INTERNATIONAL STANDARD

ISO 10303-508

First edition
2001-04-15

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

Part 508: **Application interpreted construct: Non-manifold surface**

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

Partie 508: Établissement interprété d'application: Surface non manifold

Reference number
ISO 10303-508:2001(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

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

Printed in Switzerland

Contents

Figures

Tables

Foreword

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

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

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 International Standard may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

International Standard ISO 10303-508 was prepared by Technical Committee ISO/TC 184 *Industrial automation systems and integration*, Subcommittee SC 4, *Industrial data*.

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

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

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

<http://www.nist.gov/sc4/editing/step/titles/>

Annexes A and B form a normative part of this part of ISO 10303. Annexes C and D are for information only.

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation and exchange of product data. The objective is to provide a neutral mechanism capable of describing product data throughout the life cycle of a product independent from any particular system. The nature of this description makes it suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases and archiving.

This International Standard is organized as a series of parts, each published separately. The parts of ISO 10303 fall into one of the following series: description methods, integrated resources, application interpreted constructs, application protocols, abstract test suites, implementation methods, and conformance testing. The series are described in ISO 10303–1. This part of ISO 10303 is a member of the application interpreted constructs series.

An application interpreted construct (AIC) provides a logical grouping of interpreted constructs that supports a specific functionality for the usage of product data across multiple application contexts. An interpreted construct is a common interpretation of the integrated resources that supports shared information requirements among application protocols.

This document specifies the application interpreted construct for the description of geometric shapes by means of non-manifold surface models. It includes the geometric and topological resources to define non-manifolds that may consist of elementary and sculptured curves and surfaces.

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

1 Scope

This part of ISO 10303 specifies the interpretation of the integrated resources to satisfy requirements for the description of geometric shapes by means of non-manifold surface models.

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

— 3D points;

— points defined in the parameter space of curves or surfaces;

— 3D curves;

— curves defined in the parameter space of surfaces;

NOTE - Such curves are also known as pcurves or cons, which are acronyms for parametrised curve and curve on surface.

— the elementary curve types line, circle, ellipse, parabola, and hyperbola;

— intersection curves;

— polylines that consist of at least three points;

— the elementary surface types plane, cylinder, cone, torus, and sphere;

— swept surfaces created by rotation or linear extrusion of a curve;

— sculptured curves and surfaces;

— trimming of curves and surfaces using topological entities;

— composition of curves and surfaces using topological entities;

— replication of curves, surfaces, and surface models;

— 3D offsets of curves and surfaces;

- non-manifolds.

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

- unbounded geometry;
- self-intersecting geometry;
- geometry in a 2D cartesian coordinate space;
- replication of points;
- topology without an association to a corresponding geometric domain.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO 10303. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO 10303 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 8824–1:1998, *Information technology – Abstract Syntax Notation Onw (ASN.1): Specification of basic notation.*

ISO 10303–1:1994, *Industrial automation systems and integration – Product data representation and exchange – Part 1 : Overview and fundamental principles.*

ISO 10303–11:1994, *Industrial automation systems and integration – Product data representation and exchange – Part 11 : Description methods: The EXPRESS language reference manual.*

ISO 10303–41:1994, *Industrial automation systems and integration – Product data representation and exchange – Part 41 : Integrated generic resources: Fundamentals of product description and support.*

ISO 10303–42:1994, *Industrial automation systems and integration – Product data representation and exchange – Part 42 : Integrated generic resources: Geometric and topological representation.*

ISO 10303–43:1994, *Industrial automation systems and integration – Product data representation and exchange – Part 43 : Integrated generic resources: Representation structures.*

ISO 10303–202:1996, *Industrial automation systems and integration – Product data representation and exchange: – Part 202: Application protocol: Associative draughting.*

NOTE - ISO 10303-202 is referenced normatively solely for the definition of the term AIC.

ISO 10303–511:2001, *Industrial automation systems and integration – Product data representation and exchange – Part 511 : Application interpreted construct: Topologically bounded surface.*

3 Terms, definitions, and abbreviations

3.1 Terms defined in ISO 10303–1

For the purposes of this part of ISO 10303, the following terms defined in ISO 10303-1 apply:

- abstract test suite (ATS);
- application;
- application context;
- application protocol (AP);
- data;
- data exchange;
- generic resource;
- implementation method;
- information;
- integrated resource;
- interpretation;
- model;
- product;
- product data;
- structure.

3.2 Terms defined in ISO 10303–42

For the purposes of this part of ISO 10303, the following terms defined in ISO 10303-42 apply:

- boundary;
- boundary representation solid model;
- connected;
- coordinate space;
- curve;
- dimensionality;
- domain;
- parameter space;
- self-intersect;
- surface.

3.3 Terms defined in ISO 10303–202

For the purposes of this part of ISO 10303, the following terms defined in ISO 10303-202 apply:

3.3.1
application interpreted construct (AIC)
a logical grouping of interpreted constructs that supports a specific function for the usage of product data across multiple application contexts.

3.4 Terms defined in ISO 10303–511

For the purposes of this part of ISO 10303, the following terms defined in ISO 10303-511 apply:

- advanced face;
- sculptured surface;
- swept surface.

3.5 Other terms and definitions

For the purposes of this part of ISO 10303, the following terms and definitions apply:

3.5.1
2-manifold
a shape where on any point of its boundary one may create a sufficiently small sphere so that the interior of the sphere is divided into exactly two regions by this boundary. The boundary may typically consist of edges and faces.

NOTE - This definition eliminates self-intersection of surfaces, surface intersections that are not along edges, and edges joining three or more faces.

3.5.2
non-manifold
a surface model that uses topological constructs to define its boundaries and connectivity and that includes either at least two **connected_face_set**s sharing one **face** or more than two **face**s sharing one **edge**.

3.6 Abbreviations

For the purposes of this part of ISO 10303, the following abbreviations apply.

AIC	application interpreted construct
AP	application protocol
ATS	abstract test suite

4 EXPRESS short listing

This clause specifies the EXPRESS schema that uses elements from the integrated resources and contains the types, entity specializations, and functions that are specific to this part of ISO 10303.

NOTE 1 - There may be subtypes and items of select lists that appear in the integrated resources that are not imported into the AIC. Constructs are eliminated from the subtype tree or select list through the use of the implicit interface rules of ISO 10303-11. References to eliminated constructs are outside the scope of the AIC. In some cases, all items of the select list are eliminated. Because AICs are intended to be implemented in the context of an application protocol, the items of the select list will be defined by the scope of the application protocol.

This application interpreted construct provides a consistent set of geometric and topological entities for the definition of non-manifold surface representations that consist of elementary or sculptured curves and surfaces. The highest level entity of this part of ISO 10303 is **non_manifold_surface_shape_-representation**. A **non_manifold_surface_shape_representation** is bounded. The bounding of the geometry is achieved by topological entities, such as **vertex**, **edge**, and **face**.

Topological entities shall not exist without an association to a corresponding geometric domain.

NOTE 2 - This part of ISO 10303 uses all the entities and types from ISO 10303-511, **aic_topologically_bounded_surface**.

EXPRESS specification:

```
*)
SCHEMA aic_non_manifold_surface;

  USE FROM aic_topologically_bounded_surface;       -- ISO 10303-511

  USE FROM geometric_model_schema (                 -- ISO 10303-42
      face_based_surface_model);

  USE FROM geometry_schema (                        -- ISO 10303-42
      b_spline_curve,
      b_spline_surface,
      bounded_pcurve,
      bounded_surface_curve,
      cartesian_transformation_operator_3d,
      curve,
      curve_replica,
      degenerate_pcurve,
      evaluated_degenerate_pcurve,
      intersection_curve,
      offset_curve_3d,
      offset_surface,
      point_on_curve,
      point_on_surface,
      seam_curve,
      surface,
      surface_replica);

  USE FROM product_property_representation_schema ( -- ISO 10303-41
      shape_representation);

  USE FROM representation_schema (                  -- ISO 10303-43
      mapped_item,
      representation,
      representation_item,
      representation_map);

  USE FROM topology_schema (                        -- ISO 10303-42
      closed_shell,
      connected_face_set,
      face,
      open_shell,
      oriented_face);
```

```
(*
```

NOTE 3 - The schemas referenced above can be found in the following parts of ISO 10303:

`aic_topologically_bounded_surface`	ISO 10303-511
`geometric_model_schema`	ISO 10303-42
`geometry_schema`	ISO 10303-42
`product_property_representation_schema`	ISO 10303-41
`representation_schema`	ISO 10303-43
`topology_schema`	ISO 10303-42

4.1 Fundamental concepts and assumptions

The following entities are intended to be independently instantiated in the application protocol schemas that use this AIC:

– non_manifold_surface_shape_representation.

4.2 aic_non_manifold_surface schema entity definition: non_manifold_surface_shape_representation

A **non_manifold_surface_representation** describes the shape or portions of the shape of a **product** using non-manifolds with boundaries.

NOTE 1 - Non-manifolds are topologically less constrained than manifolds. For example, the restriction that only a maximum of two **face**s may share the same **edge** does not apply. They are, therefore, not necessarily suitable for building solid models. They are, however, often found in applications that support the finite element analysis method.

NOTE 2 - A **non_manifold_surface_representation** may well hold a manifold surface model, but not vice versa.

NOTE 3 - Entity **product** is not included in this part of ISO 10303.

A **non_manifold_surface_shape_representation** is a **shape_representation** as defined in ISO 10303-41 that consists of one or many **face_based_surface_model**s. Each **face_based_surface_model** is built up of **connected_face_set**s, which in the context of this part of ISO 10303 may be instantiated as **connected_face_set** or as one of its subtypes **open_shell** or **closed_shell**. **Connected_face_set**s consist of **face**s which use **edge**s and **vertex**s; the latter three shall reference geometric entities, such as **point**s, **curve**s, and **surface**s. The link between topology and geometry may be established by either using **face_surface** or **advanced_face** as defined in ISO 10303-511. The two options differ both in the selection of valid **point**, **curve**, and **surface** subtypes and in constraints on references to underlying geometry.

NOTE 4 - The representation of **face_surface**s as **advanced_face**s is recommended for non-manifold surface models that are intended to be used together with boundary representation solid models; the integration of such a non-manifold surface model into for example an advanced boundary representation model, which is defined in ISO 10303-514, will be easier.

All geometric entities shall be of dimensionality three except for two-dimensional geometry that is used for the purpose of defining **pcurve**s. The use of one-dimensional **cartesian_point**s is excluded.

All unbounded geometry shall be trimmed by using topological constructs.

The **items** of a **non_manifold_surface_shape_representation** may also be of type **mapped_item**, which is defined in ISO 10303-43, or **axis2_placement_3d**. These are used to assemble one or several **non_manifold_surface_shape_representation**s into one other **non_manifold_surface_shape_representation**.

The WHERE-rules of this entity restrict the use of the entity data types that are imported from ISO 10303-42 and ISO 10303-43 according to the statements above. Some of these validations of entity type and constraints are specified in the following two functions:

- **nmsf_curve_check**;
- **nmsf_surface_check**.

In WR5, WR6, and WR10 below, these functions validate the **curve**s and **surface**s of all **edge**s and **face**s that are in the scope of a **non_manifold_surface_shape_representation** except for those that are in the reference tree of an **advanced_face**; the geometry of an **advanced_face** is validated by a different set of rules. The functions automatically assess all underlying geometry; for this they are called recursively.

EXAMPLE A **pcurve** may reference both a **curve** and a **surface**. Function **nmsf_curve_check** validates not only the **pcurve**, but also this underlying geometry. It will, therefore, not only call itself, but also **nmsf_surface_check**.

NOTE 5 - This part of ISO 10303 does not include a function for the validation of **point**s and their underlying **curve**s and **surface**s. This is because all **curve**s and **surface**s of a **non_manifold_surface_shape_representation** are referenced from **edge**s and **face**s and are, thus, covered by the two existing functions already.

NOTE 6 - An application protocol that uses this part of ISO 10303 should explicitly permit that the **shape_representation** entity may be instantiated as a **non_manifold_surface_shape_representation**.

EXPRESS specification:

```
*)
ENTITY non_manifold_surface_shape_representation
   SUBTYPE OF (shape_representation);
WHERE
  WR1: SIZEOF (QUERY (it <* SELF.items |
       NOT (SIZEOF (['AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL',
       'AIC_NON_MANIFOLD_SURFACE.MAPPED_ITEM',
       'AIC_NON_MANIFOLD_SURFACE.AXIS2_PLACEMENT_3D'] * TYPEOF (it)) = 1)))
       = 0;
  WR2: SIZEOF (QUERY (it <* SELF.items |
       SIZEOF ([’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’,
       ’AIC_NON_MANIFOLD_SURFACE.MAPPED_ITEM’] * TYPEOF (it)) = 1)) > 0;
  WR3: SIZEOF (QUERY (mi <* QUERY (it <* SELF.items |
       ’AIC_NON_MANIFOLD_SURFACE.MAPPED_ITEM’ IN TYPEOF (it)) |
       NOT ((’AIC_NON_MANIFOLD_SURFACE.’+
       ’NON_MANIFOLD_SURFACE_SHAPE_REPRESENTATION’
       IN TYPEOF (mi\mapped_item.mapping_source.mapped_representation))
       AND
       (SIZEOF(QUERY (mr_it <*
       mi\mapped_item.mapping_source.mapped_representation.items |
       (’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’
       IN TYPEOF (mr_it)))) > 0 )))) = 0;
  WR4: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items |
       ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
       NOT (SIZEOF (QUERY (cfs <*
       fbsm\face_based_surface_model.fbsm_faces |
       NOT (SIZEOF (QUERY (fa <* cfs.cfs_faces |
       NOT (SIZEOF ([’AIC_NON_MANIFOLD_SURFACE.FACE_SURFACE’,
       ’AIC_NON_MANIFOLD_SURFACE.ORIENTED_FACE’] * TYPEOF (fa)) = 1)))
       = 0))) = 0))) = 0;
  WR5: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items |
       ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
       NOT (SIZEOF (QUERY (cfs <*
       fbsm\face_based_surface_model.fbsm_faces |
       NOT (SIZEOF (QUERY (f_sf <* QUERY (fa <* cfs.cfs_faces |
       (’AIC_NON_MANIFOLD_SURFACE.FACE_SURFACE’ IN TYPEOF (fa))) |
       NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF (f_sf))
       OR
       (nmsf_surface_check(f_sf\face_surface.face_geometry))))) = 0)))
       = 0))) = 0;
  WR6: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items |
       ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
       NOT (SIZEOF (QUERY (cfs <*
       fbsm\face_based_surface_model.fbsm_faces |
       NOT (SIZEOF (QUERY (o_fa <* QUERY (fa <* cfs.cfs_faces |
       (’AIC_NON_MANIFOLD_SURFACE.ORIENTED_FACE’ IN TYPEOF (fa))) |
       NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF
       (o_fa\oriented_face.face_element))
       OR
       (nmsf_surface_check
       (o_fa\oriented_face.face_element\face_surface.face_geometry)))))
       = 0))) = 0))) = 0;
  WR7: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items |
       ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
       NOT (SIZEOF (QUERY (cfs <*
       fbsm\face_based_surface_model.fbsm_faces |
       NOT (SIZEOF (QUERY (fa <* cfs.cfs_faces |
       NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF (fa))
       OR
       (SIZEOF (QUERY (bnds <* fa.bounds |
       NOT (SIZEOF ([’AIC_NON_MANIFOLD_SURFACE.EDGE_LOOP’,
       ’AIC_NON_MANIFOLD_SURFACE.VERTEX_LOOP’]
```

```
      * TYPEOF (bnds.bound)) = 1))) = 0)))) = 0))) = 0))) = 0;
 WR8: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items|
      ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
      NOT (SIZEOF (QUERY (cfs <*
      fbsm\face_based_surface_model.fbsm_faces |
      NOT (SIZEOF (QUERY (fa <* cfs.cfs_faces |
      NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF (fa))
      OR
      (SIZEOF (QUERY (elp_fbnds <* QUERY (bnds <* fa.bounds |
      ’AIC_NON_MANIFOLD_SURFACE.EDGE_LOOP’ IN TYPEOF (bnds.bound)) |
      NOT (SIZEOF (QUERY (oe <* elp_fbnds\path.edge_list |
      NOT (’AIC_NON_MANIFOLD_SURFACE.EDGE_CURVE’ IN TYPEOF
      (oe.edge_element)))) = 0))) = 0)))) = 0))) = 0))) = 0;
 WR9: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items |
      ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
      NOT (SIZEOF (QUERY (cfs <*
      fbsm\face_based_surface_model.fbsm_faces |
      NOT (SIZEOF (QUERY (fa <* cfs.cfs_faces |
      NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF (fa))
      OR
      (SIZEOF (QUERY (elp_fbnds <* QUERY (bnds <* fa.bounds |
      ’AIC_NON_MANIFOLD_SURFACE.EDGE_LOOP’ IN TYPEOF (bnds.bound)) |
      NOT (SIZEOF (QUERY (oe_cv <* QUERY (oe <*
      elp_fbnds\path.edge_list |
      ’AIC_NON_MANIFOLD_SURFACE.EDGE_CURVE’ IN TYPEOF (oe.edge_element)) |
      NOT (SIZEOF ([’AIC_NON_MANIFOLD_SURFACE.B_SPLINE_CURVE’,
      ’AIC_NON_MANIFOLD_SURFACE.CONIC’,
      ’AIC_NON_MANIFOLD_SURFACE.CURVE_REPLICA’,
      ’AIC_NON_MANIFOLD_SURFACE.LINE’,
      ’AIC_NON_MANIFOLD_SURFACE.OFFSET_CURVE_3D’,
      ’AIC_NON_MANIFOLD_SURFACE.PCURVE’,
      ’AIC_NON_MANIFOLD_SURFACE.POLYLINE’,
      ’AIC_NON_MANIFOLD_SURFACE.SURFACE_CURVE’] *
      TYPEOF (oe_cv.edge_element\edge_curve.edge_geometry))
      = 1))) = 0))) = 0)))) = 0))) = 0))) = 0;
WR10: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items |
      ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
      NOT (SIZEOF (QUERY (cfs <*
      fbsm\face_based_surface_model.fbsm_faces |
      NOT (SIZEOF (QUERY (fa <* cfs.cfs_faces |
      NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF (fa))
      OR
      (SIZEOF (QUERY (elp_fbnds <* QUERY (bnds <* fa.bounds |
      ’AIC_NON_MANIFOLD_SURFACE.EDGE_LOOP’ IN TYPEOF (bnds.bound)) |
      NOT (SIZEOF (QUERY (oe <* elp_fbnds\path.edge_list |
      NOT (nmsf_curve_check (oe.edge_element\edge_curve.edge_geometry))))
      = 0))) = 0)))) = 0))) = 0))) = 0;
WR11: SIZEOF (QUERY(fbsm <* QUERY (it <* SELF.items |
      ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
      NOT (SIZEOF (QUERY (cfs <*
      fbsm\face_based_surface_model.fbsm_faces |
      NOT (SIZEOF (QUERY (fa <* cfs.cfs_faces |
      NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF (fa))
      OR
      (SIZEOF (QUERY (elp_fbnds <* QUERY (bnds <* fa.bounds |
      ’AIC_NON_MANIFOLD_SURFACE.EDGE_LOOP’ IN TYPEOF (bnds.bound)) |
      NOT (SIZEOF (QUERY (oe <* elp_fbnds\path.edge_list|
      NOT ((’AIC_NON_MANIFOLD_SURFACE.VERTEX_POINT’ IN TYPEOF
      (oe.edge_element.edge_start))
      AND
      (’AIC_NON_MANIFOLD_SURFACE.VERTEX_POINT’ IN
      TYPEOF (oe.edge_element.edge_end)))))
      = 0))) = 0)))) = 0))) = 0))) = 0;
WR12: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items |
      ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
      NOT (SIZEOF (QUERY (cfs <*
      fbsm\face_based_surface_model.fbsm_faces |
      NOT (SIZEOF (QUERY (fa <* cfs.cfs_faces |
      NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF (fa))
      OR
      (SIZEOF (QUERY (elp_fbnds <* QUERY (bnds <* fa.bounds |
      ’AIC_NON_MANIFOLD_SURFACE.EDGE_LOOP’ IN TYPEOF (bnds.bound)) |
      NOT (SIZEOF (QUERY (oe <* elp_fbnds\path.edge_list |
      NOT ((SIZEOF ([’AIC_NON_MANIFOLD_SURFACE.CARTESIAN_POINT’,
      ’AIC_NON_MANIFOLD_SURFACE.DEGENERATE_PCURVE’,
      ’AIC_NON_MANIFOLD_SURFACE.POINT_ON_CURVE’,
      ’AIC_NON_MANIFOLD_SURFACE.POINT_ON_SURFACE’] * TYPEOF
      (oe.edge_element.edge_start\vertex_point.vertex_geometry)) = 1)
      AND
      (SIZEOF ([’AIC_NON_MANIFOLD_SURFACE.CARTESIAN_POINT’,
      ’AIC_NON_MANIFOLD_SURFACE.DEGENERATE_PCURVE’,
      ’AIC_NON_MANIFOLD_SURFACE.POINT_ON_CURVE’,
      ’AIC_NON_MANIFOLD_SURFACE.POINT_ON_SURFACE’] * TYPEOF
      (oe.edge_element.edge_end\vertex_point.vertex_geometry)) = 1
      )))) = 0))) = 0)))) = 0))) = 0))) = 0;
WR13: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items |
      ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
      NOT (SIZEOF (QUERY (cfs <*
      fbsm\face_based_surface_model.fbsm_faces |
      NOT (SIZEOF (QUERY (fa <* cfs.cfs_faces |
      NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF (fa))
      OR
      (SIZEOF (QUERY (vlp_fbnds <* QUERY (bnds <* fa.bounds |
      ’AIC_NON_MANIFOLD_SURFACE.VERTEX_LOOP’ IN TYPEOF (bnds.bound)) |
      NOT (’AIC_NON_MANIFOLD_SURFACE.VERTEX_POINT’ IN TYPEOF
      (vlp_fbnds\vertex_loop.loop_vertex)))) = 0)))) = 0)))
      = 0))) = 0;
WR14: SIZEOF (QUERY (fbsm <* QUERY (it <* SELF.items |
      ’AIC_NON_MANIFOLD_SURFACE.FACE_BASED_SURFACE_MODEL’ IN TYPEOF (it)) |
      NOT (SIZEOF (QUERY (cfs <*
      fbsm\face_based_surface_model.fbsm_faces |
      NOT (SIZEOF (QUERY (fa <* cfs.cfs_faces |
      NOT ((’AIC_NON_MANIFOLD_SURFACE.ADVANCED_FACE’ IN TYPEOF (fa))
      OR
```

```
        (SIZEOF (QUERY (vlp_fbnds <* QUERY (bnds <* fa.bounds |
        'AIC_NON_MANIFOLD_SURFACE.VERTEX_LOOP' IN TYPEOF (bnds.bound)) |
        NOT (SIZEOF (['AIC_NON_MANIFOLD_SURFACE.CARTESIAN_POINT',
        'AIC_NON_MANIFOLD_SURFACE.DEGENERATE_PCURVE',
        'AIC_NON_MANIFOLD_SURFACE.POINT_ON_CURVE',
        'AIC_NON_MANIFOLD_SURFACE.POINT_ON_SURFACE'] * TYPEOF
        (vlp_fbnds\vertex_loop.loop_vertex\vertex_point.vertex_geometry))
        = 1))) = 0)))) = 0))) = 0))) = 0;
END_ENTITY;
(*
```

Formal propositions:

WR1: The **items** in a **non_manifold_surface_shape_representation** shall be **face_based_surface_models**, **mapped_items**, or **axis2_placement_3ds**.

NOTE 7 - **Axis2_placement_3d** is a valid **mapped_item.mapping_target**. To include another **representation** into the list of **items** of a **non_manifold_surface_shape_representation** (see WR3 for valid **mapped_items**), the **mapped_item.mapping_source.mapping_origin** may be any entity that is geometrically founded in the **geometric_representation_context** of the **mapped_representation**. If this entity is an **axis2_placement_3d**, the operator that maps the **mapped_representation** into the **non_manifold_surface_shape_representation** corresponds to a transformation matrix with only translation and rotation enabled. If a **cartesian_transformation_operator_3d** is used as **mapping_origin**, scaling and mirroring are possible.

WR2: At least one of the **items** in a **non_manifold_surface_shape_representation** shall be either a **face_based_surface_model** or a **mapped_item**.

WR3: If there is a **mapped_item** in a **non_manifold_surface_shape_representation**, the **mapped_representation** of its **mapping_source** shall be a **non_manifold_surface_shape_representation**. This **shape_representation** shall include at least one **face_based_surface_model**.

WR4: A **face** shall be instantiated as either a **face_surface**, its subtype **advanced_face**, or an **oriented_face**.

WR5: All basis geometry that via a **face_surface** is referenced by **surfaces** shall be either within the reference tree of an **advanced_face** or shall be valid **curves** and **surfaces**.

The **basis_surface** of an **offset_surface** shall be either an **elementary_surface**, **b_spline_surface**, **offset_surface**, **swept_surface**, or **surface_replica**.

The **parent_surface** of a **surface_replica** shall be either an **elementary_surface**, **b_spline_surface**, **offset_surface**, **swept_surface**, or **surface_replica**.

The **swept_curve** of a **swept_surface** shall be either a **line**, **conic**, **pcurve**, **surface_curve**, **offset_curve_3d**, **b_spline_curve**, **polyline**, or **curve_replica**.

The attribute **self_intersect** shall be set to FALSE or UNKNOWN for **b_spline_surfaces** and **offset_surfaces**.

NOTE 8 - A **surface** is validated against these constraints by the function **nmsf_surface_check**.

WR6: All basis geometry that is referenced via an **oriented_face.face_element** shall be either within the reference tree of an **advanced_face** or shall be valid **curves** and **surfaces**.

The **basis_surface** of an **offset_surface** shall be either an **elementary_surface**, **b_spline_surface**, **offset_surface**, **swept_surface**, or **surface_replica**.

The **parent_surface** of a **surface_replica** shall be either an **elementary_surface**, **b_spline_surface**, **offset_surface**, **swept_surface**, or **surface_replica**.

The **swept_curve** of a **swept_surface** shall be either a **line**, **conic**, **pcurve**, **surface_curve**, **offset_curve_3d**, **b_spline_curve**, **polyline**, or **curve_replica**.

The attribute **self_intersect** shall be set to FALSE or UNKNOWN for **b_spline_surface**s and **offset_surface**s.

NOTE 9 - A **surface** is validated against these constraints by the function **nmsf_surface_check**.

WR7: The **bound** of a **face_bound** that is referenced by a **face** that is referenced either directly by a **connected_face_set** or via an **oriented_face** shall be either within the reference tree of an **advanced_face** or shall be an **edge_loop** or a **vertex_loop**.

WR8: The geometry of a bounding **edge** that is the **edge_element** of an **oriented_edge** that is referenced by a **face** that is referenced either directly by a **connected_face_set** or via an **oriented_face** shall be either within the reference tree of an **advanced_face** or shall be an **edge_curve**.

WR9: The **curve** used to define the geometry of an **edge** that is the **edge_geometry** of an **edge_curve** of an **edge** that is referenced by a **face** that is referenced either directly by a **connected_face_set** or via an **oriented_face** shall be either a **b_spline_curve**, a **conic**, a **curve_replica**, a **line**, an **offset_curve_3d**, a **pcurve**, a **polyline**, a **surface_curve**, or a **curve** in the reference tree of an **advanced_face**.

WR10: All basis geometry that is referenced by **curve**s shall be either within the reference tree of an **advanced_face** or shall be valid **curve**s and **surface**s.

The **parent_curve** of a **curve_replica** shall be either a **line**, **conic**, **pcurve**, **surface_curve**, **offset_curve_3d**, **b_spline_curve**, **polyline**, or **curve_replica**.

The **basis_curve** of an **offset_curve_3d** shall be either a **line**, **conic**, **pcurve**, **surface_curve**, **offset_curve_3d**, **b_spline_curve**, or **curve_replica**.

The **curve_3d** of a **surface_curve** shall be either a **line**, **conic**, **offset_curve_3d**, **b_spline_curve**, **polyline**, or **curve_replica**.

The **basis_surface** of a **surface_curve** shall be either a **b_spline_surface**, **elementary_surface**, **offset_surface**, **surface_replica**, or **swept_surface**.

Polylines shall contain at least three **cartesian_point**s.

The attribute **self_intersect** shall be set to FALSE or UNKNOWN for **b_spline_curve**s and **offset_curve_3d**s.

NOTE 10 - A **curve** is validated against these constraints by the function **nmsf_curve_check**.

WR11: The **edge_start** and **edge_end** of an **edge** shall be either within the reference tree of an **advanced_face** or shall be **vertex_point**s.

WR12: The **vertex_geometry** of a **vertex** that is part of an **edge_loop** shall be either within the reference tree of an **advanced_face** or shall be a **cartesian_point**, **point_on_curve**, **point_on_surface**, or **degenerate_pcurve**.

WR13: The **loop_vertex** of a **vertex_loop** shall be either within the reference tree of an **advanced_face** or shall be a **vertex_point**.

WR14: The **vertex_geometry** of a **vertex** that is part of a **vertex_loop** shall be either within the reference tree of an **advanced_face** or shall be a **cartesian_point**, **point_on_curve**, **point_on_surface**, or **degenerate_pcurve**.

Informal propositions:

IP1: The portion of a **b_spline_curve** that is within the topological domain of a **non_manifold_surface_shape_representation** shall not self-intersect.

IP2: The portion of a **b_spline_surface** that is within the topological domain of a **non_manifold_surface_shape_representation** shall not self-intersect.

IP3: The portion of an **offset_curve_3d** that is within the topological domain of a **non_manifold_surface_shape_representation** shall not self-intersect.

IP4: The portion of an **offset_surface** that is within the topological domain of a **non_manifold_surface_shape_representation** shall not self-intersect.

IP5: If a **face** has only one connected outer bound, the corresponding **loop** shall be represented as **face_outer_bound**. If the outer bound is not connected, **face_outer_bound** shall not be used.

4.3 aic_non_manifold_surface function definitions

This section describes functions required to formulate constraints for the aic_non_manifold_surface schema. These functions are used in the specification of the entity **non_manifold_surface_shape_representation**.

4.3.1 nmsf_curve_check

The **nmsf_curve_check** function checks a **curve** instance for validity in the context of a **non_manifold_surface_shape_representation**. All geometry that is referenced by this **curve** instance, such as other **curve**s and **surface**s, are also validated.

EXAMPLE One of the constraints that is validated by this function is whether the self-intersection flag of a **b_spline_curve** instance is set to TRUE, FALSE, or UNKNOWN; only FALSE and UNKNOWN are valid.

Where appropriate an instance is investigated recursively. This means if a **curve** references another **curve** as a basis curve or parent curve, the **nmsf_curve_check** function is called again. If a **surface** is referenced, the **nmsf_surface_check** function is called. The recursive process terminates at entity types that do not reference any **curve**s or **surface**s.

The following curve types and their subtypes are within the scope of the **non_manifold_surface_shape_representation** and are, thus, valid input to this function:

- **b_spline_curve**;
- **conic**;
- **curve_replica**;
- **line**;
- **offset_curve_3d**;
- **pcurve**;
- **polyline**;
- **surface_curve**.

Four of these **curve** types reference basis or parent **curve**s. The lists below indicate the valid references.

NOTE 1 - This function applies to those entity types that are marked in the lists below a recursive process to check their entity references for valid instantiations.

The **parent_curve** of a **curve_replica** shall be of one of the following types:

- **b_spline_curve**;
- **conic**;
- **curve_replica** (recursive);
- **line**;
- **offset_curve_3d** (recursive);
- **pcurve** (recursive);
- **polyline**;
- **surface_curve** (recursive).

The **basis_curve** of an **offset_curve_3d** shall be of one of the following types:

- **b_spline_curve**;
- **conic**;
- **curve_replica** (recursive);

- **line**;
- **offset_curve_3d** (recursive);
- **pcurve** (recursive);
- **surface_curve** (recursive).

The one instance in the set of **items** of a **definitional_representation** that is referenced as **reference_to_curve** by a **pcurve** shall be of one of the following types:

- **b_spline_curve**;
- **conic**;
- **curve_replica** (recursive);
- **line**;
- **polyline**.

The **curve_3d** of a **surface_curve** shall be of one of the following types:

- **b_spline_curve**;
- **conic**;
- **curve_replica** (recursive);
- **line**;
- **offset_curve_3d** (recursive);
- **polyline**;
- **surface_curve** (recursive).

Pcurve and **surface_curve** reference **surface**s. Function **nmsf_surface_check** is called for validation of these **surface**s. The **non_manifold_surface_shape_representation** requires the same constraints on valid **surface** references for **pcurve**s and **surface_curve**s as specified in ISO 10303-42.

A valid **polyline** shall consist of at least three **cartesian_point**s.

The attribute **self_intersect** shall for B-spline and offset geometry be set to FALSE or UNKNOWN.

This function returns TRUE, if the types of all referenced geometries are within the scope of the **non_-manifold_surface_shape_representation** and if all constraints are fulfilled, otherwise the function returns FALSE.

> NOTE 2 - This function does not check the correctness of references with respect to requirements specified by ISO 10303-42. Only additional requirements due to the scope of the **non_manifold_surface_shape_representation** are checked.

EXPRESS specification:

```
*)
FUNCTION nmsf_curve_check (cv : representation_item) : BOOLEAN;

(* This function verifies the validity of a curve in the context of a
   non-manifold surface model. Representation_items are
   valid input, however, they are supposed to be curves; otherwise
   this function will return false.
*)

 (* complex subtypes of curve that are both bounded_curve and
    oneof conic, curve_replica, line, or offset_curve_3d are not
    valid
 *)
 IF SIZEOF ([’AIC_NON_MANIFOLD_SURFACE.BOUNDED_CURVE’,
   ’AIC_NON_MANIFOLD_SURFACE.CONIC’,
   ’AIC_NON_MANIFOLD_SURFACE.CURVE_REPLICA’,
   ’AIC_NON_MANIFOLD_SURFACE.LINE’,
   ’AIC_NON_MANIFOLD_SURFACE.OFFSET_CURVE_3D’] * TYPEOF(cv)) > 1
 THEN RETURN(FALSE);
 ELSE

  (* b_spline_curves shall not self-intersect
   *)
  IF ((’AIC_NON_MANIFOLD_SURFACE.B_SPLINE_CURVE’ IN TYPEOF (cv)) AND
    (cv\b_spline_curve.self_intersect = FALSE) OR
    (cv\b_spline_curve.self_intersect = UNKNOWN))
  THEN RETURN(TRUE);
  ELSE

    (* conics and lines are valid curve types
     *)
    IF SIZEOF ([’AIC_NON_MANIFOLD_SURFACE.CONIC’,
    ’AIC_NON_MANIFOLD_SURFACE.LINE’] * TYPEOF (cv)) = 1 THEN
      RETURN(TRUE);
    ELSE

      (* a curve_replica shall reference a valid curve
       *)
      IF ’AIC_NON_MANIFOLD_SURFACE.CURVE_REPLICA’ IN TYPEOF(cv) THEN
        RETURN (nmsf_curve_check(cv\curve_replica.parent_curve));
```

```
ELSE

  (* an offset_curve_3d shall not self-intersect and
     shall reference a valid curve; a polyline is not a
     valid basis_curve
   *)
  IF ((’AIC_NON_MANIFOLD_SURFACE.OFFSET_CURVE_3D’ IN TYPEOF (cv))
    AND
    ((cv\offset_curve_3d.self_intersect = FALSE) OR
    (cv\offset_curve_3d.self_intersect = UNKNOWN))
    AND
    (NOT (’AIC_NON_MANIFOLD_SURFACE.POLYLINE’ IN TYPEOF
    (cv\offset_curve_3d.basis_curve)))) THEN
    RETURN (nmsf_curve_check(cv\offset_curve_3d.basis_curve));
  ELSE

    (* a pcurve shall reference a valid curve and a valid
       basis_surface
     *)
    IF ’AIC_NON_MANIFOLD_SURFACE.PCURVE’ IN TYPEOF(cv) THEN
      RETURN ((nmsf_curve_check
      (cv\pcurve.reference_to_curve\representation.items[1]))
      AND
      (nmsf_surface_check(cv\pcurve.basis_surface)));
    ELSE

      (* a surface_curve references a curve_3d and one or
         two pcurves or one or two surfaces or one of
         each; all of these references shall be valid
       *)
      IF ’AIC_NON_MANIFOLD_SURFACE.SURFACE_CURVE’ IN TYPEOF(cv) THEN

        (* if the curve reference is correct, check also the rest
         *)
        IF nmsf_curve_check(cv\surface_curve.curve_3d) THEN
          REPEAT i := 1 TO SIZEOF
          (cv\surface_curve.associated_geometry);

            (* do for one or two associated_geometrys:
             *)
            IF ’AIC_NON_MANIFOLD_SURFACE.SURFACE’ IN
              TYPEOF (cv\surface_curve.associated_geometry[i]) THEN
              IF NOT nmsf_surface_check
                (cv\surface_curve.associated_geometry[i]) THEN
                RETURN(FALSE);
              END_IF;
            ELSE
              IF ’AIC_NON_MANIFOLD_SURFACE.PCURVE’ IN TYPEOF
                (cv\surface_curve.associated_geometry[i]) THEN
                IF NOT nmsf_curve_check
                  (cv\surface_curve.associated_geometry[i]) THEN
                  RETURN(FALSE);
```

```
                      END_IF;
                    END_IF;
                  END_IF;
                END_REPEAT;
                RETURN(TRUE);
              END_IF;
            ELSE

              (* a polyline shall have at least 3 points
               *)
              IF 'AIC_NON_MANIFOLD_SURFACE.POLYLINE' IN TYPEOF(cv) THEN
                IF (SIZEOF (cv\polyline.points) >= 3) THEN RETURN (TRUE);
                END_IF;
              END_IF;
            END_IF;
          END_IF;
        END_IF;
      END_IF;
    END_IF;
  END_IF;
 END_IF;
 (* FALSE is returned if the input parameter cv is not a valid curve.
  *)
 RETURN (FALSE);
END_FUNCTION;
(*
```

Argument definitions:

cv: (input) a **representation_item** that is supposed to be a **curve** that is checked for being valid in the context of a **non_manifold_surface_shape_representation**.

BOOLEAN: (output) is TRUE if the **representation_item** is a valid **curve**; else FALSE.

4.3.2 nmsf_surface_check

The **nmsf_surface_check** function checks a **surface** instance for validity in the context of a **non_manifold_surface_shape_representation**. All geometry that is referenced by this **surface** instance, such as other **curve**s and **surface**s, are also validated.

EXAMPLE One of the constraints that is validated by this function is whether the self-intersection flag of a **b_spline_surface** instance is set to TRUE, FALSE, or UNKNOWN; only FALSE and UNKNOWN are valid.

Where appropriate an instance is investigated recursively. This means if a **surface** references another **surface** as a basis surface or parent surface the **nmsf_surface_check** function is called again. If a **curve** is referenced, the **nmsf_curve_check** function is called. The recursive process terminates at entity types that do not reference any **curve**s or **surface**s.

The following surface types and their subtypes are within the scope of the **non_manifold_surface_-shape_representation** and are, thus, valid input to this function:

- **b_spline_surface**;
- **elementary_surface**;
- **offset_surface**;
- **surface_replica**;
- **swept_surface**.

Two of these **surface** types reference basis or parent **surface**s. The **parent_surface** of a **surface_replica** and the **basis_surface** of an **offset_surface** shall both be of one of the following types:

- **b_spline_surface**;
- **elementary_surface**;
- **offset_surface** (recursive);
- **surface_replica** (recursive);
- **swept_surface**.

NOTE 1 - This function applies to those entity types that are marked in the list above a recursive process to check their entity references for valid instantiations.

Swept_surfaces reference **curve**s. Function **nmsf_curve_check** is called for validation of these **curve**s. The **non_manifold_surface_shape_representation** requires the same constraints on valid sweeping curves as specified in ISO 10303-42. All **curve**s that are in the scope of a **non_manifold_surface_-shape_representation** are valid as **swept_surface.swept_curve**s.

The attribute **self_intersect** shall for B-spline and offset geometry be set to FALSE or UNKNOWN.

This function returns TRUE, if the types of all referenced geometries are within the scope of the **non_-manifold_surface_shape_representation** and if all constraints are fulfilled, otherwise the function returns FALSE.

NOTE 2 - This function does not check the correctness of references with respect to requirements specified by ISO 10303-42. Only additional requirements due to the scope of the **non_manifold_surface_shape_representation** are checked.

EXPRESS specification:

```
*)
FUNCTION nmsf_surface_check (surf : surface) : BOOLEAN;

(* This function verifies the validity of a surface in the
   context of a non-manifold surface model.
*)

  (* elementary_surfaces are valid surface types
   *)
  IF 'AIC_NON_MANIFOLD_SURFACE.ELEMENTARY_SURFACE' IN TYPEOF(surf) THEN
    RETURN(TRUE);
  ELSE

    (* a swept_surface shall have a valid sweeping curve
     *)
    IF 'AIC_NON_MANIFOLD_SURFACE.SWEPT_SURFACE' IN TYPEOF (surf) THEN
      RETURN (nmsf_curve_check(surf\swept_surface.swept_curve));
    ELSE

      (* an offset_surface shall not self-intersect and shall
         reference a valid surface
       *)
      IF (('AIC_NON_MANIFOLD_SURFACE.OFFSET_SURFACE' IN TYPEOF (surf)) AND
        (surf\offset_surface.self_intersect = FALSE) OR
        (surf\offset_surface.self_intersect = UNKNOWN)) THEN
        RETURN (nmsf_surface_check(surf\offset_surface.basis_surface));
      ELSE

        (* a surface_replica shall have a valid parent surface
         *)
        IF 'AIC_NON_MANIFOLD_SURFACE.SURFACE_REPLICA' IN TYPEOF(surf) THEN
          RETURN(nmsf_surface_check(surf\surface_replica.parent_surface));
        ELSE

          (* a b_spline_surface shall not self-intersect
           *)
          IF (('AIC_NON_MANIFOLD_SURFACE.B_SPLINE_SURFACE' IN TYPEOF(surf))
            AND
            (surf\b_spline_surface.self_intersect = FALSE) OR
            (surf\b_spline_surface.self_intersect = UNKNOWN)) THEN
            RETURN(TRUE);
          END_IF;
        END_IF;
      END_IF;
    END_IF;
  END_IF;
  RETURN(FALSE);
END_FUNCTION;
(*
```

Argument definitions:

surf: (input) the **surface** that is being checked for a valid **surface** in a **non_manifold_surface_shape_representation**.

BOOLEAN: (output) is TRUE if the **surface** is a valid **surface**; else FALSE.

```
*)
END_SCHEMA; -- aic_non_manifold_surface
(*
```

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 the short names are found in the implementation methods included in ISO 10303.

Table A.1 – Short names of entities

Entity name	Short name
NON_MANIFOLD_SURFACE_SHAPE_REPRESENTATION	NMSSR

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(508) 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 aic-non-manifold-surface in an open information system, the object identifier

{ iso standard 10303 part(508) version(1) object(1) aic-non-manifold-surface(1) }

is assigned to the **aic_non_manifold_surface** 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)

EXPRESS-G diagrams

Figures C.1 through C.22 correspond to the EXPRESS generated from the short listing given in clause 4 using the interface specifications of ISO 10303-11. The diagrams use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined in annex D of ISO 10303-11.

NOTE 1 - The following select types are interfaced into the AIC expanded listing according to the implicit interface rules of ISO 10303-11. These select types are not used by other entities in this part of ISO 10303.

— curve_on_surface;

— founded_item_select;

— geometric_set_select;

— measure_value;

— reversible_topology;

— reversible_topology_item;

— shell;

— surface_model;

— transformation;

— trimming_select;

— vector_or_direction.

NOTE 2 - The following entity types are interfaced into the AIC expanded listing according to the implicit interface rules of ISO 10303-11. These entity types are not used by other entities in this part of ISO 10303 and are not intended to be instantiated independently.

— composite_curve;

— composite_curve_on_surface;

— composite_curve_segment;

— oriented_closed_shell;

— oriented_open_shell.

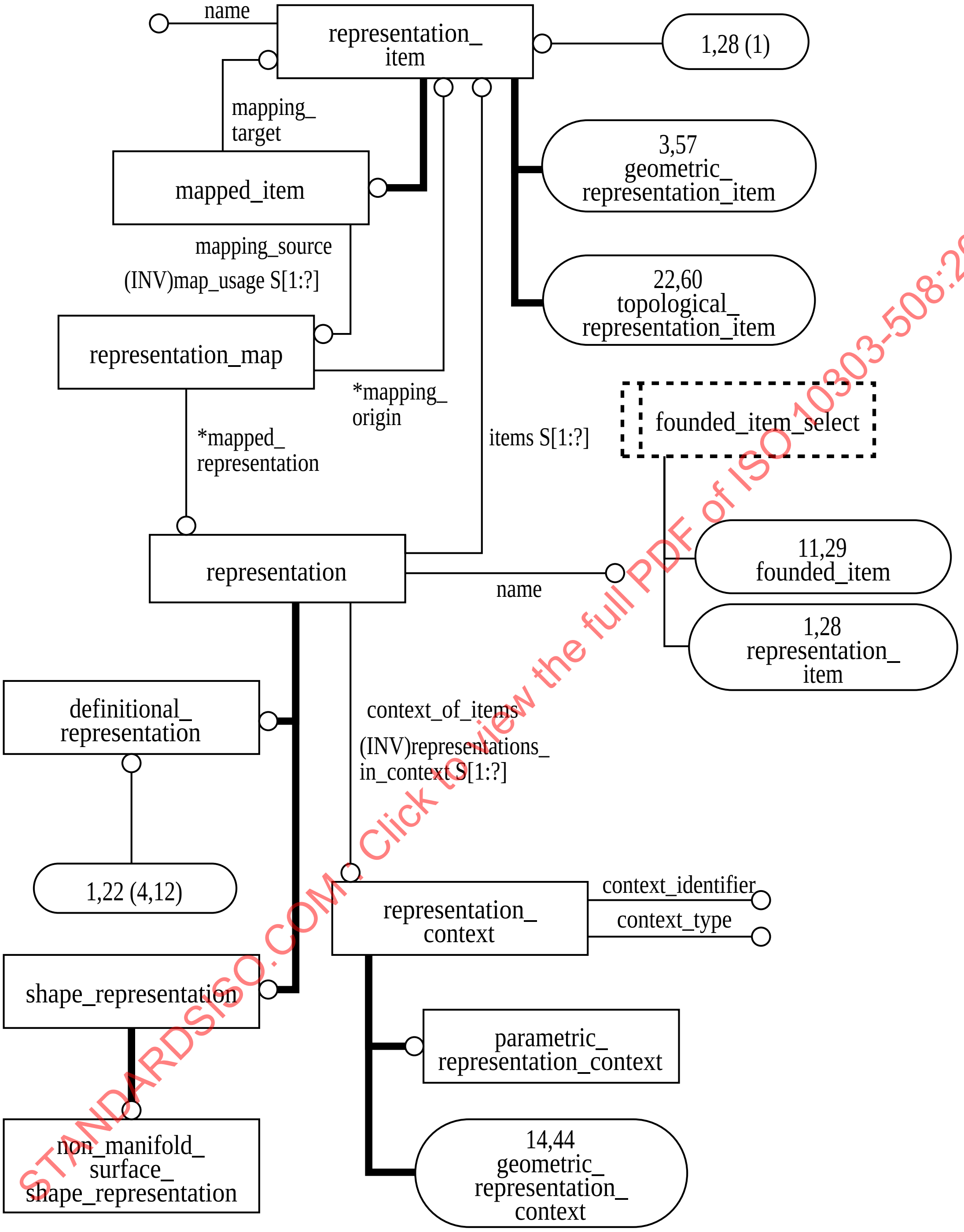

Figure C.1 – EXPRESS-G diagram 1 of 22

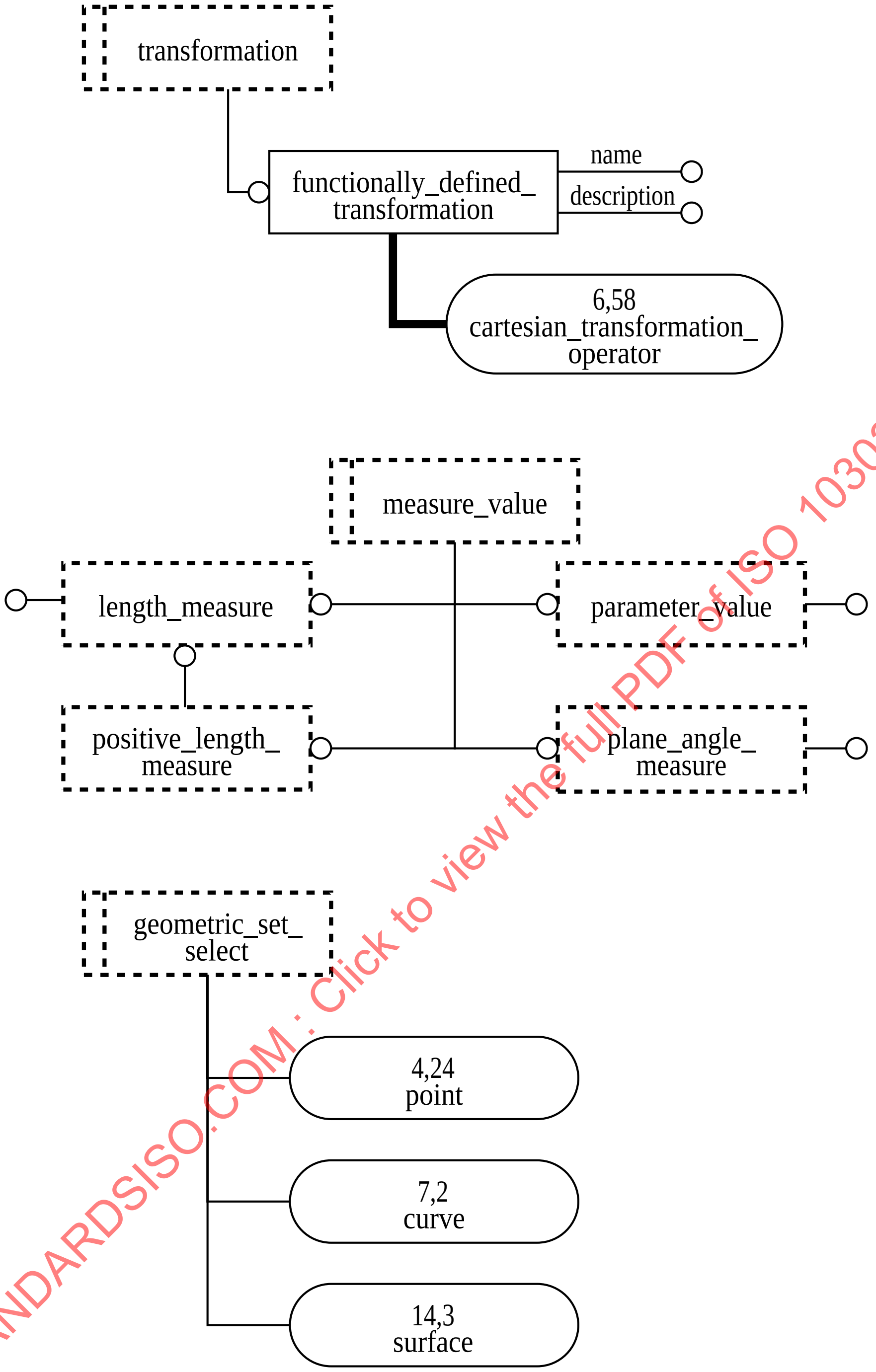

Figure C.2 – EXPRESS-G diagram 2 of 22

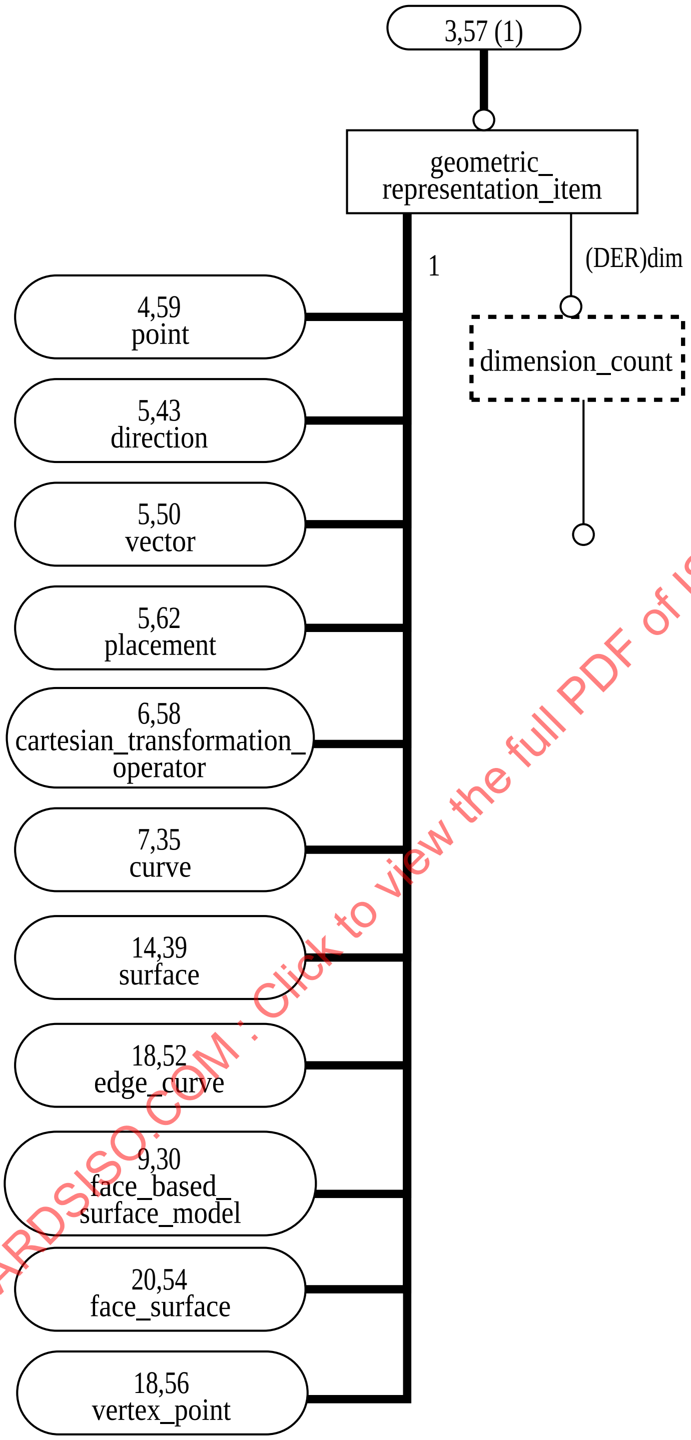

Figure C.3 – EXPRESS-G diagram 3 of 22

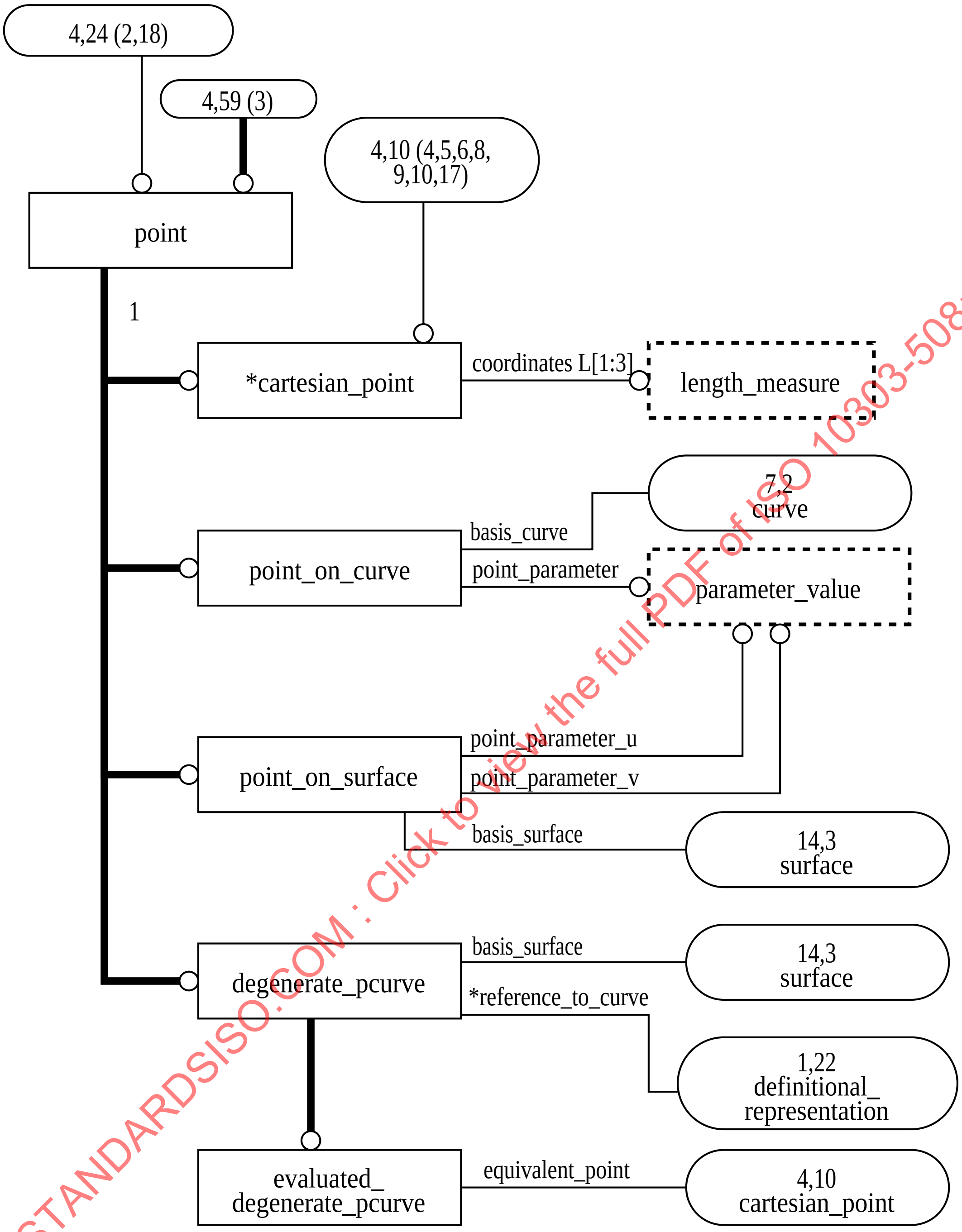

Figure C.4 – EXPRESS-G diagram 4 of 22

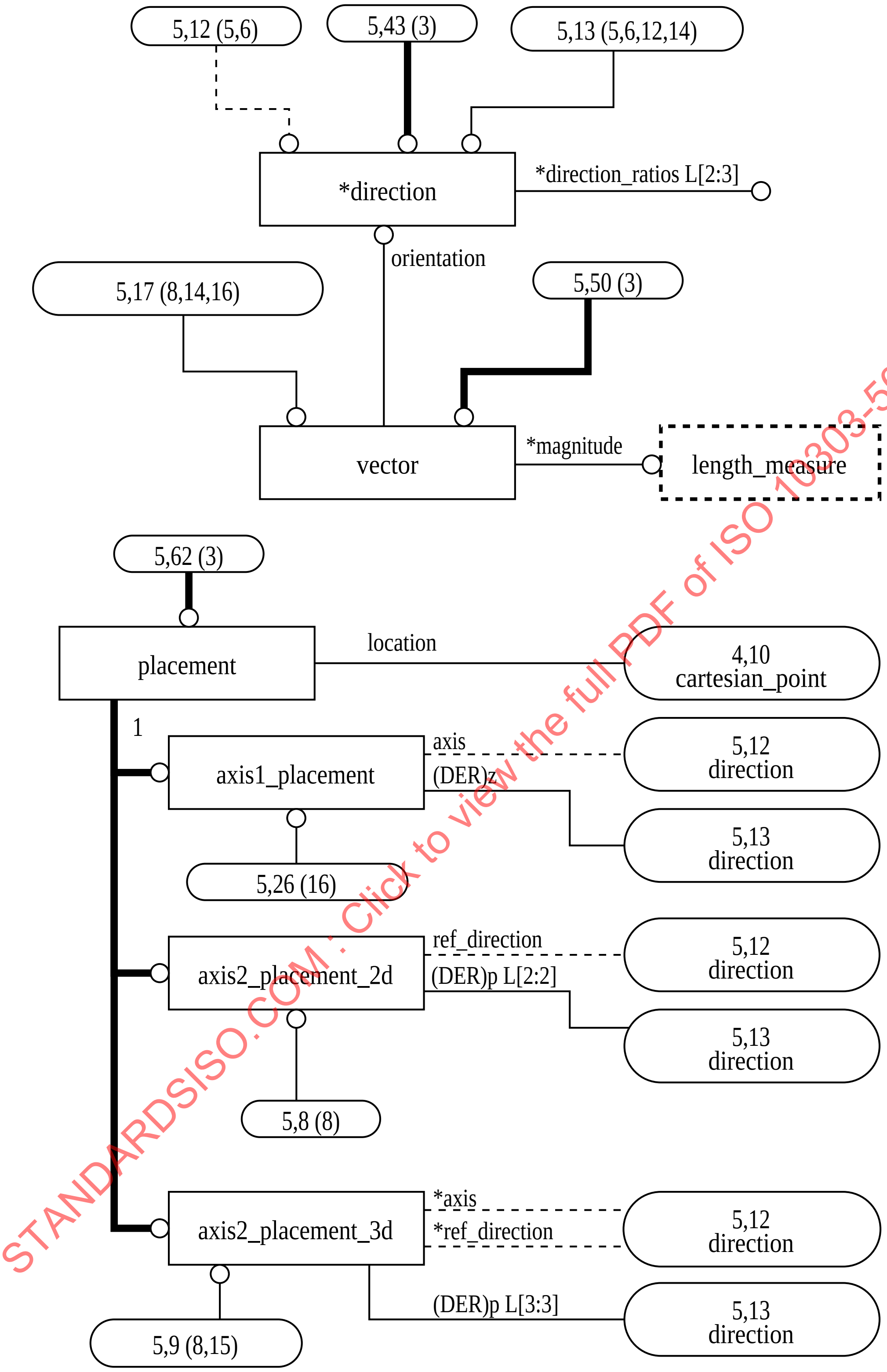

Figure C.5 – EXPRESS-G diagram 5 of 22

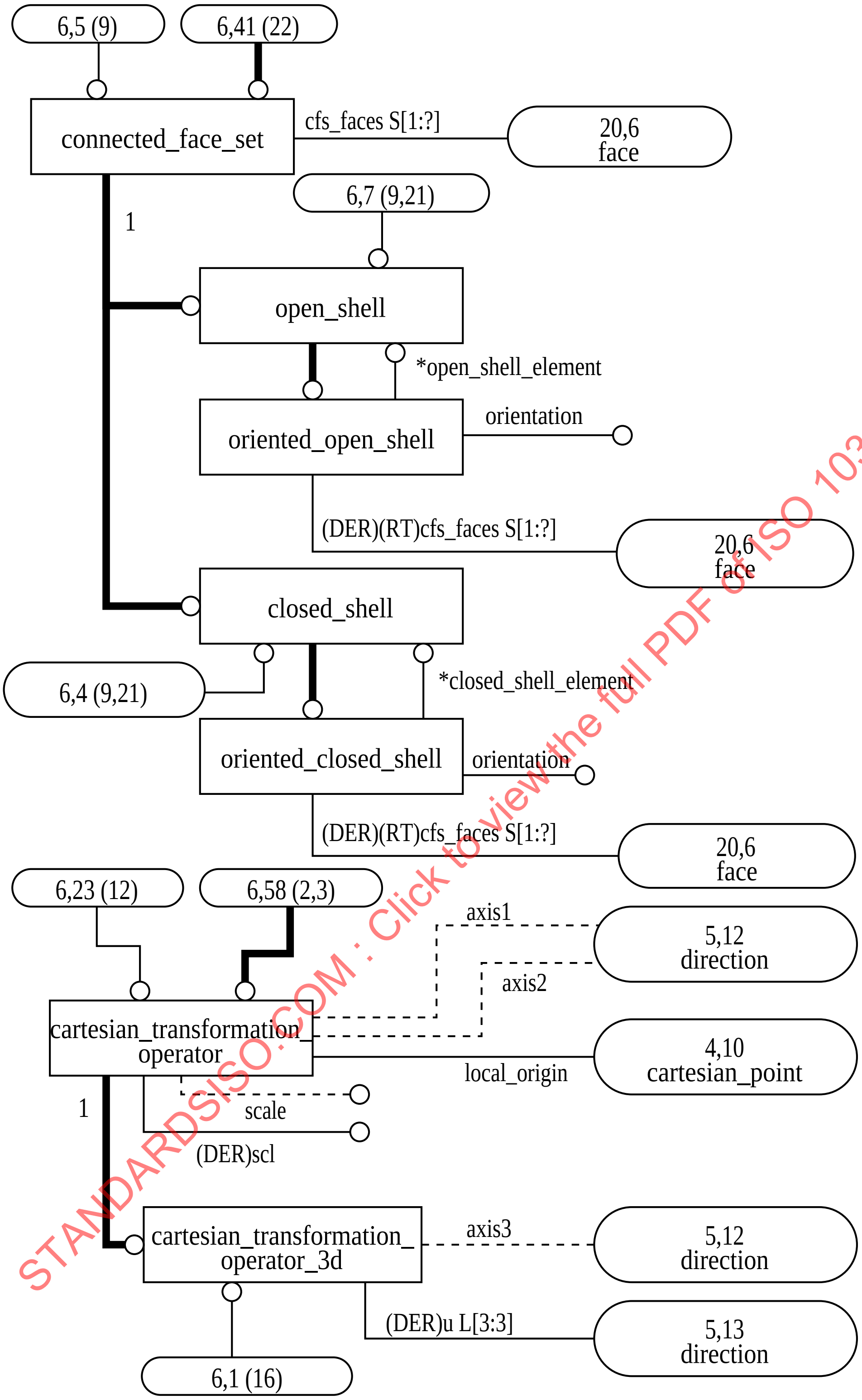

Figure C.6 – EXPRESS-G diagram 6 of 22

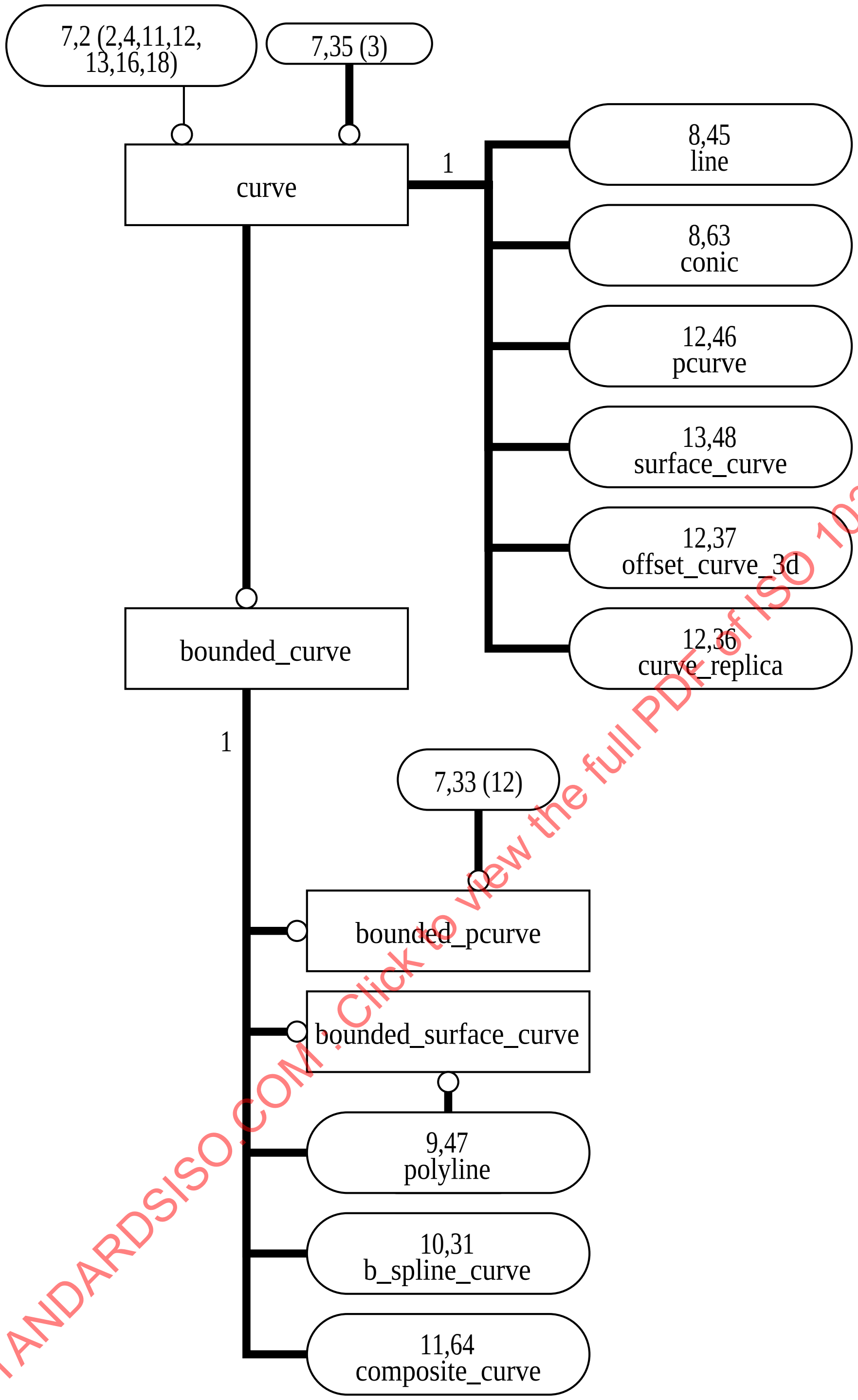

Figure C.7 – EXPRESS-G diagram 7 of 22

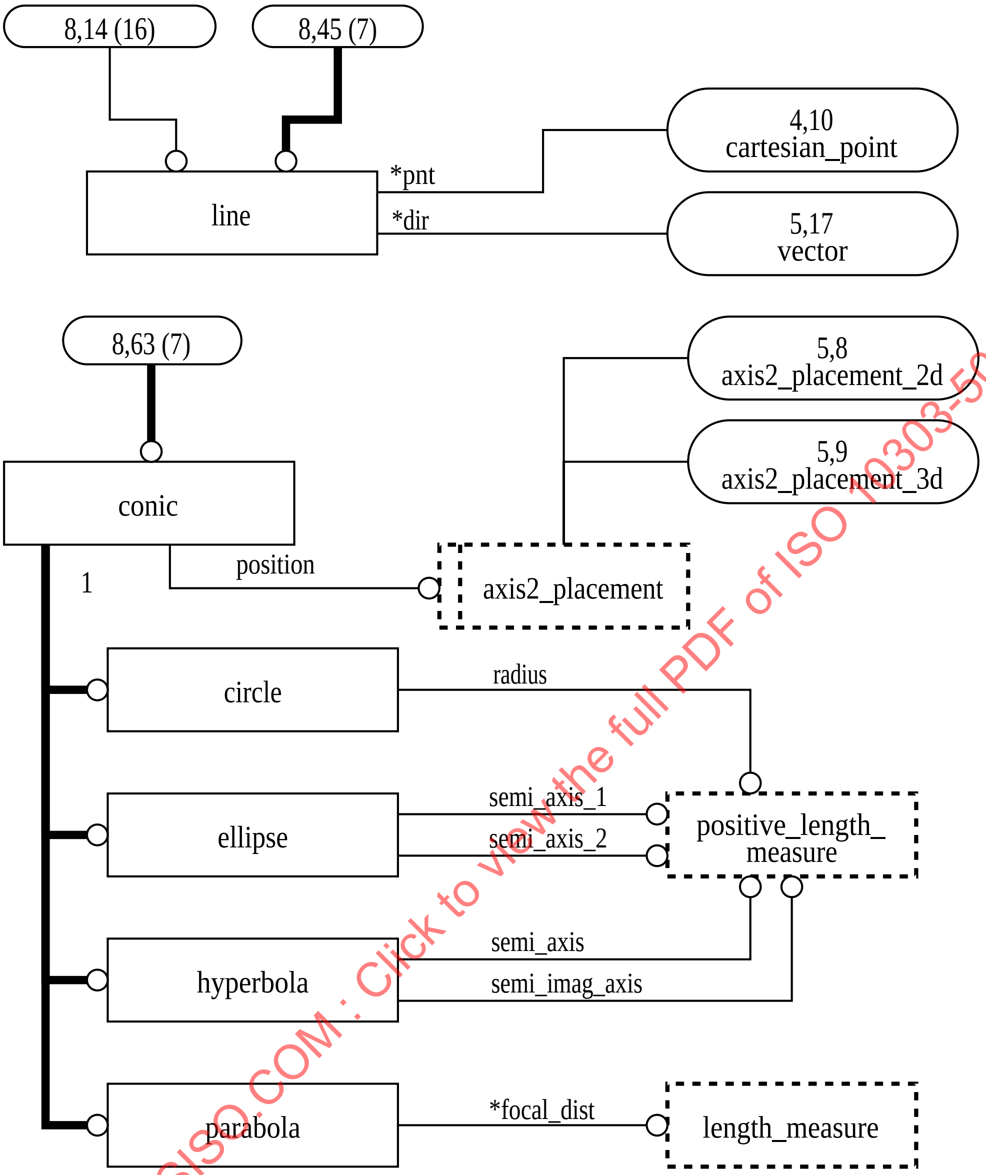

Figure C.8 – EXPRESS-G diagram 8 of 22

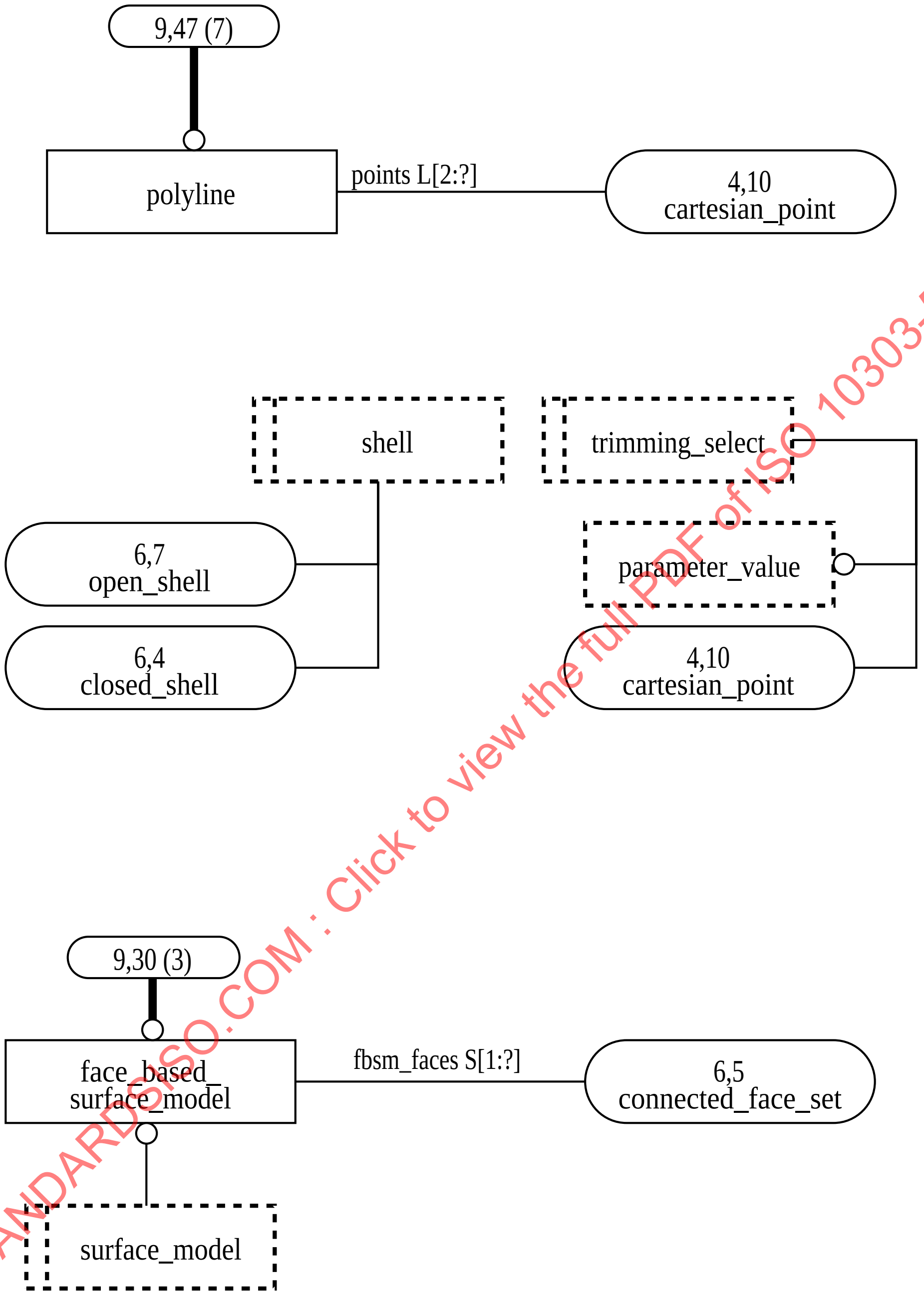

Figure C.9 – EXPRESS-G diagram 9 of 22

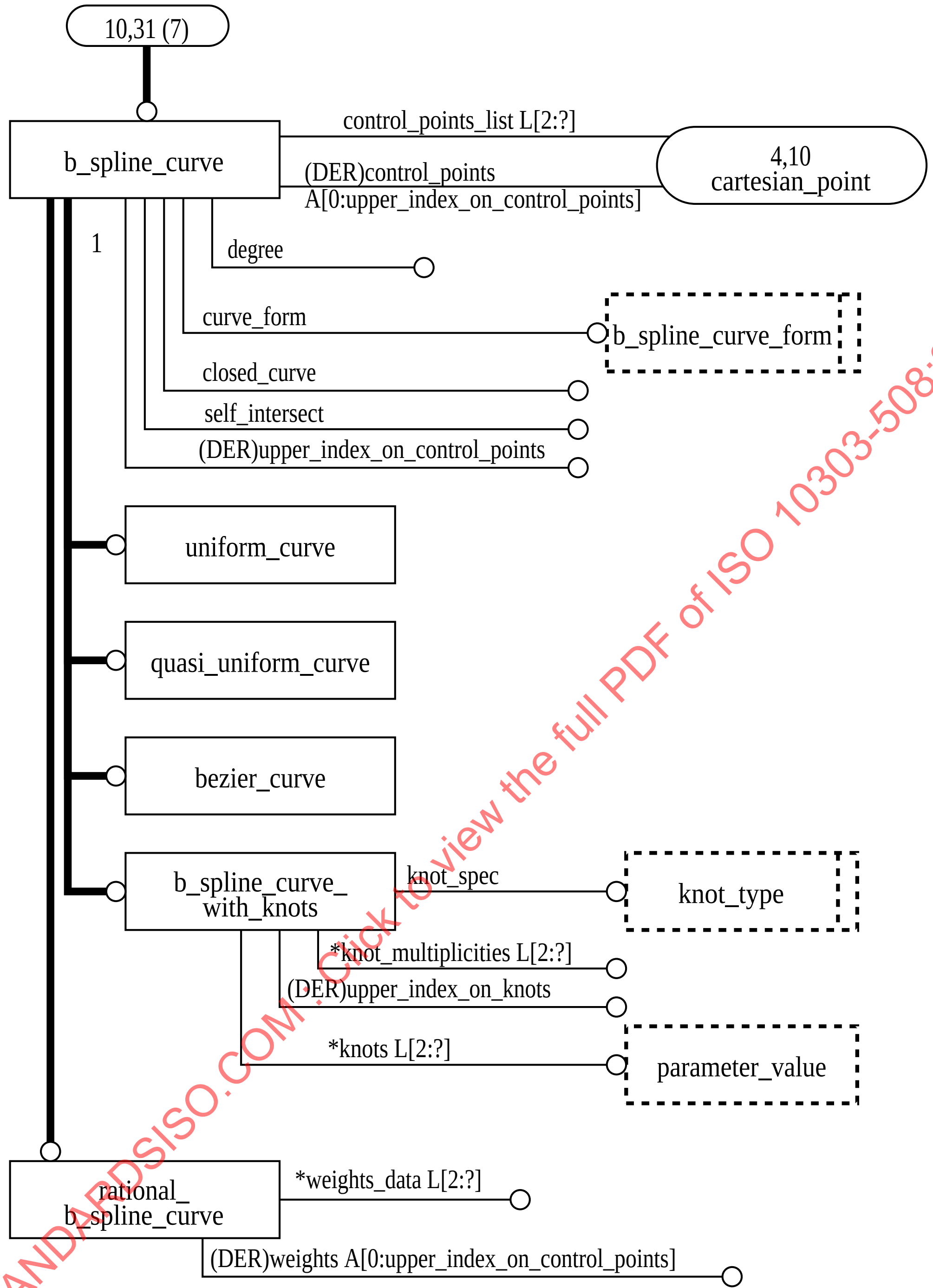

Figure C.10 – EXPRESS-G diagram 10 of 22

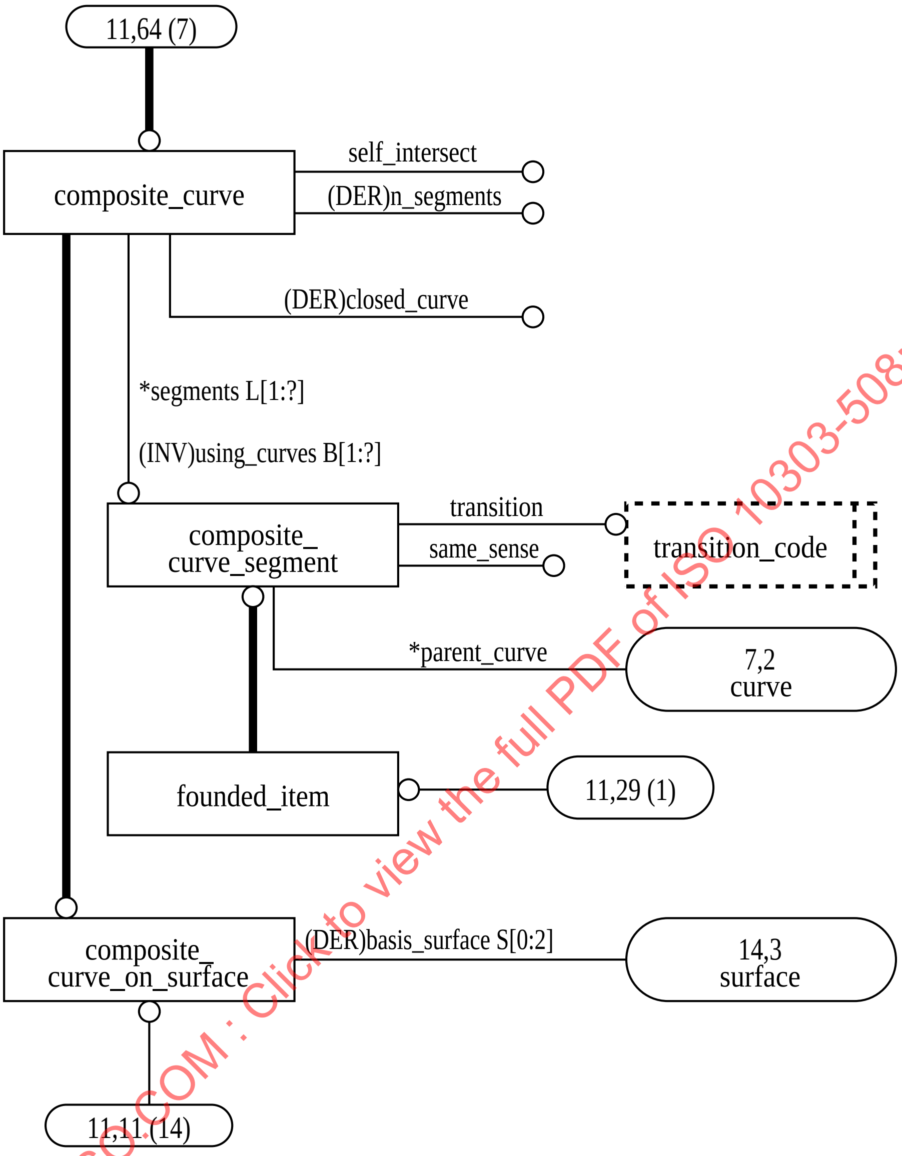

Figure C.11 – EXPRESS-G diagram 11 of 22

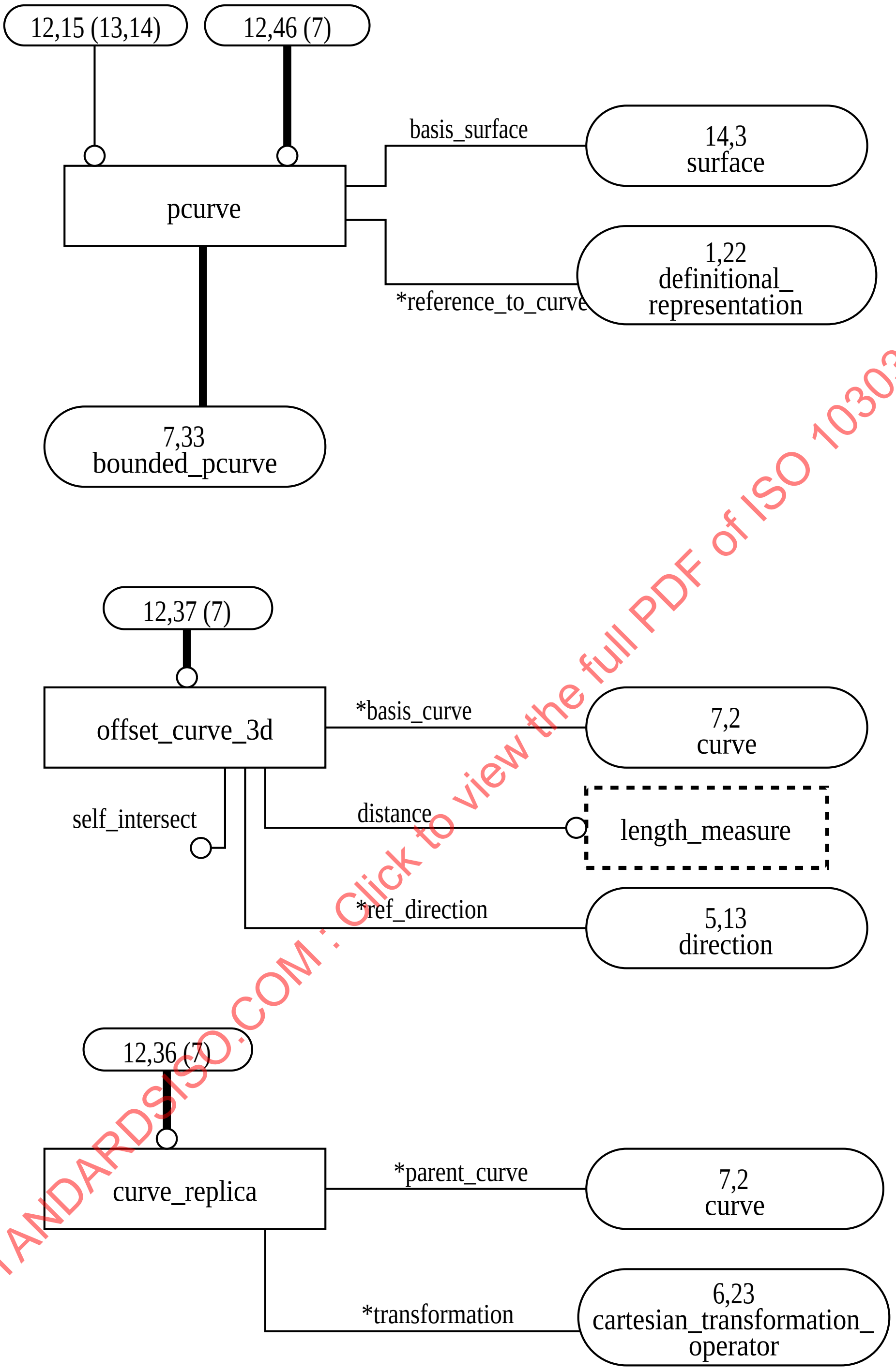

Figure C.12 – EXPRESS-G diagram 12 of 22

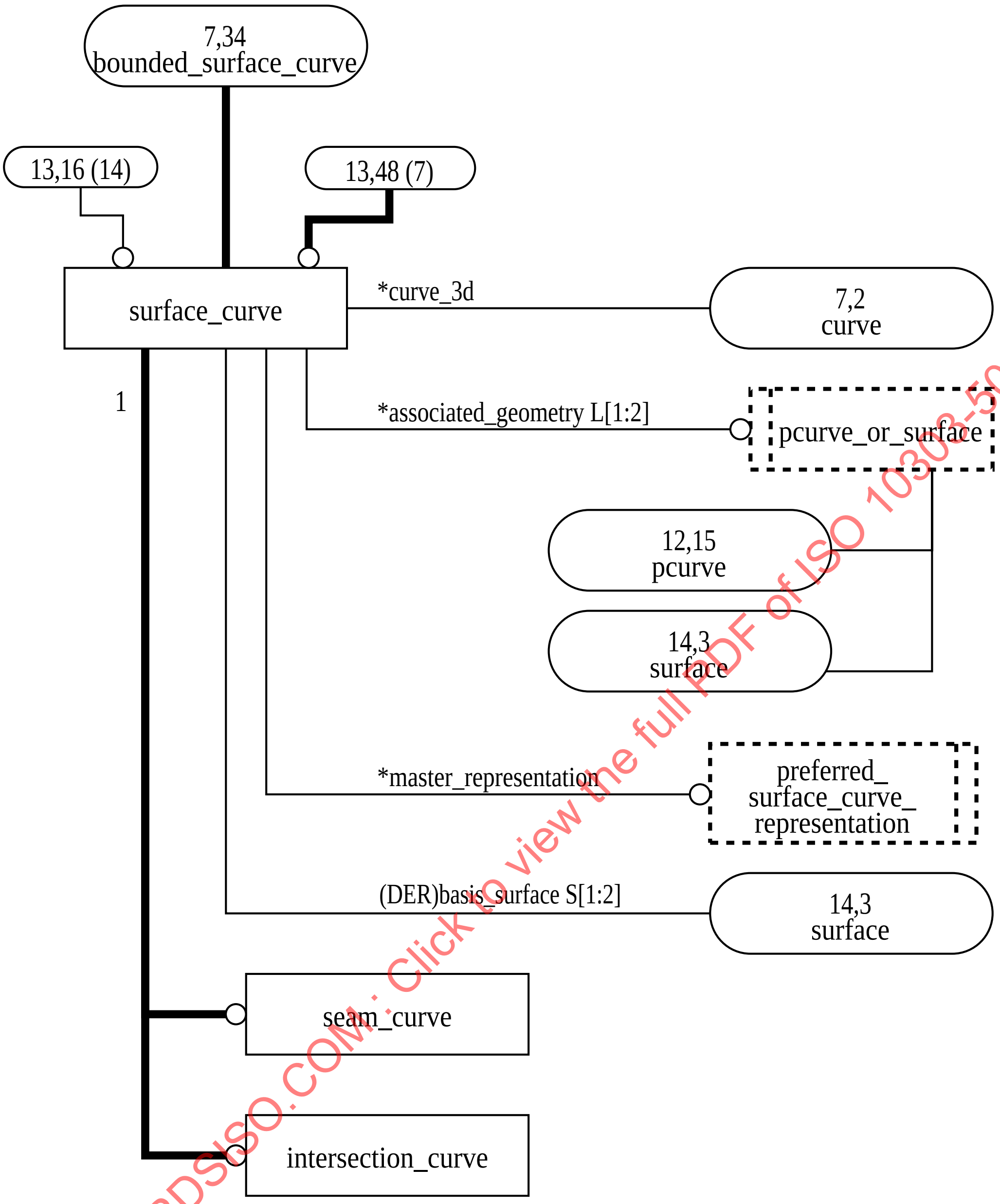

Figure C.13 – EXPRESS-G diagram 13 of 22

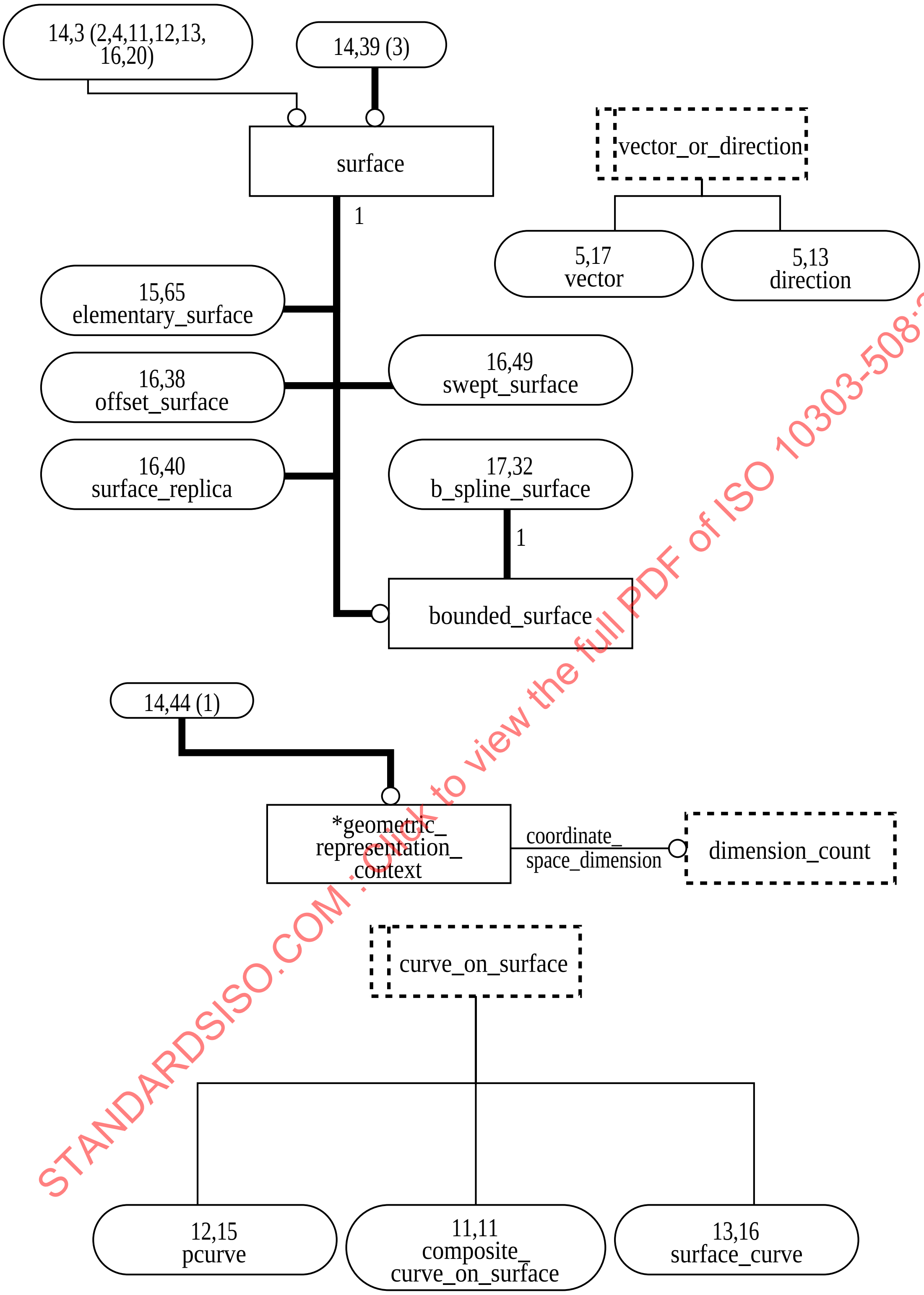

Figure C.14 – EXPRESS-G diagram 14 of 22

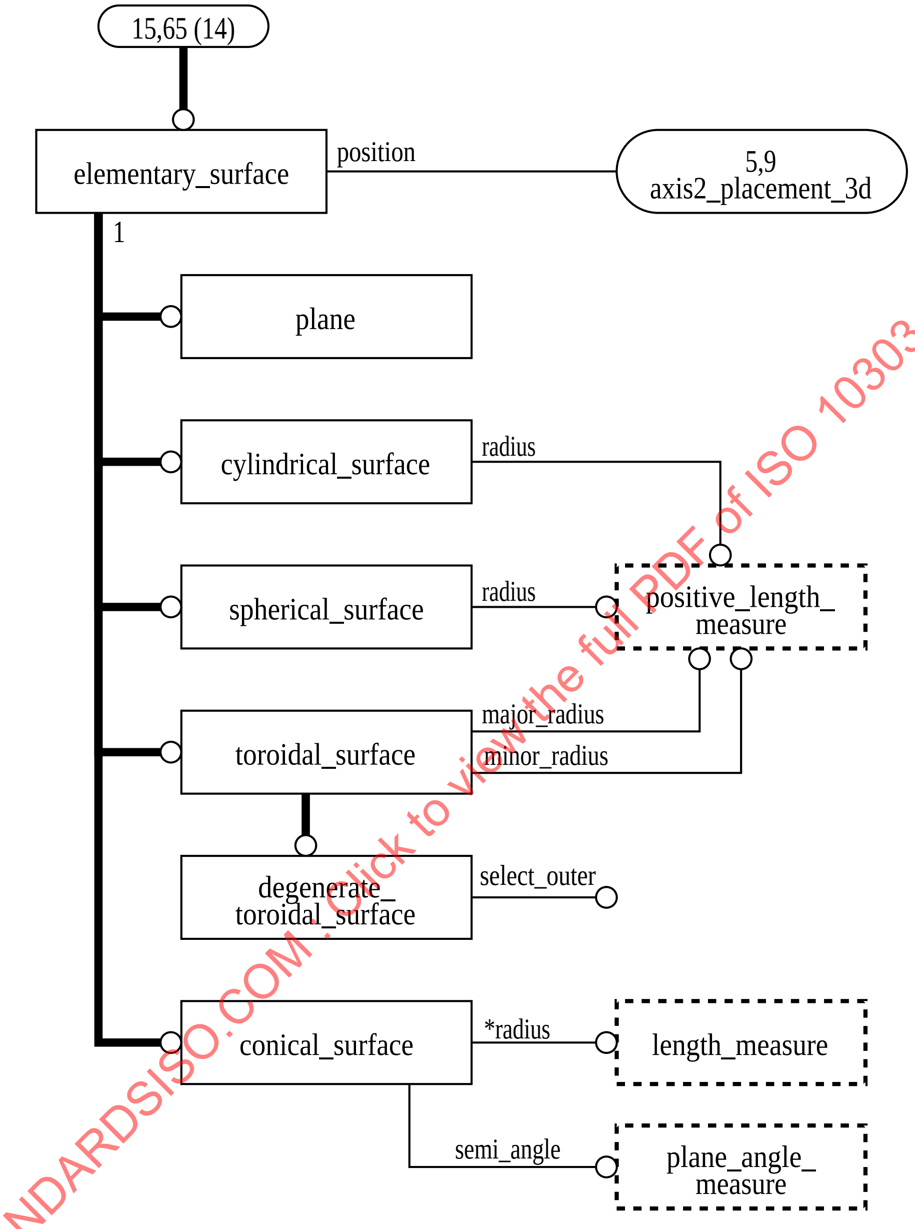

Figure C.15 – EXPRESS-G diagram 15 of 22

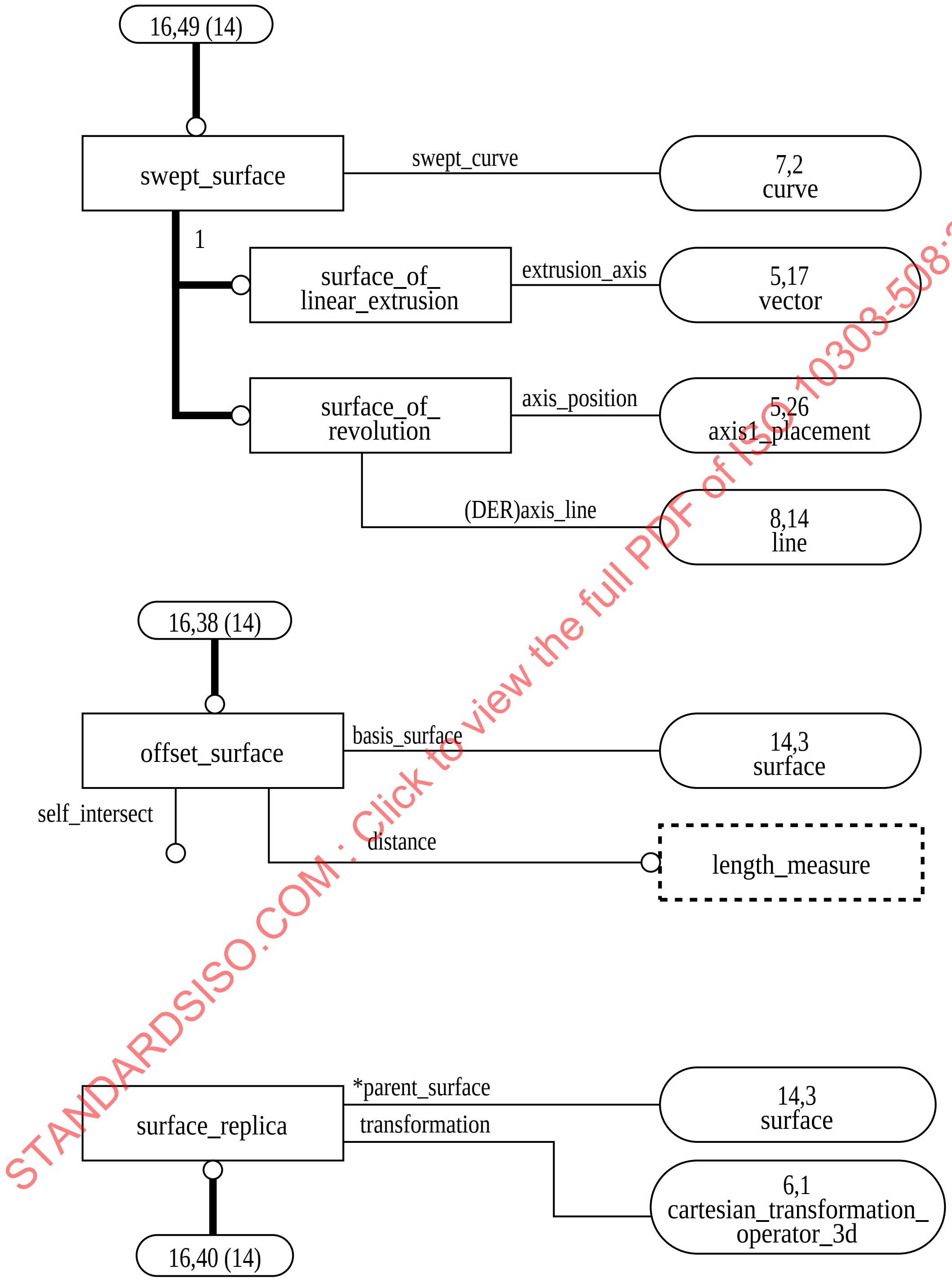

Figure C.16 – EXPRESS-G diagram 16 of 22

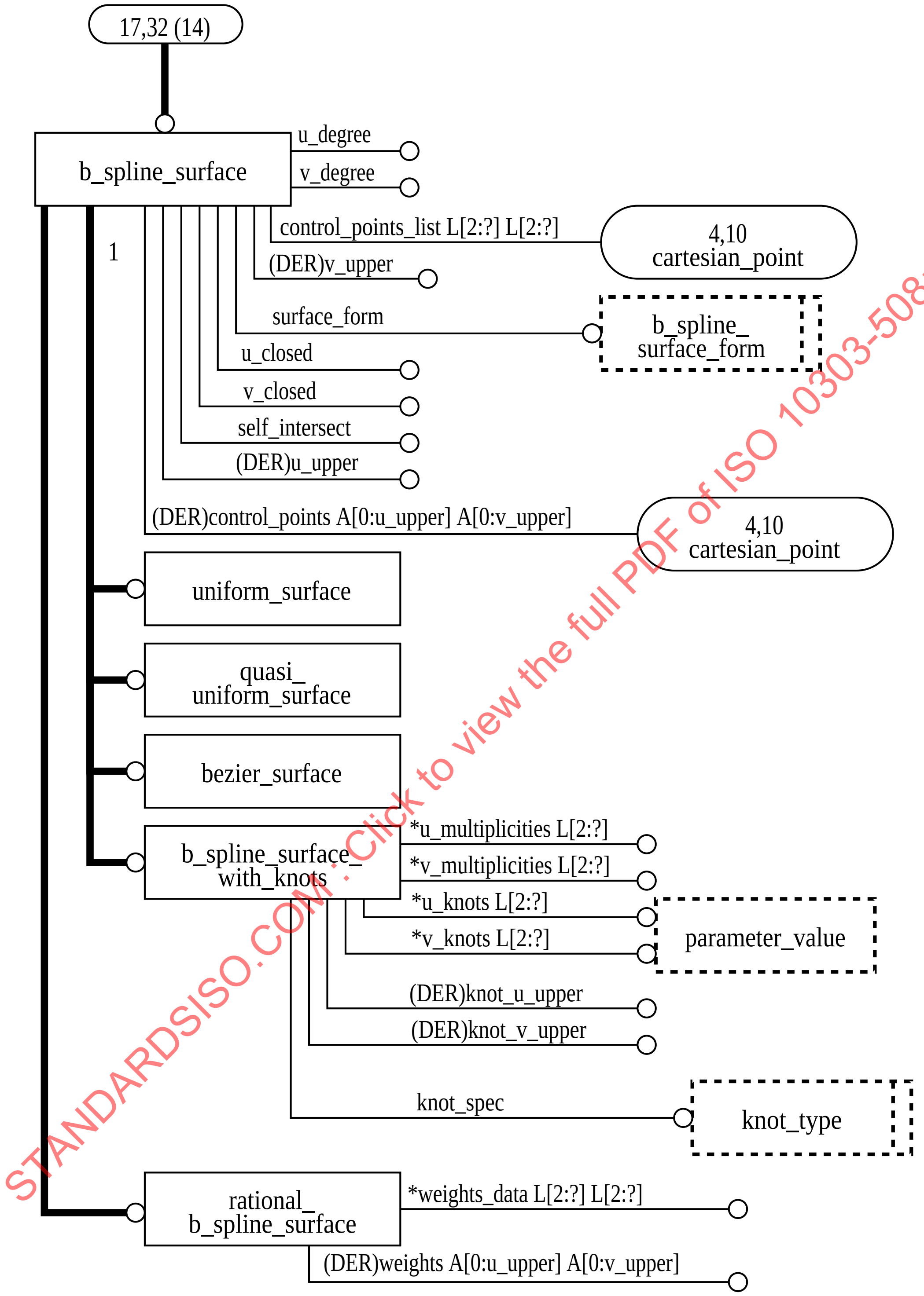

Figure C.17 – EXPRESS-G diagram 17 of 22