought of as an input to a procedure, in this case a procedure that

computes one instance of a family of shape models. It is unfortunate that the word 'parameter' is in widespread current use for such a variety of purposes. Although at a broad conceptual level the usages within ISO 10303 are similar, there are significant differences in such matters as the way the functions or procedures are defined and in the scope of parameters in a model.

Clause 3.7 more generally
The terms defined in this clause need to be reordered into alphabetical order, and other references to the superseded entity name need to be changed as follows: (a) Renumber clause 3.7.24, as modified above, to 3.7.34; (b) Renumber the current clauses 3.7.25 – 3.7.34 as 3.7.24 – 3.7.33, keeping their sequence the same; (c) In the clause newly numbered 3.7.25, defining the term **parameter**, *replace the existing definition by* variational parameter (in the context of this part of ISO 10303) — see the definition of **variational parameter** given in clause 3.7.34.

Clause 4, p. 15
Clause 4.2 and most of clause 4.4 need to be replaced. Subclauses 4.1 and 4.3 may remain as they are (subject to the correction noted below), and so may subclause 4.5. Clause 4.4.9 needs only one name replacement.

Clause 4.1, p. 15
In NOTE 2, *replace* Figure *by* Figures.

Clause 4.2, p. 15
Replace subclause 4.2 by the following:

4.2 Fundamental concepts and assumptions

This schema provides representation methods for the following:

— Variables, represented by instances of **bound_variational_parameter** or **unbound_variational_parameter**, expressing variation or design freedom in a representation or model;

— A means for binding a **bound_variational_parameter** instance to an attribute of another entity data type instance in the same **representation**;

— Domains of validity for instances of **bound_variational_parameter** and **unbound_variational_parameter**;

— A means for fixing the values of attributes of specific entity data type instances in a model, equivalent to the use of **bound_variational_parameter** instances with constant associated values.

These resource constructs are of general utility in the exchange and sharing of ISO 10303 models embodying

— the capability for variation of attribute values in a model following an exchange;

— the capture and transfer of constraint relationships defined in terms of mathematical expressions, functions or procedures. Specifically, variational parameters can participate in instances of **free_form_constraint** as defined in clause 5.4.4.

Clause 6.3.1 of this part of ISO 10303 defines **variational_representation_item** as a subtype of the ISO 10303-43 entity data type **representation_item**. Variational parameters are defined as subtypes of

variational_representation_item, which is the supertype of all entity data types used to express the variational aspects of models with explicit parameterization and constraints. The type of **representation** in which they participate is a **variational_representation**, as defined in clause 6.3.3.

4.2.1 Variational parameters

An abstract entity data type **variational_parameter** is provided, with two instantiable subtypes, **bound_variational_parameter** and **unbound_variational_parameter**. These allow for the capture and transmission of permitted aspects of model variation that can be exploited in a receiving system. A **bound_variational_parameter** is bound to an attribute of an entity data type instance in an ISO 10303 model, in which case it provides a syntactic representation of the value of that attribute, for example a dimensional value. By contrast, an **unbound_variational_parameter** is not directly associated with any model attribute. Either kind of **variational_parameter** may be used in mathematical relationships defined in free-form constraints. The current value of a **variational_parameter** is specified by one of its attributes; in the bound case the value of this attribute is required by an informal proposition to be the same as the value of the attribute to which it is bound.

The entity data type **variational_parameter** is defined as a subtype of **variational_representation_item**, and the scope of its instantiable subtypes is therefore defined by those instances of **variational_representation** in which they participate. It is also a subtype of the ISO 10303-50 entity data type **maths_variable**, from which it inherits an attribute **values_space**, of ISO 10303-50 type **maths_space**. This attribute specifies the domain of validity for values of the **variational_parameter**. These may include domains corresponding to those of the EXPRESS data types REAL, INTEGER, BOOLEAN and STRING, together with various bounded subsets of the REAL and INTEGER domains. This part of ISO 10303 does not directly provide the use of parameters having values belonging to aggregate types, but applications may define such extensions if they are required.

EXAMPLE 1 Consider a rectangle, with length x units and width y units. Here x and y are variables or parameters. An explicit constraint relationship $x = y^2 + 2$ relates these dimensions. Valid parameter ranges $10.0 \leq x \leq 30.0$ and $2.0 \leq y \leq 5.0$ are defined. In this case the two variables correspond to instances of **bound_variational_parameter**, both bound to physical quantities in the model, i.e., dimensional attributes of the rectangle. The parameterization and constraint information may be transmitted together with a 'current result' — an explicit model of a rectangle with length 18.0 units and width 4.0 units. These parameter values satisfy the constraint and fall within the required parameter ranges. When model transfer is complete, if one of the parameters is edited the other should adjust accordingly to maintain satisfaction of the constraint, provided the parameters remain within their valid ranges. It is assumed that the necessary functionality for parameter variation and constraint maintenance will be provided by the receiving system.

The following example illustrates the use of an **unbound_variational_parameter**.

EXAMPLE 2 Suppose an instance of **right_circular_cylinder** (as defined in ISO 10303-42), has associated instances of **bound_variational_parameter** associated with its **radius** and **height** attributes, here denoted by r and h respectively. A third parameter, denoted by t, may be used to control the values of both r and h according to the relationships $r = 3t - 2$, $h = t^2 + 1$. In the case when t is not bound to an attribute of any entity data type instance, it will appropriately be modelled in terms of an **unbound_variational_parameter**.

4.2.2 Parameter binding to an instance attribute

A **bound_variational_parameter** is associated with an attribute of an entity data type instance in a populated schema, whose value represents the value of the parameter. This association is defined through the use of an entity data type **instance_attribute_reference** that simply specifies the name of an attribute

and the instance to which it belongs (see clause 4.4.6). A simple example is given below to illustrate the principle, and the intended usage of the mechanism is more fully documented in clause F.1 of annex F. Once the parameter binding has been established, the parameter may participate in a relationship that governs its value if the model is subsequently edited in a receiving system.

EXAMPLE For the purpose of the example, entity data types defined in the ISO 10303 integrated generic resources are treated as though they are instantiable elements in an application protocol.

It is desired to parameterize one dimension of a **block** solid, as defined in ISO 10303-42. This has three attributes, `x`, `y` and `z`, that prescribe its three principal dimensions. In any instantiation of the block these will have specific real numerical values. Consider now the following fragment of an ISO 10303-21 transfer file:

```
#290 = AXIS2_PLACEMENT_3D(...);
#300 = BLOCK('BLOCK1', #290, 4.0, 6.0, 8.0);
#310 = INSTANCE_ATTRIBUTE_REFERENCE
         ('GEOMETRIC_MODEL_SCHEMA.BLOCK.X', #300);
#320 = FINITE_REAL_INTERVAL(2.0, .CLOSED., 10.0, .CLOSED.);
#330 = BOUND_VARIATIONAL_PARAMETER
         ('XPARAM', #320, 'XPARAM', 'BLOCK X-DIMENSION', *);
#340 = BOUND_PARAMETER_ENVIRONMENT(#310, #330);
```

The instances represented above are explained as follows:

`#290`: defines an ISO 10303-42 axis placement (details omitted) for the next instance;

`#300`: the **block** instance. As a subtype of ISO 10303-43 **representation_item**, this inherits a **name** attribute of type **label**, whose value in this instance is `'block1'`. The block is defined with respect to the axis placement `#290` and has dimensions 4.0, 6.0 and 8.0 units;

`#310`: an instance of **instance_attribute_reference**; `'geometric_model_schema.block.x'` is the specified attribute name and the referenced **block** instance is `#300`. Note that the attribute name appears fully qualified with the name of the owning entity data type and its defining schema. This entry in the file identifies the particular instance whose specified attribute is to be associated with the **bound_variational_parameter** instance;

`#320`: defines the domain of that parameter, a real interval closed at both ends, bounded below by 2.0 and above by 10.0. The entity data type **finite_real_interval** is defined in ISO 10303-50;

`#330`: specifies the **bound_variational_parameter** itself, as defined in clause 4.4.2 of this schema. Its attribute value list contains these entries:

— a label, `'xparam'`, corresponding to the **name** attribute of its **representation_item** supertype;

— a domain `#320`, corresponding to the **values_space** attribute of its **maths_variable** supertype;

— a label, `'xparam'`, corresponding to the **name** attribute of its **maths_variable** supertype — the two inherited **name** attributes are required by a WHERE rule to have the same values;

— a textual description `'block x-dimension'`;

— the value of the **variational_parameter**, given as a derived value, although no formal method is available in EXPRESS for deriving it from instance `#300`;

`#340`: an instance of **bound_parameter_environment**, defined in clause 4.4.4, providing the link between the specified instance attribute, `#310`, and the parameter bound to it, `#330`.

At this point the binding of the parameter to the desired attribute is complete. The intention is that the value attribute of the **bound_variational_parameter** instance `#330` is equal to the value 4.0 associated with the `'x'` attribute in the **block** instance, and lies within the domain of validity represented by `#320`. However, because the

EXPRESS language provides no formal way of asserting this the value attribute of the parameter is recorded as indeterminate, and an informal proposition in clause 4.4.2 requires that the two values shall be equal on completion of the transfer. The achievement of this is the responsibility of the translation software. The parametric relationship having been captured in an exchange file as shown above, the x-dimension of the block may now be controlled in terms of the parameter associated with it if the model is edited following transfer into a receiving system.

Clause 4.4, p. 19
Replace subclause 4.4 up to and including 4.4.8 by the following:

4.4 Parameterization entity definitions

4.4.1 variational_parameter

The **variational_parameter** entity data type is a type of **variational_representation_item** that represents a variable quantity in a **variational_representation** (see clause 6). It is also a type of **maths_variable** as defined in ISO 10303-50, and can therefore participate in mathematical relationships. Its attributes include an optional textual description of the significance of the parameter, and a current parameter value. A **variational_parameter** instance inherits a **name** attribute from both its supertypes; for consistency, the **maths_variable** name is required to be the same as the **representation_item** name. It also inherits an attribute **values_space** from its **maths_variable** supertype, specifying the domain (permissible set of values) of the **variational_parameter**.

NOTE 1 The fact that **variational_parameter** is a type of **maths_variable** restricts its underlying domain of values to subsets of the real or integer numbers, Booleans or strings. Future editions of this part of ISO 10303 have the possibility of extending that spectrum of domains, if applications require it, by making use of the more general capabilities of ISO 10303-50.

Because **variational_parameter** is a type of **maths_variable**, and hence ultimately of **generic_variable** as defined in ISO 13584-20, each instance of it is required to have an associated instance of the ISO 13584-20 entity data type **environment**, which links the parameter with its associated semantics. For that purpose, this schema provides appropriate subtypes of **environment**, namely **bound_parameter_environment** and **unbound_parameter_environment**, as defined in clauses 4.4.4 and 4.4.5 respectively. The key relationships between these ISO 10303-108 and ISO 13584-20 entity data types are shown in clause F.1 of annex F.

EXPRESS specification:

```
*)
ENTITY variational_parameter
  ABSTRACT SUPERTYPE OF (ONEOF (bound_variational_parameter,
                                unbound_variational_parameter))
  SUBTYPE OF (variational_representation_item, maths_variable);
  parameter_description : OPTIONAL text;
  parameter_current_value : maths_value;
WHERE
  WR1 : member_of(parameter_current_value,
    SELF\maths_variable.values_space);
  WR2 : SELF\maths_variable.name = SELF\representation_item.name;
END_ENTITY;
(*
```


Attribute definitions:

parameter_description: An optional description, for human interpretation, of the significance of the **variational_parameter** instance.

parameter_current_value: The current value associated with the **variational_parameter** instance.

SELF\maths_variable.values_space: The domain of validity of the current value associated with the **variational_parameter** instance.

SELF\maths_variable.name: The **name** attribute of the **maths_variable** supertype.

SELF\representation_item.name: The **name** attribute of the **representation_item** supertype, whose value is required to be the same as for the previous attribute.

Formal propositions:

WR1: The current value of the **variational_parameter** instance shall lie within the domain specified by the attribute **SELF\maths_variable.values_space**.

WR2: The **name** attributes of supertypes **maths_variable** and **representation_item** shall be the same.

NOTE 2 No requirement has been imposed for the **name** attribute of a **variational_parameter** instance to have a value that is unique in any **representation** it participates in. This is because it is not anticipated that **variational_parameter** instances will be referred to by their name attributes. In general, **variational_parameter** instances used in an exchange are referenced by ephemeral identifiers created during the translation process and discarded when the exchange is complete, by which time system-dependent identifiers have been generated in the receiving system. However, if uniqueness of name attributes is required for some application purpose the necessary restriction can be imposed in schemas that specialize definitions from this part of ISO 10303.

4.4.2 bound_variational_parameter

The **bound_variational_parameter** entity data type is a type of **variational_parameter** whose instances can be bound to (associated with) explicit attributes of entity instances participating in a **variational_representation**. The current value of any instance of **bound_variational_parameter** is indeterminate during an exchange, but is required by an informal proposition to be set equal in the receiving system to the value of the attribute to which it is bound. That attribute is therefore required to have an explicit value in a populated schema, which rules out the association of a **bound_variational_parameter** with a derived or inverse attribute.

NOTE 1 This approach to the association of a value with a **bound_variational_parameter** instance is necessary because the EXPRESS language provides no means for the formal derivation of the value from the referenced entity data type instance in a populated schema.

EXPRESS specification:

```
*)
ENTITY bound_variational_parameter
  SUBTYPE OF (variational_parameter);
DERIVE
  SELF\variational_parameter.parameter_current_value : maths_value := ?;
WHERE
```



```
WR1 : 'PARAMETERIZATION_SCHEMA.BOUND_PARAMETER_ENVIRONMENT'
    IN TYPEOF(SELF\generic_variable.interpretation);
END_ENTITY;
(*
```

Attribute definitions:

parameter_current_value: The current value of the attribute to which the parameter is bound, always derived as indeterminate for an instance of this entity data type (see notes 1, 2 and 3).

SELF\generic_variable.interpretation: The instance of **bound_parameter_environment** that links a **bound_variational_parameter** instance to a particular entity instance attribute.

Formal propositions:

WR1: Every instance of **bound_variational_parameter** shall reference an instance of type **bound_-parameter_environment**.

NOTE 2 The indeterminate value of the the attribute **parameter_current_value** does not give rise to a violation of WR1 of the **variational_parameter** supertype, which for an instance of **bound_variational_parameter** will evaluate to UNKNOWN rather than FALSE. Clause 9.2.2.2 of ISO 10303-11 states that this does not constitute a violation of the rule. In practice it is the responsibility of the translator software to check that the value of the referenced attribute lies within the domain of the parameter.

NOTE 3 Because the indeterminate value of **parameter_current_value** is derived — in this case, by a simple assigment — an instance of **bound_variational_parameter** in an ISO 10303-21 exchange file will represent it by an asterisk, *, as illustrated in the examples in clause 4.2.2 and annex F.

NOTE 4 The ISO 13584-20 entity data type **environment** has two attributes, **semantics** and **syntactic_representation**. As a subtype of **environment**, **bound_parameter_environment** also possesses these attributes, which are treated as follows:

> **semantics:** This attribute is of type **instance_attribute_reference** (see clause 4.4.6).
>
> **syntactic_representation:** A WHERE rule applying to **bound_parameter_environment** requires that the value of this attribute shall be of type **bound_variational_parameter**.

The ISO 13584-20 entity data type **generic_variable** has an inverse attribute **interpretation**, corresponding to the **syntactic_representation** attribute of **environment**. For **bound_variational_parameter**, a subtype of **generic_-variable**, this inverse attribute is required by WR1 to be of type **bound_parameter_environment**.

The entity data types **environment** and **variable_semantics** are subtyped in this part of ISO 10303 to satisfy a requirement of ISO 13584-20 regarding the binding of values to variables. An EXPRESS-G representation of their relationships with entity data types defined in this schema is given in clause F.1 of annex F.

Informal propositions:

IP1: The **parameter_current_value** attribute shall have the same value as the entity data type instance attribute referenced via the inverse attribute **SELF\generic_variable.interpretation**, and shall be type-compatible with it.

NOTE 5 It will be crucial for implementations to ensure that the foregoing informal proposition is satisfied. Means cannot be provided in this schema for checking its validity because no formal mechanism exists for accessing the value of the attribute with which the **bound_variational_parameter** is associated.

NOTE 6 A local rule in the definition of **variational_representation** (see clause 6.3.3) ensures that any instance of **bound_variational_parameter** shall belong to the same **variational_representation** as the entity data type instance to whose specified attribute it is bound.

NOTE 7 No restriction is imposed in this schema to prevent the binding of more than one **bound_variational_parameter** to a single attribute of an entity data type instance. However, if such a restriction is required for some application purpose it can be specified in schemas that specialize definitions from this part of ISO 10303.

NOTE 8 The mechanism defined in this schema does not allow the direct association of a **bound_variational_parameter** instance with more than one entity data type instance attribute. The effect of such a multiple binding can be achieved through the use of multiple **bound_variational_parameter** instances related by the **equal_parameter_constraint** as defined in clause 5.4.3.

4.4.3 unbound_variational_parameter

The **unbound_variational_parameter** entity data type is a type of **variational_parameter** representing a variable that is not bound to an attribute of any entity instance in the model. The value attribute of an **unbound_variational_parameter** instance is specified explicitly, rather than by association with an attribute of some other instance in the model.

NOTE 1 An instance of **unbound_variational_parameter** may be used in mathematical expressions in free-form constraints that govern values associated with instances of **bound_variational_parameter**. Examples of this usage are given in clause 4.2.1 and clause F.1 of annex F.

EXPRESS specification:

```
*)
ENTITY unbound_variational_parameter
  SUBTYPE OF (variational_parameter);
WHERE
  WR1: 'PARAMETERIZATION_SCHEMA.UNBOUND_PARAMETER_ENVIRONMENT'
    IN TYPEOF(SELF\generic_variable.interpretation);
END_ENTITY;
(*
```

Attribute definitions:

SELF\generic_variable.interpretation: The instance of **unbound_parameter_environment** providing the link between the **unbound_variational_parameter** instance and its associated instance of **unbound_variational_parameter_semantics**. The definitions of these entities are given in clauses 4.4.5 and 4.4.7.

Formal propositions:

WR1: Every instance of **unbound_variational_parameter** shall be referenced by an instance of **unbound_parameter_environment**.

NOTE 2 The ISO 13584-20 entity data type **environment** has two attributes, **semantics** and **syntactic_representation**. As a subtype of **environment**, **unbound_parameter_environment** also possesses these attributes, which are treated as follows:

semantics: The value of this attribute is required to be of type **unbound_variational_parameter_semantics** (see clause 4.4.7).

syntactic_representation: A WHERE rule applying to **unbound_parameter_environment** requires that the value of this attribute shall be of type **unbound_variational_parameter**.

The ISO 13584-20 entity data type **generic_variable** has an inverse attribute **interpretation**, corresponding to the **syntactic_representation** attribute of **environment**. For **unbound_variational_parameter**, a subtype of **generic_variable**, this inverse attribute is required by WR1 to be of type **unbound_parameter_environment**.

The entity data types **environment** and **variable_semantics** are subtyped in this part of ISO 10303 to satisfy a requirement of ISO 13584-20 regarding the binding of values to variables. An EXPRESS-G representation of their relationships with entity data types defined in this schema is given in clause F.1 of annex F.

4.4.4 bound_parameter_environment

The **bound_parameter_environment** entity data type is a type of **environment** as defined in ISO 13584-20. It provides a link between the syntactic and semantic aspects of a **bound_variational_parameter** instance.

NOTE 1 ISO 13584-20 requires an instance of **environment** to be defined for every instance of **generic_variable**, and since **bound_variational_parameter** as defined in this schema is a subtype of **generic_variable** it has been necessary to provide an appropriate subtype of **environment** in this schema.

EXPRESS specification:

```
*)
ENTITY bound_parameter_environment
  SUBTYPE OF (environment);
WHERE
  WR1: ('PARAMETERIZATION_SCHEMA.BOUND_VARIATIONAL_PARAMETER' IN
    TYPEOF(SELF\environment.syntactic_representation)) AND
    ('PARAMETERIZATION_SCHEMA.INSTANCE_ATTRIBUTE_REFERENCE' IN
    TYPEOF(SELF\environment.semantics));
END_ENTITY;
(*
```

Formal propositions:

WR1: For every instance of **bound_parameter_environment**, the **syntactic_representation** attribute of the **environment** supertype shall be of type **bound_variational_parameter** and the **semantics** attribute shall be of type **instance_attribute_reference**.

NOTE 2 The relationships between the ISO 13584-20 entity data type **environment** and other entity data types defined in this schema are illustrated by the EXPRESS-G diagram in clause F.1 of annex F.

4.4.5 unbound_parameter_environment

The **unbound_parameter_environment** entity data type is a type of **environment** as defined in ISO 13584-20. It provides a link between the syntactic and semantic aspects of an **unbound_variational_-parameter** instance.

NOTE 1 ISO 13584-20 requires an instance of **environment** to be defined for every instance of **generic_-variable**, and since **unbound_variational_parameter** as defined in this schema is a subtype of **generic_variable** it has been necessary to provide an appropriate subtype of **environment** in this schema.

EXPRESS specification:

```
*)
ENTITY unbound_parameter_environment
  SUBTYPE OF (environment);
WHERE
  WR1: ('PARAMETERIZATION_SCHEMA.UNBOUND_VARIATIONAL_PARAMETER' IN
    TYPEOF(SELF\environment.syntactic_representation)) AND
    ('PARAMETERIZATION_SCHEMA.UNBOUND_VARIATIONAL_PARAMETER_SEMANTICS' IN
    TYPEOF(SELF\environment.semantics));
END_ENTITY;
(*
```

Formal propositions:

WR1: For any instance of **unbound_parameter_environment**, the **syntactic_representation** attribute of the **environment** supertype shall be of type **unbound_variational_parameter** and the **semantics** attribute shall be of type **unbound_variational_parameter_semantics**.

NOTE 2 The relationships between the ISO 13584-20 entity data type **environment** and other entity data types defined in this schema are illustrated by the EXPRESS-G diagram in clause F.1 of annex F.

4.4.6 instance_attribute_reference

The **instance_attribute_reference** entity data type is a type of **variable_semantics** (see ISO 13584-20). It identifies a named explicit attribute of a specific **representation_item** instance in a populated EXPRESS schema. The name of the attribute is specified as an **attribute_identifier** (see clause 4.3.1). Derived or inverse attributes shall not be referenced by the use of **instance_attribute_reference**.

NOTE 1 This entity data type is used in the definition of an association between a bound **variational_parameter** and an attribute of an instance of a **representation_item**. It is defined as a subtype of the ISO 13584-20 entity data type **variable_semantics** to satisfy a requirement of that standard regarding the binding of values to variables. The intention is to provide an interpretation of the role of the variable in its modelling context.

Examples of the use of this entity data type are provided in clause F.1 of annex F.

EXPRESS specification:

```
*)
ENTITY instance_attribute_reference
  SUBTYPE OF (variable_semantics);
  attribute_name  : attribute_identifier;
  owning_instance : representation_item;
END_ENTITY;
(*
```

Attribute definitions:

attribute_name: An **attribute_identifier** specifying the name of the referenced attribute.

owning_instance: The **represention_item** instance owning the referenced attribute.

Informal propositions:

IP1: Any attribute referenced by an instance of **instance_attribute_reference** shall be specified as a fully qualified attribute in the form `'SCHEMA_NAME.ENTITY_NAME.ATTRIBUTE_NAME'` (see ISO 10303-11).

NOTE 2 The foregoing informal proposition is imposed to ensure that ambiguities are avoided, for example in the case of complex instances, and to enable effective checking for compatibility of the referenced attribute identifier and its owning entity data type instance.

4.4.7 unbound_variational_parameter_semantics

The **unbound_variational_parameter_semantics** entity data type is a type of **variable_semantics** as defined in ISO 13584-20. It represents the semantics of an unbound **variational_parameter**.

NOTE ISO 13584-20 requires a subtype of **variable_semantics** to be defined for any subtype of **generic_variable**. This is intended to provide an interpretation of the role of the variable in its modelling context. The definition of **unbound_variational_parameter_semantics** given below reflects the fact that an unbound **variational_parameter** has no semantics beyond what is implied by its use in an instance of **free_form_constraint** (see clause 5.4.4). It does not necessarily have any immediate physical significance in its own right, though it may indirectly control quantities that do have physical significance.

EXPRESS specification:

```
*)
ENTITY unbound_variational_parameter_semantics
  SUBTYPE OF (variable_semantics);
END_ENTITY;
(*
```


4.4.8 fixed_instance_attribute_set

The **fixed_instance_attribute_set** entity data type is a type of **variational_representation_item** as defined in clause 6.3.1. It specifies a set of explicit attribute values in a populated schema whose values are required to remain fixed in the model after transfer to a receiving system. In particular, these fixed attributes may represent values of dimensions in a shape model. This entity data type shall not be used to constrain values of derived or inverse attributes.

NOTE Although this entity data type may appear to have the nature of a constraint, it has been included here rather than in the **explicit_constraint_schema** (clause 5) for the following reasons:

a) It makes use of **instance_attribute_reference** instances in the same manner as the **bound_variational_parameter** entity data type defined in this schema;

b) The **explicit_constraint** entity data type defined in clause 5 constrains instances of **representation_item** rather than individual attributes of such instances;

c) A fixed attribute may be considered to be parameterized with a domain restricted to a single value.

EXPRESS specification:

```
*)
ENTITY fixed_instance_attribute_set
  SUBTYPE OF (variational_representation_item);
  fixed_attributes : SET[1:?] OF instance_attribute_reference;
WHERE
  WR1: SIZEOF(QUERY(q <* using_representations(SELF) |
    SIZEOF(QUERY(r <* q.items |
    'PARAMETERIZATION_SCHEMA.FIXED_INSTANCE_ATTRIBUTE_SET'
    IN TYPEOF(r))) > 1)) = 0;
END_ENTITY;
(*
```

Attribute definitions:

fixed_attributes: The set of entity data type instance attributes that have fixed values in the model.

Formal propositions:

WR1: The current instance shall be the only instance of **fixed_instance_attribute_set** in any **representation** that contains it.

Clause 4.4.9, p. 26

In NOTE 1, *replace* **model_parameter** *by* **variational_parameter**.

Clause 5.1, p.30

Replace the reference to the **parameterization_schema** *by the following:*


```
REFERENCE FROM parameterization_schema                -- ISO 10303-108
  (bound_variational_parameter,
   variational_parameter,
   unbound_variational_parameter);
```

Clause 5.2, p.31

In paragraph 5, line 5, replace **model_parameter** *by* **variational_parameter**.

Clause 5.2.5, p.33

In the subclause heading, replace **model parameters** *by* **variational parameters**. *In line 2 and line 4 of this clause, and also in the* NOTE, *replace* **model_parameter** *by* **variational_parameter**.

Clause 5.4, p. 34

This clause needs to be replaced from subclause 5.4.3 to 5.4.7, inclusive. Replace the existing text with the following:

5.4.3 equal_parameter_constraint

The **equal_parameter_constraint** entity data type is a type of **defined_constraint** that constrains a set of **variational_parameter** instances to have equal value attributes. There are directed and undirected forms:

directed: All the constrained elements are required to have the same value as a single reference element;

undirected: There is no reference element, and all the constrained elements are required to share a common value.

EXAMPLE 1 It may be desired to constrain all constant radius blends in a geometric model to have the same radius. In this case, a **variational_parameter** may be associated with each blend radius and the undirected form of **equal_parameter_constraint** used to assert the equality of value of all these parameters.

NOTE The **free_form_relation** constraint, defined in clause 5.4.6 allows a the value of a parameter to be constrained through its participation in a mathematical relationship. The directed form of the **equal_parameter_constraint** can achieve the effect of constraining a *set* of parameters according to a relationship in which only one of them participates, by using as its reference element the parameter controlled by the **free_form_relation**.

EXAMPLE 2 Suppose that **variational_parameter** instances corresponding to variables x, y, z are constrained by a **free_form_relation** constraint to satisfy the relationship $x^2 + y^2 = z^2$. If it is additionally desired that parameters corresponding to variables q, r, s always have values equal to that of z, then a directed **equal_parameter_constraint** may be used, in which the set of **constrained_elements** contains the **variational_parameter** instances corresponding to q, r, s, while z is used as a **reference_element**.

The **equal_parameter_constraint** is an elementary example of the **defined_constraint** class. It asserts that the values of a set of parameters are equal, but the relationship is expressed descriptively rather than in explicit mathematical terms as would be the case in a **free_form_constraint**.

EXPRESS specification:

```
*)
ENTITY equal_parameter_constraint
  SUBTYPE OF (defined_constraint);
  SELF\explicit_constraint.constrained_elements :
    SET[1:?] OF variational_parameter;
  SELF\explicit_constraint.reference_elements   :
    SET[0:1] OF variational_parameter;
WHERE
  WR1: SIZEOF(SELF\explicit_constraint.constrained_elements +
    SELF\explicit_constraint.reference_elements) >= 2;
END_ENTITY;
(*
```

Attribute definitions:

SELF\explicit_constraint.constrained_elements: A set of **variational_parameter** instances whose values are constrained to be equal.

SELF\explicit_constraint.reference_elements: A set of zero or one **variational_parameter** instances; if a reference element is present its value is assigned to all the constrained elements.

Formal propositions:

WR1: The total number of **variational_parameter** instances involved in an **equal_parameter_constraint** shall not be less than two.

5.4.4 free_form_constraint

The **free_form_constraint** entity data type is a type of **explicit_constraint** defining a special purpose constraint that cannot be modelled with the defined constraint entities available in any particular application context.

As with other explicit constraints, the constrained elements are those controlled by the constraint, and the reference elements specify, for the case of directed constraints, elements upon which they are constrained to be dependent. The dependency generally involves relations between values of attributes of entity data type instances, and the types of the constrained and reference elements are therefore both specified as **variational_parameter**. Two specialized subtypes of this constraint are defined below.

EXPRESS specification:

```
*)
ENTITY free_form_constraint
  ABSTRACT SUPERTYPE OF (ONEOF(free_form_assignment, free_form_relation))
  SUBTYPE OF (explicit_constraint);
  SELF\explicit_constraint.constrained_elements :
    SET[1:?] OF variational_parameter;
  SELF\explicit_constraint.reference_elements   :
    SET[0:?] OF variational_parameter;
```



```
  constraining_expression : expression;
END_ENTITY;
(*
```

Attribute definitions:

SELF\explicit_constraint.constrained_elements: The set of constrained instances of **variational_parameter**.

SELF\explicit_constraint.reference_elements: The set of **variational_parameter** instances used as reference elements.

constraining_expression: The **expression** used to define a relationship between the **variational_parameter** instances involved in the constraint.

NOTE 1 The **expression** entity data type from ISO 13584-20 allows the modelling of mathematical expressions whose values are of types NUMBER, BOOLEAN or STRING.

NOTE 2 Any application protocol making use of this schema will need to USE a set of domains from the ISO 10303-50 **mathematical_functions_schema** that is appropriate for the **variational_parameter** value types employed.

5.4.5 free_form_assignment

The **free_form_assignment** entity data type is a type of **free_form_constraint** that represents the assignment of a value, the current value of a mathematical expression involving **variational_parameter** instances, to one or more specified **variational_parameter** instances. The value of the expression in the exchange model is obtained by evaluating it with the current values of all parameters involved in it. In the context of this entity data type, **variational_parameter** instances involved in the expression are regarded as reference elements. The value of the expression is assigned to the value attribute of one or more constrained **variational_parameter** instances.

EXAMPLE Suppose that real-valued **variational_parameter** instances are associated with variables t_1, t_2, t_3, x_1, x_2, where the parameters corresponding to t_1, t_2, t_3 are constrained elements, and those corresponding to x_1, x_2 are reference elements. Then the reference parameters may be used in the **constraining_expression** representation of the expression $\sin^2 x_1 + \sin^2 x_2$, and the resulting value of the expression, evaluated for the current values of those parameters, assigned to the parameters in the set of **constrained_elements**, corresponding to variables t_1, t_2, t_3. Thus the three constrained parameters are all constrained to have the value of the expression involving the two reference parameters.

NOTE 1 A **free_form_assignment** is always a directed constraint. In editing a model containing such a constraint, only the values of the reference elements may be changed by the system user — the values of the constrained elements are then determined by the system in accordance with the new set of reference values.

EXPRESS specification:

```
*)
ENTITY free_form_assignment
  SUBTYPE OF (free_form_constraint);
WHERE
  WR1: SIZEOF(QUERY(q <* SELF\free_form_constraint.constrained_elements |
```



```
    q IN used_variables
    (SELF\free_form_constraint.constraining_expression))) = 0;
  WR2: SIZEOF(QUERY(q <* SELF\free_form_constraint.reference_elements |
    NOT (q IN used_variables(
    SELF\free_form_constraint.constraining_expression)))) = 0;
  WR3: SIZEOF(SELF\free_form_constraint.reference_elements) >= 1;
  WR4: SIZEOF(QUERY(q <* SELF\free_form_constraint.constrained_elements |
    NOT (compatible_spaces(values_space_of(
    SELF\free_form_constraint.constraining_expression),
    q\maths_variable.values_space)))) = 0;
END_ENTITY;
(*
```

Formal propositions:

WR1: No member of the set of **SELF\explicit_constraint.constrained_elements** shall occur in the **SELF\free_form_constraint.constraining_expression**.

WR2: Every member of the set of **SELF\free_form_constraint.reference_elements** shall occur in the **SELF\free_form_constraint.constraining_expression**.

WR3: The set of **SELF\free_form_constraint.reference_elements** shall contain at least one member — otherwise the **constraining_expression** has a constant value and the use of this constraint is inappropriate.

WR4: The possible range of values of the **constraining_expression** shall be compatible with the specified domains of values of all the constrained **variational_parameter** instances.

NOTE 2 The compatibility referred to in the description of **WR4** requires that the specified domain of each constrained instance of **variational_parameter** has a non-null intersection with the values space of the **constraining_expression**. This ensures type compatibility of the values concerned.

NOTE 3 An example of the use of **free_form_assignment** is given in Example 2 of clause F.1.

5.4.6 free_form_relation

The **free_form_relation** entity data type is a type of **free_form_constraint** representing a BOOLEAN-valued relation between the values of **variational_parameter** instances. The constraint is satisfied if the relation has the value TRUE when evaluated with current values of all the parameters involved at the time of model exchange. Within the scope of this type of constraint, **variational_parameter** instances listed as reference elements are regarded as constants used in determining values of the **constrained_elements**.

The use of this constraint to control *sets* of **variational_parameter** instances to have the same associated value requires the **equal_parameter_constraint** of clause 5.4.3 to be employed.

NOTE 1 A **free_form_relation** may represent a directed constraint, an undirected constraint or some combination of the two (hybrid constraint). If only one constrained element is present, the constraint will always be directed. If no reference element is present the constraint will be undirected. However, if more than one constrained element is present, together with one or more reference elements, some freedom will generally remain in the determination of the values of the constrained elements after satisfaction of the constraint relation. In such a case, what remains is an undirected relationship in which one or more of the constrained element values may be changed and the others should adjust accordingly. The following example illustrates such a situation.

EXAMPLE Three real-valued **variational_parameter** instances correspond to variables x, y and z. A **free_form_relation** instance is defined in which the parameters corresponding to x and y are constrained elements, and that corresponding to z is a reference element. An **expression** in this instance, corresponding to the relation $\sqrt{(x^2 + y^2)} < z^2 + 3.0$, specifies the relationship between the three parameters.

In geometric terms, if x, y, z represent cartesian coordinates, the constraint requires x and y to lie inside a circle of radius $z^2 + 3$. The set of all such points form the interior of the solid resulting from rotation of the parabola $x = z^2 + 3$ about the z-axis.

This example illustrates a hybrid constraint, directed with respect to the **variational_parameter** representing z but undirected as regards the relationship between x and y.

The behaviour is clear following modification of one constrained element — the other constrained element changes accordingly. However, there is no uniquely defined behaviour when the reference element is modified. In such cases a fully constrained situation may be achieved through the presence of other constraints or through user interaction following transfer of the model.

EXPRESS specification:

```
*)
ENTITY free_form_relation
  SUBTYPE OF (free_form_constraint);
WHERE
  WR1: 'ISO13584_EXPRESSIONS_SCHEMA.BOOLEAN_EXPRESSION' IN TYPEOF
    (SELF\free_form_constraint.constraining_expression);
  WR2: SIZEOF(QUERY(q <* (SELF\free_form_constraint.constrained_elements +
    SELF\free_form_constraint.reference_elements) |
    NOT (q IN (SELF\free_form_constraint.constraining_expression)))) = 0;
END_ENTITY;
(*
```

Formal propositions:

WR1: The **SELF\free_form_constraint.constraining_expression** attribute shall be of type **boolean_expression** as defined in ISO 13584-20.

WR2: Every **variational_parameter** instance in the sets **SELF\explicit_constraint.constrained_elements** and **SELF\explicit_constraint.reference_elements** shall participate in **SELF\free_form_constraint.constraining_expression**.

NOTE 2 A **boolean_expression** is defined in ISO 13584-20 to be an expression whose range is the BOOLEAN data type defined in ISO 10303-11 (i.e., the expression evaluates to TRUE or FALSE).

NOTE 3 A single **free_form_relation** instance can capture both aspects of a reciprocal dimensional relationship. Assume that two **bound_variational_parameter** instances have been defined and associated with the radius attributes of two different **circle** instances, whose values will be denoted here by r_1 and r_2. Suppose now that that the radius of one of the circles must always be twice that of the other. This relationship may be modelled through the use of a constraint relation corresponding to $r_1^2 = 2.0 * r_2^2$. If both variational parameters are specified as constrained elements then this is an undirected constraint defining a reciprocal relationship. It will ensure the desired behaviour in the receiving system, whichever of the two radii is modified.

An example of this application of the **free_form_relation** is given in Example 1 of clause F.1.

5.4.7 simultaneous_constraint_group

The **simultaneous_constraint_group** entity data type is a type of **variational_representation_item**, defining a set of constraints that are required to hold simultaneously. Such constraint sets may have multiple solutions. Instances of **simultaneous_constraint_group** are also permitted to be members of the set, and this allows recursive structuring of constraint groups. The embedding of instances of **simultaneous_constraint_group** may reflect the sequence of stages of model creation.

EXAMPLE A collection of constraints applying to a two-dimensional sketch may be represented by an instance of **simultaneous_constraint_group**. The sketch may then be used as the basis for the creation of a three-dimensional object participating in an assembly. A group of assembly constraints may be used to position and orient the component in the assembly, and these form a constraint group at a higher level. However, the lower level sketch constraints are still present in the overall definition, and if a parametric change is made at that level they will need to be solved first, and the solution then taken into account at the higher level. For this application it is appropriate to embed the lower level sketch constraint group in the higher-level assembly constraint group.

NOTE 1 The time-dependent sequencing of constraints is outside the scope of this part of ISO 10303, except insofar as it is implied by the embedding of constraint groups within each other.

EXPRESS specification:

```
*)
ENTITY simultaneous_constraint_group
  SUBTYPE OF (variational_representation_item);
  constraint_group : SET[2:?] OF constraint_group_member;
WHERE
  WR1: SIZEOF(QUERY(q <* using_representations(SELF) |
    SIZEOF(QUERY(r <* q.items |
    ('EXPLICIT_CONSTRAINT_SCHEMA.SIMULTANEOUS_CONSTRAINT_GROUP'
    IN TYPEOF(r)) AND (SIZEOF(QUERY(s <* constraint_group |
    (s IN r.constraint_group) AND NOT (r :=: SELF))) > 0))) > 0)) = 0;
  WR2: SIZEOF(QUERY(q <* using_representations(constraint_group[1]) |
    (SIZEOF(QUERY(r <* constraint_group |
    item_in_context(r,q.context_of_items)))
    = SIZEOF(constraint_group)))) > 0;
  WR3: SIZEOF(QUERY(q <* constraint_group |
    (('EXPLICIT_CONSTRAINT_SCHEMA.EXPLICIT_CONSTRAINT' IN TYPEOF(q))
    AND (SIZEOF(QUERY(r <* q.constrained_elements |
    SIZEOF(QUERY(s <* constraint_group |
    r IN s.reference_elements)) > 0)) > 0)))) = 0;
END_ENTITY;
(*
```

Attribute definitions:

constraint_group: A set with members of type **explicit_constraint** or **simultaneous_constraint_group**.

Formal propositions:

WR1: Any constraint in the group shall be a member of no other **simultaneous_constraint_group**.

WR2: There shall exist at least one **representation_context** that contains all members of the group of explicit constraints (including members of embedded groups).

WR3: No instance that serves as a constrained element in any one constraint in the group shall serve as a reference element in any other. This restriction shall apply at each level in the constraint hierarchy, but not between the levels defined by embedding of constraint groups.

NOTE 2 In the absence of WR1, a given instance of **explicit_constraint** could belong to two or more instances of **simultaneous_constraint_group**. This would imply that all the constraint groups containing that constraint could be assembled into a single, larger, constraint group. WR1 ensures that all such constraint groups are maximal, and that such overlaps in membership cannot occur. Additionally, it ensures that a constraint that is a member of an embedded group cannot also be a member at any other level in a constraint hierarchy.

NOTE 3 An important consequence of WR2 concerns the dimensionality of elements in shape models. The geometric subtypes of **explicit_constraint** defined in clause 7 of this part of ISO 10303 are also subtypes of **geometric_representation_item** (see ISO 10303-42), and have an associated dimensionality. If one or more explicit constraints in the group are **geometric_representation_item** instances, then any common **representation_context** shared by the members must be a **geometric_representation_context** — this is required by a local rule in the definition of **geometric_representation_item**. Then WR2 ensures that all geometric constraints in an instance of **simultaneous_constraint_group** necessarily have the same dimensionality, consistent with the dimensionality of their shared **geometric_representation_context**. The fact that WR2 applies equally to embedded constraint groups ensures that complete hierarchies of geometric constraints are dimensionally consistent.

NOTE 4 In the absence of **WR3** a situation could arise such as the following:

— A **variational_parameter** with name `'x'` is a reference element in a free-form constraint F, which has a **variational_parameter** with name `'y'` as a constrained element;

— Conversely, free-form constraint G uses the **variational_parameter** named `'y'` as a reference element and that named `'x'` as a constrained element.

Here constraint F forbids the value associated with `'y'` to be changed to achieve constraint satisfaction, while constraint G similarly forbids any change in the value associated with `'x'`. Hence the value attributes of both **variational_parameter** instances must remain fixed within the context of the **simultaneous_constraint_group**. This and similar situations are avoided by requiring that no instance shall play the role of both reference and constrained element in constraints occurring at the same level in the group. Thus any instance involved in constraints at a given level of the constraint hierarchy must play the role of either reference element or constrained element to the group as a whole at that level, but not both.

The situation is different in the following cases:

— It is permitted that a constrained element from an embedded constraint group at a lower level shall be a reference element for a constraint group at a higher level;

— In a sequentially imposed set of constraints, a constrained element from one constraint may play the role of a reference element in a succeeding constraint.

However, the representation of time-dependent sequences of constraints is out of scope of this part of ISO 10303.

Clause 6.1, p.43
Replace the reference to the **parameterization_schema** *by the following:*

```
REFERENCE FROM parameterization_schema              -- ISO 10303-108
  (bound_variational_parameter,
   fixed_instance_attribute_set,
   instance_attribute_reference,
   unbound_variational_parameter);
```

Clause 6.2, p.44
In the paragraph following the enumerated list, replace **model_parameter** *by* **variational_parameter**.
Replace NOTE *by* NOTE 2 *immediately following this paragraph.*

Clause 6.3.1, p.45
In the Example, replace **model_parameter** *by* **variational_parameter**.

Clause 6.3.3, p.47
In Example 2, replace **bound_model_parameter** *by* **bound_variational_parameter** *in line 5, and replace* model parameters *by* variational parameters *in line 9.*

Clause 6.4.1, p.49
In the first item of the enumerated list, replace **bound_model_parameter** *by* **bound_variational_parameter**, *and replace* model parameter *by* variational parameter.
In the second item of the enumerated list, replace **unbound_model_parameter** *by* **unbound_variational_parameter**, *and replace* model parameter *by* variational parameter.
Replace the EXPRESS code by the following:

EXPRESS specification:

```
*)
FUNCTION invalidate_vrep_item(item : variational_representation_item)
                                   : BOOLEAN;

  LOCAL
    reps    : SET[1:?] OF representation := using_representations(item);
    svri    : SET[1:?] OF variational_representation_item;
    iar     : instance_attribute_reference;
    i       : INTEGER;
    n       : INTEGER := HIINDEX(reps);
  END_LOCAL;

  IF ('PARAMETERIZATION_SCHEMA.BOUND_VARIATIONAL_PARAMETER' IN TYPEOF(item))
  THEN
    IF 'PARAMETERIZATION_SCHEMA.INSTANCE_ATTRIBUTE_REFERENCE'
      IN TYPEOF(item\generic_variable.interpretation.semantics)
    THEN
      BEGIN
        iar := item\generic_variable.interpretation.semantics;
        IF (reps <> using_representations(iar.owning_instance))
        THEN
```



```
        RETURN(TRUE);
      END_IF;
    END;
  ELSE RETURN(TRUE); -- parameter not attached to an instance attribute
  END_IF;
END_IF;

IF ('PARAMETERIZATION_SCHEMA.UNBOUND_VARIATIONAL_PARAMETER'
  IN TYPEOF(item))
THEN
  BEGIN
    REPEAT i := 1 TO n;
      svri := QUERY(q <* reps[i].items |
        'EXPLICIT_CONSTRAINT_SCHEMA.FREE_FORM_CONSTRAINT' IN TYPEOF(q));
      IF SIZEOF(QUERY(r <* svri |
        item IN (r.reference_elements + r.constrained_elements))) = 0
      THEN
        RETURN(TRUE);
      END_IF;
    END_REPEAT;
  END;
END_IF;

IF ('PARAMETERIZATION_SCHEMA.FIXED_INSTANCE_ATTRIBUTE_SET'
  IN TYPEOF(item))
THEN
  REPEAT i := 1 TO SIZEOF(item.fixed_attributes);
    IF (reps <> using_representations(item.fixed_attributes[i]))
    THEN
      RETURN(TRUE);
    END_IF;
  END_REPEAT;
END_IF;

IF ('EXPLICIT_CONSTRAINT_SCHEMA.EXPLICIT_CONSTRAINT' IN TYPEOF(item))
THEN
  IF SIZEOF(QUERY(q <*
    (item.reference_elements + item.constrained_elements) |
    reps <> using_representations(q))) > 0
  THEN
    RETURN(TRUE);
  END_IF;
END_IF;

IF ('VARIATIONAL_REPRESENTATION_SCHEMA.
     AUXILIARY_GEOMETRIC_REPRESENTATION_ITEM' IN TYPEOF(item))
THEN
  BEGIN
    REPEAT i := 1 TO n;
      svri := QUERY(q <* reps[i].items |
        'EXPLICIT_GEOMETRIC_CONSTRAINT_SCHEMA.
         EXPLICIT_GEOMETRIC_CONSTRAINT' IN TYPEOF(q));
      IF SIZEOF(QUERY(r <* svri | item IN r.reference_elements)) = 0
      THEN
        RETURN(TRUE);
      END_IF;
    END_REPEAT;
  END;
```



```
  END_IF;

  RETURN(FALSE); -- no invalid cases have been found

END_FUNCTION;
(*
```

Clause 7.1, p.52

Replace the reference to the measure_schema *by the following:*

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

Clause 7.2.2, p.55

Replace the entire text of this clause by the following:

This part of ISO 10303 defines a dimension in terms of the numerical value of an attribute of certain entity data type instances. The type of the numerical value may be one of the SELECT values of **measure_value** as defined in ISO 10303-41.

A dimensional attribute may have an associated **bound_variational_parameter** or belong to a **fixed_instance_attribute_set** (see clause 4.4.8).

If referenced by an instance of **bound_variational_parameter**, the dimensional attribute is potentially subject to modification in the receiving system, subject to restrictions imposed by

— the parameter's specified domain;

— any free-form constraints in which the parameter participates;

— any explicit geometric or other constraints that affect the entity instance to which the attribute belongs.

Alternatively, if a numerical attribute defining a dimension is a member of a **fixed_instance_attribute_set**, the intention is that it should be constrained to be invariant in the receiving system.

EXAMPLE 1 A bound **variational_parameter** may be associated with a dimensional value in a shape model, but an unbound **variational_parameter**, by definition, has no such direct association. For example, the **x**, **y** and **z** dimensional attributes of a **block** solid, as defined in ISO 10303-42, may be associated with **bound_variational_parameter** instances with name attributes `'L','W'` and `'H'` respectively. Then relations $L = T^2$, $W = 2T$ and $H = 2 + \cos T$ may be modelled by **free_form_assignment** constraints, in which T represents an **unbound_variational_parameter** named `'T'`. This unbound parameter will not be directly associated with a dimension, though it will indirectly control the values of three dimensions.

EXAMPLE 2 In the previous example, the dimensions of the block can only be changed by editing the parameter named `'T'`. This is because the values of the parameters named `'L','W','H'` are specified through free-form assignments in which `'T'` is the reference element and they are constrained elements. In such a case the values of the constrained elements can only be edited by changing the value of the reference element. This shows that the

value of a dimensional attribute having a bound parameter cannot always be edited directly. Account must always be taken of other constraints in which the parameter participates.

Finally, it should be emphasized once again that logical or dimensional constraints in a shape model are redundant for purposes of static model transfer. They do not affect the shape of the exchanged model. The assertions they make are expected to be true when a model is exchanged, their effect only becoming apparent when that model is modified in the receiving system. In that event they are intended to govern the nature of the permissible changes that may be made to it.

Clause 7.3.10, non_negative_length_measure*, p. 60*

Remove this clause from the document

Clause 7.4.19, p.78

At the end of this clause, replace NOTE *by* NOTE 4.

Clause 7.4.24, p.83

Replace EXAMPLE 1 *by* EXAMPLE.

Clause 8.4.10, p.104

Replace the EXPRESS code for **positioned_sketch** *on p.104 by the following, in which WHERE rules WR1 and WR2 have been modified:*

```
ENTITY positioned_sketch
  SUBTYPE OF (geometric_representation_item);
  sketch_basis       : sketch_basis_select;
  auxiliary_elements :
    SET[0:?] OF auxiliary_geometric_representation_item;
WHERE
  WR1: NOT (('GEOMETRY_SCHEMA.CURVE_BOUNDED_SURFACE' IN
    TYPEOF(sketch_basis)) AND NOT ('GEOMETRY_SCHEMA.PLANE' IN
    TYPEOF(sketch_basis\curve_bounded_surface.basis_surface)));
  WR2: NOT (('TOPOLOGY_SCHEMA.FACE_SURFACE' IN TYPEOF(sketch_basis))
    AND NOT ('GEOMETRY_SCHEMA.PLANE'
    IN TYPEOF(sketch_basis\face_surface.face_geometry)));
  WR3: SIZEOF(QUERY(q <* auxiliary_elements | (SIZEOF(TYPEOF(q) *
    ['GEOMETRY_SCHEMA.POINT','GEOMETRY_SCHEMA.CURVE']) = 0))) = 0;
  WR4: SIZEOF(QUERY(q <* auxiliary_elements |
    q\geometric_representation_item.dim <> 3)) = 0;
END_ENTITY;
```

Annex A, pp. 113 and 114

The name changes made require replacement of the the Table of Short names of entities.
Replace the current Table A.1 by the following:

Table A.1 – Short names of entities

Entity data type names	Short names
agc_with_dimension	AGWTDM
angle_geometric_constraint	ANGMCN
auxiliary_geometric_representation_item	AGRI
bound_variational_parameter	BNVRPR
bound_parameter_environment	BNPREN
cdgc_with_dimension	CDWTDM
clgc_with_dimension	CLWTDM
coaxial_geometric_constraint	CXGMCN
curve_distance_geometric_constraint	CDGC
curve_length_geometric_constraint	CLGC
curve_segment_set	CRSGST
curve_smoothness_geometric_constraint	CSGC
defined_constraint	DFNCNS
equal_parameter_constraint	EQPRCN
explicit_constraint	EXPCNS
explicit_geometric_constraint	EXGMCN
fixed_element_geometric_constraint	FEGC
fixed_instance_attribute_set	FIAS
free_form_assignment	FRFRAS
free_form_constraint	FRFRCN
free_form_relation	FRFRRL
generated_finite_numeric_space	GFNS
implicit_explicit_positioned_sketch_relationship	IEPSR
implicit_intersection_curve	IMINCR
implicit_model_intersection_curve	IMIC
implicit_planar_curve	IMPLCR
implicit_planar_intersection_point	IPIP
implicit_planar_projection_point	IPPP
implicit_point_on_plane	IPOP
implicit_projected_curve	IMPRCR
implicit_silhouette_curve	IMSLCR
incidence_geometric_constraint	INGMCN

Table A.1 – (continued)

Entity data type names	Short names
instance_attribute_reference	INATRF
near_point_relationship	NRPNRL
neutral_sketch_representation	NTSKRP
parallel_geometric_constraint	PRGMCN
parallel_offset_geometric_constraint	POGC
pdgc_with_dimension	PDWTDM
perpendicular_geometric_constraint	PRG2
pgc_with_dimension	PGWTDM
pogc_with_dimension	PGW0
point_distance_geometric_constraint	PDGC
positioned_sketch	PSTSKT
radius_geometric_constraint	RDGMCN
repositioned_neutral_sketch	RPNTSK
rgc_with_dimension	RGWTDM
rigid_subsketch	RGDSBS
sdgc_with_dimension	SDWTDM
simultaneous_constraint_group	SMCNGR
skew_line_distance_geometric_constraint	SLDGC
subsketch	SBSKTC
surface_distance_geometric_constraint	SDGC
surface_patch_set	SRPTST
surface_smoothness_geometric_constraint	SSGC
swept_curve_surface_geometric_constraint	SCSGC
swept_point_curve_geometric_constraint	SPCGC
symmetry_geometric_constraint	SYGMCN
tangent_geometric_constraint	TNGMCN
unbound_parameter_environment	UNPREN
unbound_variational_parameter	UNVRPR
unbound_variational_parameter_semantics	UVPS
variational_current_representation_relationship	VCRR
variational_parameter	VRTPRM
variational_representation	VRTRPR
variational_representation_item	VRRPIT

Annex B, p. 115

The changes identified in this Technical Corrigendum require the document identifiers and the schema information object identifiers to be changed. Remove the object identifier for the part and replace with the following:

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

```
{ iso standard 10303 part(108) version(2) }
```

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.

Remove the object identifier for the **parameterization_schema** *and replace with the following:*

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

```
{ iso standard 10303 part(108) version(2) schema(1)
            parameterization-schema(1) }
```

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

Remove the object identifier for the **explicit_constraint_schema** *and replace with the following:*

To provide for unambiguous identification of the explicit-constraint-schema in an open information system, the object identifier

```
{ iso standard 10303 part(108) version(2) schema(1)
          explicit-constraint-schema(2) }
```

is assigned to the **explicit_constraint_schema** (see clause 5). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Remove the object identifier for the **variational_representation_schema** *and replace with the following:*

To provide for unambiguous identification of the variational-representation-schema in an open information system, the object identifier

```
{ iso standard 10303 part(108) version(2) schema(1)
       variational-representation-schema(3) }
```

is assigned to the **variational_representation_schema** (see clause 6). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Remove the object identifier for the **explicit_geometric_constraint_schema** *and replace with the following:*

To provide for unambiguous identification of the explicit-geometric-constraint-schema in an open information system, the object identifier


```
{ iso standard 10303 part(108) version(2) schema(1)
    explicit-geometric-constraint-schema(4) }
```

is assigned to the **explicit_geometric_constraint_schema** (see clause 7). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Remove the object identifier for the **sketch_schema** *and replace with the following:*

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

```
{ iso standard 10303 part(108) version(2) schema(1) sketch-schema(5)
                                    }
```

is assigned to the **sketch_schema** (see clause 8). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Annex D, p. 118

The changes identified in this Technical Corrigendum require four of the EXPRESS-G diagrams to be changed. Replace Figures D.1, D.3, D.9 and D.10, respectively, by the following diagrams:

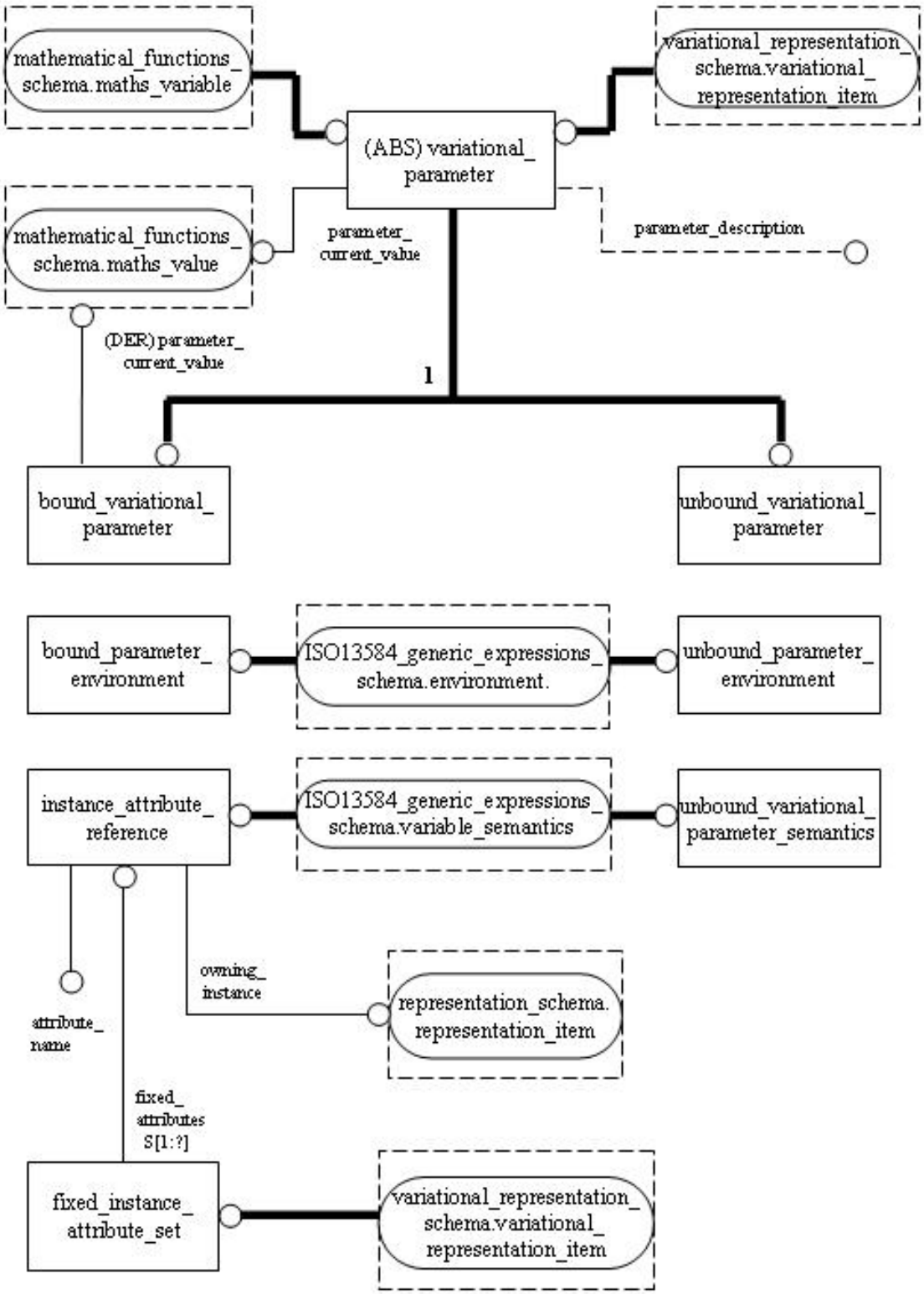

Figure D.1 – EXPRESS-G diagram of the parameterization_schema (1 of 2)

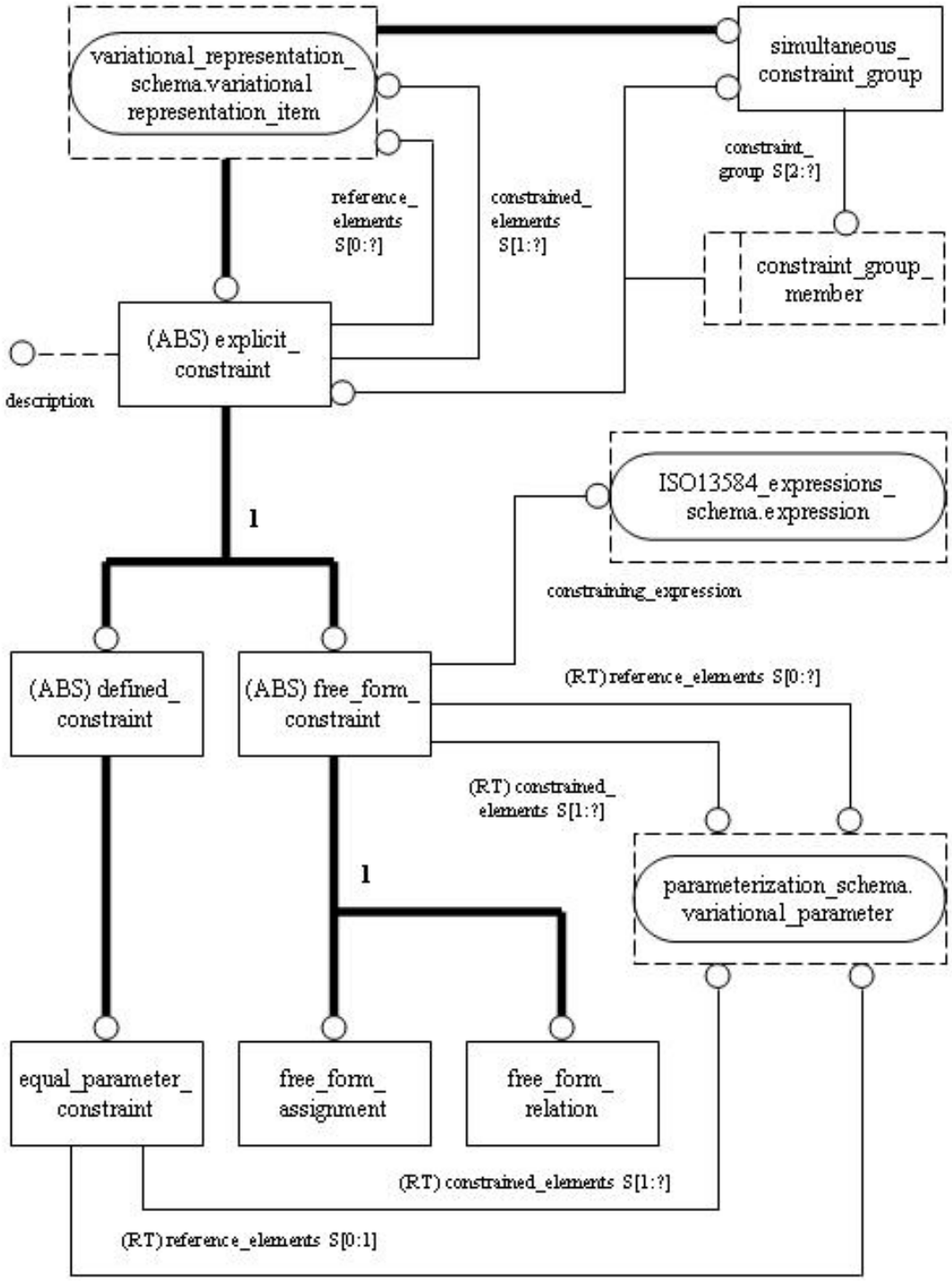

Figure D.3 – EXPRESS-G diagram of the explicit_constraint_schema (1 of 1)

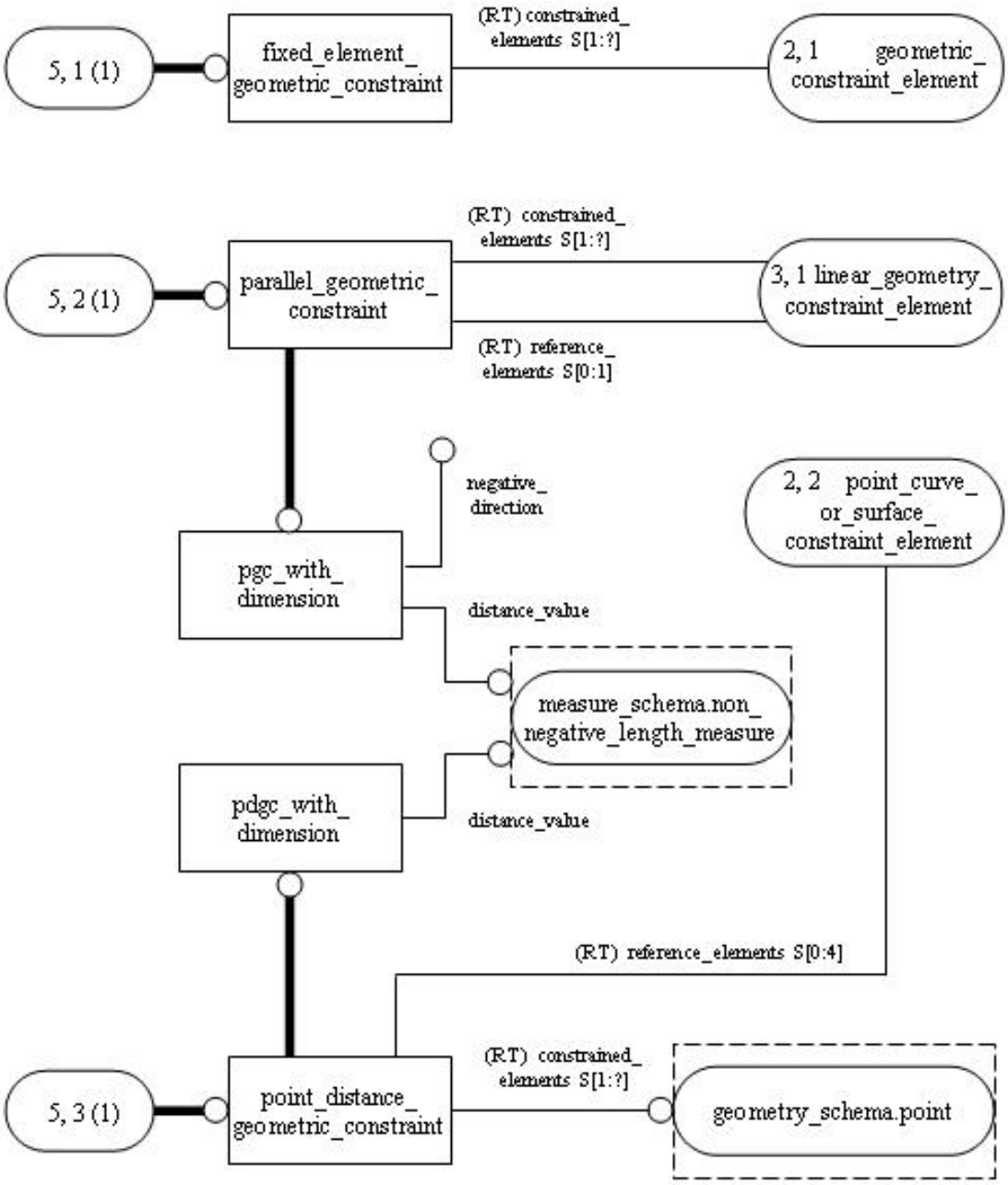

Figure D.9 – EXPRESS-G diagram of the explicit_geometric_constraint_schema (5 of 10)

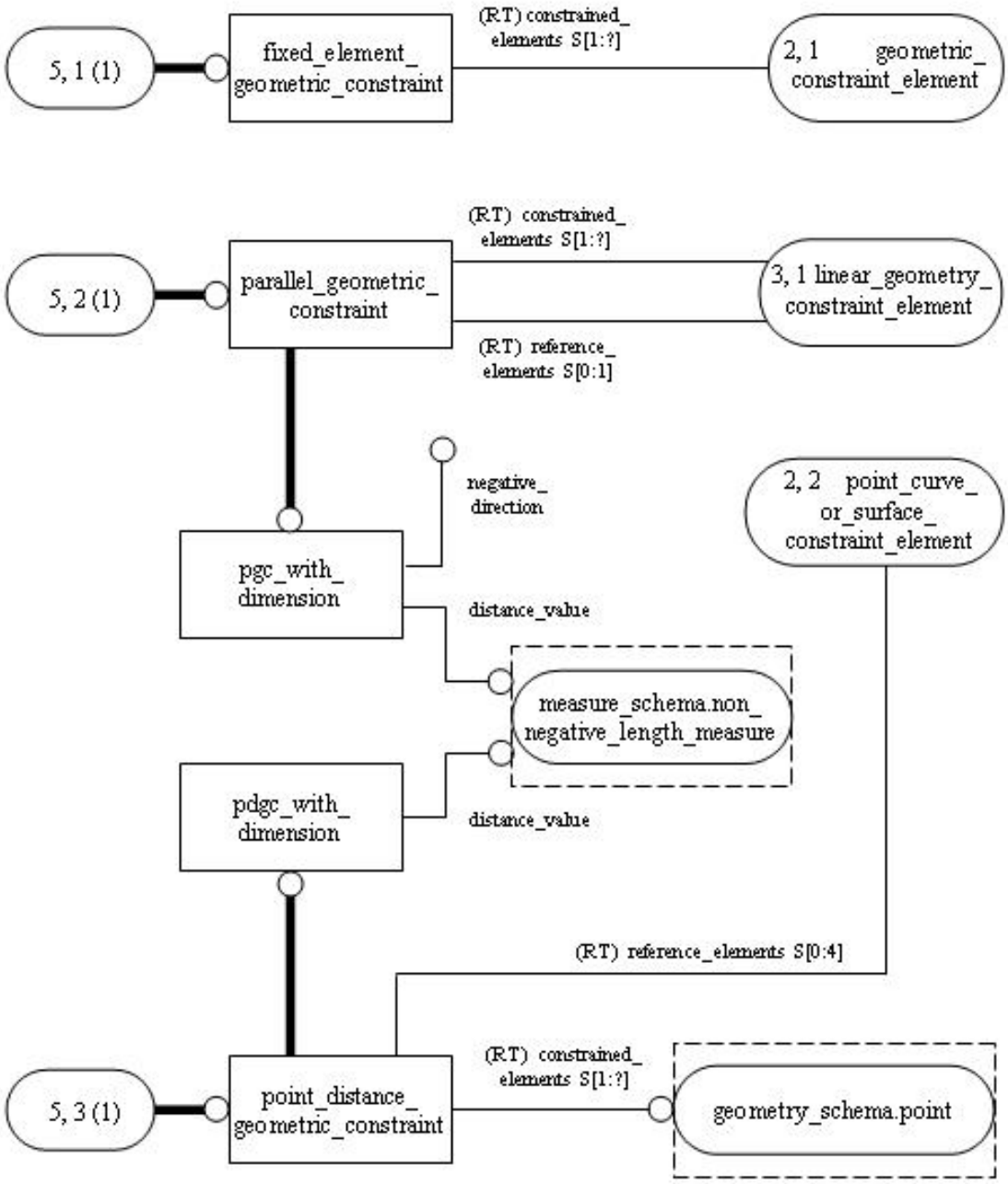

Figure D.10 – EXPRESS-G diagram of the explicit_geometric_constraint_schema (6 of 10)

Clause E.1, p. 138, last paragraph
Replace ISO 10303-203 *by* ISO 10303-203:1994.

Clause E.2.1, p. 139
In the fifth item of the bulleted list, replace **model_parameter** *by* **variational_parameter**. *Make the same replacement in line 2 of the* EXAMPLE

Clause F.1, p. 142
Replace the entire text of this clause by the following:

The basic principle for the referencing of attributes and the binding of **variational_parameter** instances to them was illustrated in clause 4.2.2. It is recommended that the earlier example be studied before reading the following additional examples, because it provides some of the necessary preliminary understanding. The further illustrations show the use of free-form constraints for the specification of mathematical relationships between instances of **variational_parameter** subtypes.

F.1.1 Example 1

In this and the following example, entity data types from the ISO 10303 integrated generic resources are treated as though they are instantiable elements of an application protocol. This first example illustrates the modelling of a reciprocal constraint requiring the radius of circle C6 to be 2.5 times that of circle C5 (and conversely, the radius of C5 to be 0.4 times that of C6). Thus both **variational_parameter** instances required are constrained elements (see clause 4.2.1), since if the value of one is changed in the receiving system the other is required to adjust correspondingly.

In the example, all entity data types occurring are defined in this part of ISO 10303 unless otherwise mentioned. The ISO 10303-21 transfer file must first contain definitions of the circle instances. The necessary entity data types are specified in ISO 10303-42. The current values of the radii of the circles are 10.0 and 25.0 respectively:

```
#200 = AXIS2_PLACEMENT_3D(.....);
#210 = AXIS2_PLACEMENT_3D(.....);
.....
#500 = CIRCLE('C5', #200, 10.0);
#510 = CIRCLE('C6', #210, 25.0);
```

Next, the file must include two instances of **bound_variational_parameter**, each with domain specified by an instance of **finite_real_interval** (all entities defining such domains are taken from ISO 10303-50):

```
#520 = FINITE_REAL_INTERVAL(8.0, .CLOSED., 15.0, .CLOSED.);
#530 = BOUND_VARIATIONAL_PARAMETER('P1', #520, 'P1', 'C5 RADIUS', *);

#540 = FINITE_REAL_INTERVAL(20.0, .CLOSED., 30.0, .CLOSED.);
#550 = BOUND_VARIATIONAL_PARAMETER('P2', #550, 'P2', 'C6 RADIUS', *);
```

Now the binding must be established between these two **bound_variational_parameter** instances and the radius attributes of the two circles. The mechanism for achieving this is defined in ISO 13584-20, and the relations between the entities defined there and their subtypes as defined in this part of ISO 10303 are depicted in an EXPRESS-G diagram in clause F.1.3 following the examples. For each binding, an instance of **environment** (subtyped in this part of ISO 10303 as **bound_parameter_environment** and **unbound_parameter_environment**) must exist. The **environment** entity data type has two attributes,

as follows:

syntactic_representation: a reference to a mathematical variable (in the present case an instance of **bound_variational_parameter**);

semantics: a reference to an instance of the ISO 13584-20 entity data type **variable_semantics**, which is intended to indicate the significance of the variable in its modelling context.

In this part of ISO 10303 a subtype of **variable_semantics** called **instance_attribute_reference** is provided. This simply declares the specific attribute (by name) and the specific entity data type instance (by reference) that the **variational_parameter** is to be associated with. The next fragment of the exchange file creates this association:

```
#560 = INSTANCE_ATTRIBUTE_REFERENCE
         ('GEOMETRY_SCHEMA.CIRCLE.RADIUS', #500);
#570 = BOUND_PARAMETER_ENVIRONMENT(#530, #560);
#580 = INSTANCE_ATTRIBUTE_REFERENCE
         ('GEOMETRY_SCHEMA.CIRCLE.RADIUS', #510);
#590 = BOUND_PARAMETER_ENVIRONMENT(#550, #580);
```

Note that the **radius** attribute is specified in fully qualified form, as required by an informal proposition applying to the **instance_attribute_reference** entity data type.

The parameter bindings having been established, an instance of **free_form_relation** is now required, to model the mathematical relationship between the values of the **bound_variational_parameter** instances. The constraint instance is #600 below, in which both parameters occur as constrained elements. The first two attribute values of this instance correspond to name and textual description attributes inherited from the supertypes **representation_item** and **explicit_constraint** respectively. These are followed by the set of (two) constrained elements, the (empty) set of reference elements and a reference to the **constraining_expression**, represented by #610. This and the two following instances build a relationship asserting that the value of parameter #550 is 2.5 times that of the parameter #530:

```
#600 = FREE_FORM_RELATION('FFR1', 'RADIUS-RELATION',
         (#530, #550), (), #610);
#610 = COMPARISON_EQUAL((#550, #620));
#620 = MULT_EXPRESSION((#630, #530));
#630 = REAL_LITERAL(2.5);
```

Instances #610 – #630 are of entities defined in the **expressions_schema** of ISO 13584-20. The modelled relationship is true in the model as transmitted; the existence of the constraint is intended to ensure that the relationship can be maintained if the model is edited in the receiving system.

NOTE 1 The first attribute value in instances #560 and #580 corresponds to the attribute **attribute_name**, declared in the **instance_attribute_reference** entity data type defined in clause 4.4.6. In the present context this value must be interpreted as an *intelligent string*, i.e., an EXPRESS string with semantics. This has obvious implications for implementers; analysis of the string by reference to the schemas in current use is necessary to enable the appropriate references to be established in the model. However, that appears to be unavoidable for the transfer of models containing specific relationships between parameters and instance attributes, regardless of the precise details of how the relationships are represented in the underlying EXPRESS schema.

NOTE 2 It will be necessary for the label referred to above to allow references to attributes of underlying entities.

EXAMPLE The string 'GEOMETRY_SCHEMA.TRIMMED_CURVE.BASIS_CURVE.RADIUS' could be used as the **attribute_name** value in constraining the radius of a circular arc expressed as an ISO 10303-42 **trimmed_curve** instance.

NOTE 3 It will similarly be necessary to provide for references to individual members of aggregate-valued attributes.

F.1.2 Example 2

The second example shows the use of **unbound_variational_parameter**. The details are much the same as in the previous example, but it is necessary to use a different subtype of ISO 13584-20 **environment** for an unbound parameter. The example modelled here was used in clause 4.2.1 to illustrate a situation where an unbound parameter may be used. To recapitulate, it is desired to constrain the height h and radius r of an instance of ISO 10303-42 **right_circular_cylinder** to satisfy $h = t^2 + 1,\ r = 3t - 2$, where t is a parameter that does not represent any physical quantity in the model. In this case the model parameters corresponding to h and r are required to be bound to attributes of the **right_circular_cylinder** instance, but that corresponding to t is unbound.

The circular cylinder is represented by the ISO 10303-42 entity data type instances

```
#340 = AXIS1_PLACEMENT(...);
#350 = RIGHT_CIRCULAR_CYLINDER('CYL1', #340, 10.0, 7.0);
```

Here the current values of the radius and height are specified as 10.0 and 7.0 respectively. Two **bound_variational_parameter** instances are next defined and bound to the relevant attributes of #350, as shown in the previous example:

```
#360 = FINITE_REAL_INTERVAL(2.0, .CLOSED., 20.0, .CLOSED.);
#370 = BOUND_VARIATIONAL_PARAMETER('H', #360, 'H', 'CYL_HT', *);
#380 = INSTANCE_ATTRIBUTE_REFERENCE
         ('GEOMETRIC_MODEL_SCHEMA.RIGHT_CIRCULAR_CYLINDER.HEIGHT',
          #350);
#390 = BOUND_PARAMETER_ENVIRONMENT(#370, #380);

#400 = FINITE_REAL_INTERVAL(1.0, .CLOSED., 12.0, .CLOSED.);
#410 = BOUND_VARIATIONAL_PARAMETER('R', #360, 'R', 'CYL_RAD', *);
#420 = INSTANCE_ATTRIBUTE_REFERENCE
         ('GEOMETRIC_MODEL_SCHEMA.RIGHT_CIRCULAR_CYLINDER.RADIUS',
          #350);
#430 = BOUND_PARAMETER_ENVIRONMENT(#410, #420);
```

Next the **unbound_variational_parameter** corresponding to t is defined, and **unbound_parameter_semantics** is instanced rather than **instance_attribute_reference**. It may be noted that **unbound_parameter_semantics** has no attributes, and that the instance #460 therefore conveys no information, but its presence is required by the structures defined in ISO 13584-20.

```
#440 = REAL_INTERVAL_FROM_MIN(2.0, .CLOSED.);
#450 = UNBOUND_VARIATIONAL_PARAMETER('T', #440, 'T',
         'FREE_PARAM', 3.0);
#460 = UNBOUND_PARAMETER_SEMANTICS;
#470 = UNBOUND_PARAMETER_ENVIRONMENT(#450, #460);
```


Note that, whereas in an instance of **bound_variational_parameter** the value of the parameter is represented by an asterisk (denoting a derived value), in an instance of **unbound_variational_parameter** it is given explicitly (as 3.0 in the present case).

Finally, the relationships between the parameters are established by means of two **free_form_assignment** instances, in both of which the parameter `#450` corresponding to t plays the role of a reference element:

```
#480 = FREE_FORM_ASSIGNMENT('FFA1', 'HEIGHT-VALUE',
         (#370), (#450), #490);
#490 = PLUS_EXPRESSION((#500, #510));
#500 = MULT_EXPRESSION((#450, #450));
#510 = REAL_LITERAL(1.0);

#520 = FREE_FORM_ASSIGNMENT('FFA2', 'RADIUS-VALUE',
         (#410), (#450), #530);
#530 = MINUS_EXPRESSION((#540, #550));
#540 = MULT_EXPRESSION((#560, #450));
#550 = REAL_LITERAL(2.0);
#560 = REAL_LITERAL(3.0);
```

Figure F.1, p. 146

Replace Figure F.1 by the following figure:

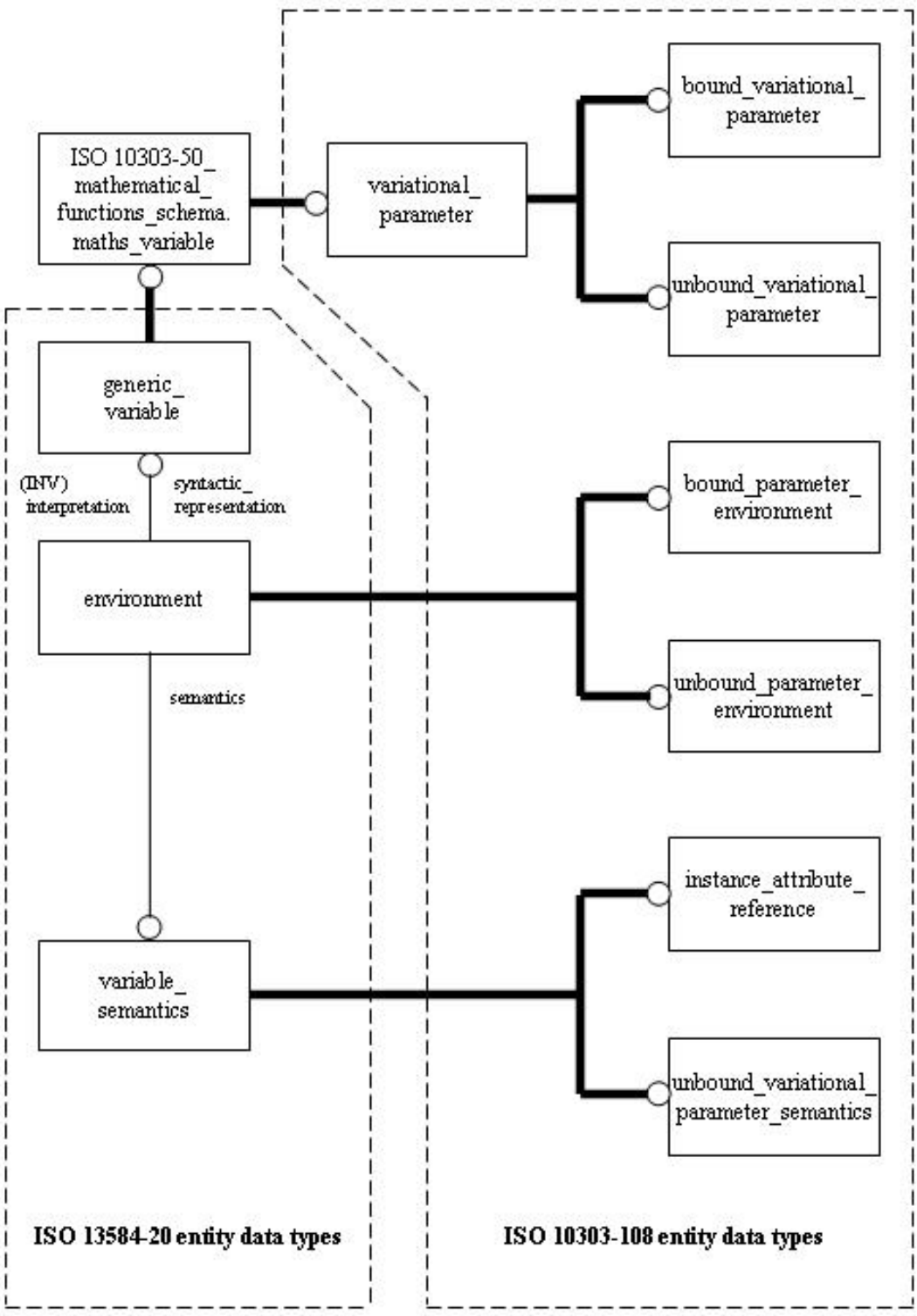

Figure F.1 – Key relationships between ISO 10303-108 parameterization_schema and ISO 13548 generic_expressions_schema

Clause F.3, p. 149

In the first item of the second bulleted list, replace **bound_model_parameter** *by* **bound_variational_-parameter**

Index, p. 153

Delete the entry non_negative_length_measure *from the Index*

