Recommended Practices
for
Geometric and Assembly Validation Properties

Release 4.5

August 22, 2019

CAx-IF

Jochen Boy
PROSTEP AG
Dolivostraße 11
64293 Darmstadt / Germany
jochen.boy@prostep.com

Jean-Marc Crepel
AFNeT
30 Rue de Miromesnil
75008 Paris / France
jean-marc.crepel@afnet.fr

Phil Rosché
ACCR, LLC.
125 King Charles Circle
Summerville, SC 29485 USA
phil.rosche@accr-llc.com

Technical

Doug Cheney
ITI - International TechneGroup
doug.cheney@iti-global.com

Table of Contents

List of Figures

Document History

This document replaces the following two CAx-IF Recommended Practices:

- Recommended Practices for Geometric Validation Properties; 2nd Extension; published June 16, 2008
- Recommended Practices for Assembly Validation Properties; Release 1.0; published June 11, 2008

This document combines and extends the scope of the previous documents by adding new and updated concepts.

Revision	Date	Change
3.1	2011-10-19	First release of new combined document
3.2	2012-12-18	Addition of Independent Curve Centroid (4.9.2)
3.3(*)	2013-05-17	Addition of Bound Box agreement (4.10) Addition of Independent Curve Length (0)
4.0c(*)	2014-01-30	Combination of several VP in one property / representation (0) Addition of ID and GISU for shape_aspects (0, 4.5.3) Update of imposed attribute value summary (8)
4.1(*)	2014-06-16	Addition of separate VP for surface data (4.7)
4.2	2014-10-09	Editorial changes for publication
4.3(*)	2015-07-16	Added Independent Points VP (4.9) Updated section 5 for multiple Centroids.
4.4	2016-08-17	Fixed implementation of integer_representation_item
4.5	2019-08-22	Updated Introduction (1) and Scope (2) Added recommendations for handling of geometry (4.2) Updated definitions validation property evaluation (4.13)

(*): Internal review versions; not published.

1 Introduction

This document specifies recommended practices for the exchange of geometric validation properties for solid and surface models. For solid models the properties addressed are centroid, volume, and surface area, and for surface models the properties specified are centroid and surface area. In addition, validation properties for independent curves – i.e. curves included in solid or surface models that are not edges of solids or faces – may be added as well. Geometric validation properties may also be assigned to collections of solids/surfaces which represent the overall validation properties for assembly nodes. This set of data is often referred to as “basic validation properties.”

The first extension of the geometric validation properties introduced the assignment of centroid values for the validation of correct positioning of instances within an assembly. This approach is typically known as “extended validation properties.”

The second revision of the recommended practices added the mechanisms for including an additional set of surface checking properties, commonly known as the Cloud Of PointS, or COPS, method. This methodology allows for the inclusion in the exchange file of a set of points on each individual face, to ensure that the translated face does not deviate from the original surface by more than an accepted amount. This was deemed to be a required check for Long Term Archiving and Retrieval. COPS is also a requirement for certified data delivery using STEP.

The third version of this recommended practices document includes the definition of the so-called “assembly validation properties”, which were previously published as a separate document. These provide a verification capability where geometry is not present when external references are used. They will make it possible for the exchange of assembly data to be verified in two ways. The first will ensure that the number of instances found at each node is correct. The second will ensure that the position and orientation information for each instance is correct.

This current, fourth edition adds support for new, more streamlined implementation approaches as supported by AP242, as well as breaking down the geometric validation properties into separate values for each class of geometry (solids, surfaces, curves and points). It also adds clarifications to previous agreements.

In the meantime, as the general concept proved to add significant value to STEP data exchange, validation properties have been defined for many more capabilities than just geometry and assembly structure. These validation properties are defined in the respective Recommended Practices covering the capability they are designed to validate and not in scope of this document. Their implementation consistently follows the general patterns described below, though.

2 Scope

The following are within scope of this document:

- Assignment of geometric validation properties to individual solids or surface models. These properties include centroid, volume, and total surface area for solids. For surface models the properties addressed are centroid and surface area.
- Assignment of geometric validation properties to independent curves (i.e. curves that are not edges of faces or solids) in solid or surface models, or curves in wireframe models. These properties include total length and centroid.
- Assignment of geometric validation properties to structures of surfaces and solids that represent assemblies and their components.
- Assignment of geometric validation properties to component instances in an assembly.
- Assignment of COPS validation properties to topological face entities, both in surface models and solids.
- Assignment of properties for the top and intermediate nodes of a product structure.
    - Assignment of a numerical property to such nodes to define the number of child instances in that node.
    - Assignment of a centroid type property to such nodes to simulate the existence of a pre-defined notional solid within each child instance node.

The following are outside the scope of this document:

- Assignment of any properties to a product or its definition
- Validation Properties for any characteristics of the model other than geometry and assembly structure. Refer to the respective documents on validation properties for:
    - 3D Tessellated Geometry
    - Product Manufacturing Information
    - User Defined Attributes
    - Composite Structures
- Support for other forms of product structure where `ASSEMBLY_COMPONENT_USAGE` entities or other subtypes are used to relate `PRODUCT_DEFINITIONS` instead of `NEXT_ASSEMBLY_USAGE_OCCURRENCE` entities.

3 Document Identification

For validation purposes, STEP processors shall state which Recommended Practice document and version thereof have been used in the creation of the STEP file. This will not only indicate what information a consumer can expect to find in the file, but even more important where to find it in the file.

This shall be done by adding a pre-defined ID string to the `description` attribute of the `file_description` entity in the STEP file header, which is a list of strings. The ID string consists of four values delimitated by a triple dash ('---'). The values are :

```
Document Type---Document Name---Document Version---Publication Date
```

The string corresponding to this version of this document is:

CAx-IF Rec.Pracs.---Geometric and Assembly Validation Properties---4.5---2019-08-22

It will appear in a STEP file as follows:

```
FILE_DESCRIPTION(('...','CAx-IF Rec.Pracs.---Geometric and Assembly
Validation Properties---4.5---2019-08-22'),'2;1');
```

4 Geometric Validation Properties

The following sections cover the aspects of where to define validation properties in the product structure, and the different types of geometric validation properties.

4.1 Fundamental Concepts

A geometric validation property is a characteristic of a solid/surface model or a collection of them. When used to validate an exchange:

- The sender would populate the geometric validation properties in the exchange file, usually calculated by their CAD/geometry system.
- The receiver would perform geometric translations or transformations that are necessary on the solid or surface model.
- The receiver would then calculate the properties of the resultant geometry.
- A comparison would be performed against these values in the exchange file.
- If they are within an agreed tolerance, the exchange is deemed to be valid.

There are several agreed levels of Validation Properties:

Geometric validation properties assigned to a solid or surface model and/or individual shape representations within an assembly are known as “basic validation properties”. This gives the capability of validating the exchange of each shape and assembly level.

When validation properties are assigned to an assembly instance, then this is known as “extended validation properties”. This extension allows the receiving system to determine which instance in an assembly has failed if the properties do not match. See section 5.

Validation properties included in the form of sampling points for individual faces are referred to as Cloud Of PointS, or COPS. This extension of the recommended practices allows the receiving system to check for misplacements or shape changes of individual faces, as described in section 6.

Validation properties in an assembly structure to verify its completeness and correctness of the included transformations without having to process the part geometries are called “assembly validation properties”, and are defined in section 7.

4.2 Handling of Geometry for Validation

For validation properties to work as expected, it is important that their definitions are consistent across different systems. Thus, when defining properties such as volume or curve length, it is important to agree on what model elements are considered.

The CAx-IF agreement for calculating Geometric Validation Properties at part level is:

- Only use visible 3D elements of the geometry that would be exported to STEP
- Do not include any annotations into the calculation
- Do not include any geometry that is referenced solely by composite structures
- Do not consider supplemental geometry elements

The first point addresses the fact that invisible elements can be much larger than the actual part shape. Also, the relevance of invisible part geometry is unclear, and in most cases, it is not exported at all. Note that there is no issue with providing Geometric Validation Properties for individual invisible elements at the geometry level (see section 4.5), but for the GVP values accumulated at part level, they must not be taken into account.

The second and third points are managed separately and addressed by the PMI Validation Properties for Graphic PMI Presentation (see section 10.2 in the PMI Recommended Practices) and the Recommended Practices for Composite Structure Validation Properties respectively.

For the last point, refer to the Recommended Practices for Supplemental Geometry. Reference elements shall not be included in the Geometric Validation Properties as their definition may change between CAD systems.

4.3 Definitions of Units

The validation properties defined in this document represent different types of measures: for volume, for area, and for length. Each of these requires a correct definition of the applied unit of measure in the STEP file.

A comprehensive guide on the correct definition of these and other units is given in Annex C of the CAx-IF Recommended Practices for User Defined Attributes, which can be found on the CAx-IF Homepage (www.cax-if.de and www.cax-if.org) under “Joint Testing Information”.

4.4 Validation Properties at the Part/Product level

Geometric Validation Properties can be attached to the geometry in a STEP file at different levels of granularity, i.e. single solids, faces or curves, or entire parts. Certain CAx systems can determine the validation information for individual geometries within a part, whereas others are designed as a “one solid per part” system and can only determine validation properties at the part level. The following diagram shows how validation properties are attached to the geometry defining the entire product.

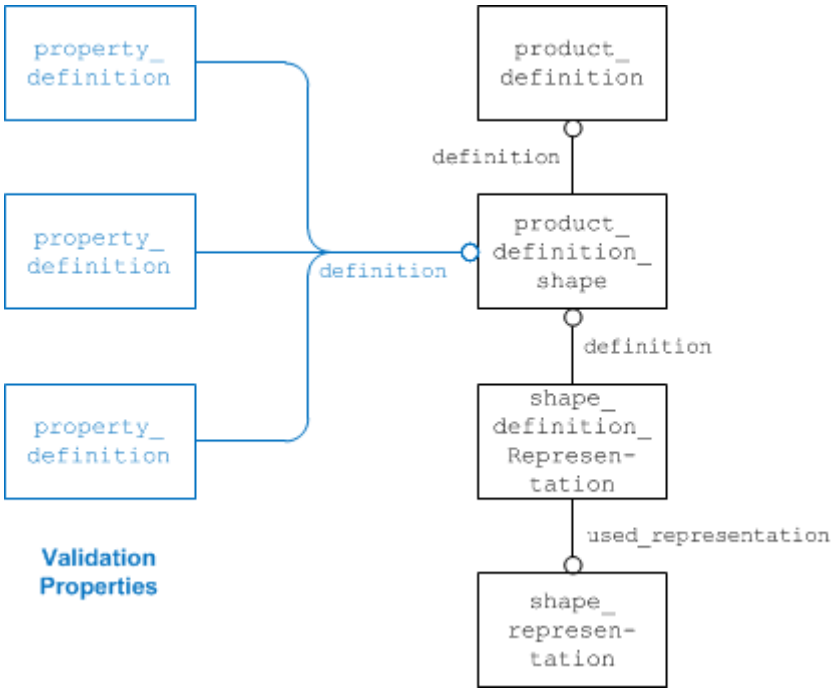

Figure 1: Validation Properties at the Part/Product level

Part 21 Example:

```
#15=PRODUCT_DEFINITION('design',$,#14,#9);
#907=PRODUCT_DEFINITION_SHAPE('',$,#15);
#960=SHAPE_DEFINITION_REPRESENTATION(#907,#896);
#896=SHAPE_REPRESENTATION('#896',(#895,#442,#447,#452,#889),#891);
#910=PROPERTY_DEFINITION('geometric validation property', 'volume of shape
rep #896 - part44_lbrackass',#907);
#914=PROPERTY_DEFINITION('geometric validation property', 'area of shape
rep #896 - part44_lbrackass',#907);
#918=PROPERTY_DEFINITION('geometric validation property', 'centroid of
shape rep #896 - part44_lbrackass',#907);
```

Note that Geometric Validation Properties at the Part/Product level shall be computed solely based on the (manufactured) part shape. They shall not take any Supplemental Geometry into account, since not all target applications support Supplemental Geometry. In addition, Supplemental Geometry may contain unbounded elements. Validation Properties for Supplemental Geometry may be defined at the geometry level.

4.5 Validation Properties at the Geometry Level

Important Agreement:

Every CAD system supporting validation properties on export shall attach them at the part/product level. If an exporting CAD system also supports validation properties at the geometry level, it may add them there additionally.

Motivation for this agreement:

- Only if the validation properties are attached at the part/product level, it can be guaranteed that every CAD system will find them. If system A only attaches them at the geometry level and importing system B does not support multiple bodies per part, it will not be able to find the information as it will look for it at the part/product level only.
- Another important motivation for this is PDM interoperability: If an assembly is exported using the "external references" mechanism, i.e. it is split into a structure file and several geometry files, the validation properties (which are deemed PDM-relevant data) will be included in the structure file only if they are attached at the part/product level. Validation properties at the geometry level will be stored in the geometry files and hence are inaccessible for the PDM system.

The attachment of validation properties to a single solid, surface or curve within the product geometry is handled via the `SHAPE_ASPECT` entity. There are two ways to associate the `SHAPE_ASPECT` with its geometric content: the “old” way using a `SHAPE_REPRESENTATION`, and a newer way using `GEOMETRIC_ITEM_SPECIFIC_USAGE`.

Geometric validation properties can be associated with the following geometric items:

- `MANIFOLD_SOLID_BREP`
- `BREP_WITH_VOIDS`
- `SHELL_BASED_SURFACE_MODEL`
- `GEOMETRICALLY_BOUNDED_WIREFRAME_SHAPE_REPRESENTATION`
- `OPEN_SHELL`
- `CLOSED_SHELL`
- `ADVANCED_FACE`
- `GEOMETRIC_CURVE_SET`
- `CURVE`

For details see 0 and 4.5.3 below.

Note: For the different entity types, different sets of validation properties apply. While for solids (BREPs) volume, area and centroid apply, for shells and surfaces only area and cloud of points are applicable. For curves and wireframes, curve length and centroid apply.

4.5.1 Shape Aspect Identification in AP242

In AP242, there is a uniqueness rule on each of SHAPE_ASPECT, DIMENSIONAL_LOCATION, DIMENSIONAL_SIZE and SHAPE_ASPECT_RELATIONSHIP which requires the attribute pair (ID, OF_SHAPE) to be unique if the ID attribute exists. There is also a global rule requiring uniqueness of the ID attribute across population of a collection of the above entity types if the ID attributes exist. These rules have been introduced in the context of the Semantic Product and Manufacturing Information (PMI) Representation capabilities and External Element References (EER). The second rule is more restrictive as it requires coordination amongst several entity types. For backward compatibility reasons, AP242 does not formally require the ID attribute to exist.

Since the ID attribute is derived, an instance of ID_ATTRIBUTE must be populated, which has the ID string as its ATTRIBUTE_VALUE and any of the aforementioned entity types as IDENTIFIED_ITEM.

While adding the ID_ATTRIBUTE is allowed but not required in the formal AP242 document, omitting it in an AP242 file will violate the business agreement for Semantic PMI and EER. Also, in order not to have to make the decision what purpose a SHAPE_ASPECT is used for, it is recommended to add an ID_ATTRIBUTE to all instances of the above entities, with an ATTRIBUTE_VALUE string that is unique among all instances of ID_ATTRIBUTE in the context of the respective PRODUCT_DEFINITION_SHAPE, i.e. if there are 8 ID_ATTRIBUTES that reference a combination of the above types which all reference the same PRODUCT_DEFINITION_SHAPE in their OF_SHAPE attribute, there shall be 8 distinct values of ATTRIBUTE_VALUE.

There is no business requirement to add ID_ATTRIBUTE in AP203e2 or AP214 files, since Semantic PMI and EER are out of scope for these APs. It is, however, technically legal to do so.

4.5.2 Geometry Assignment using Shape Representation

This method has been used for Validation Properties at the Geometry Level since its inception. It can associate several geometric elements to a SHAPE_ASPECT but uses three entities to do so. Also, a large number of additional REPRESENTATIONS may have an impact on system performance. The structure by which this relationship is conveyed is shown below.

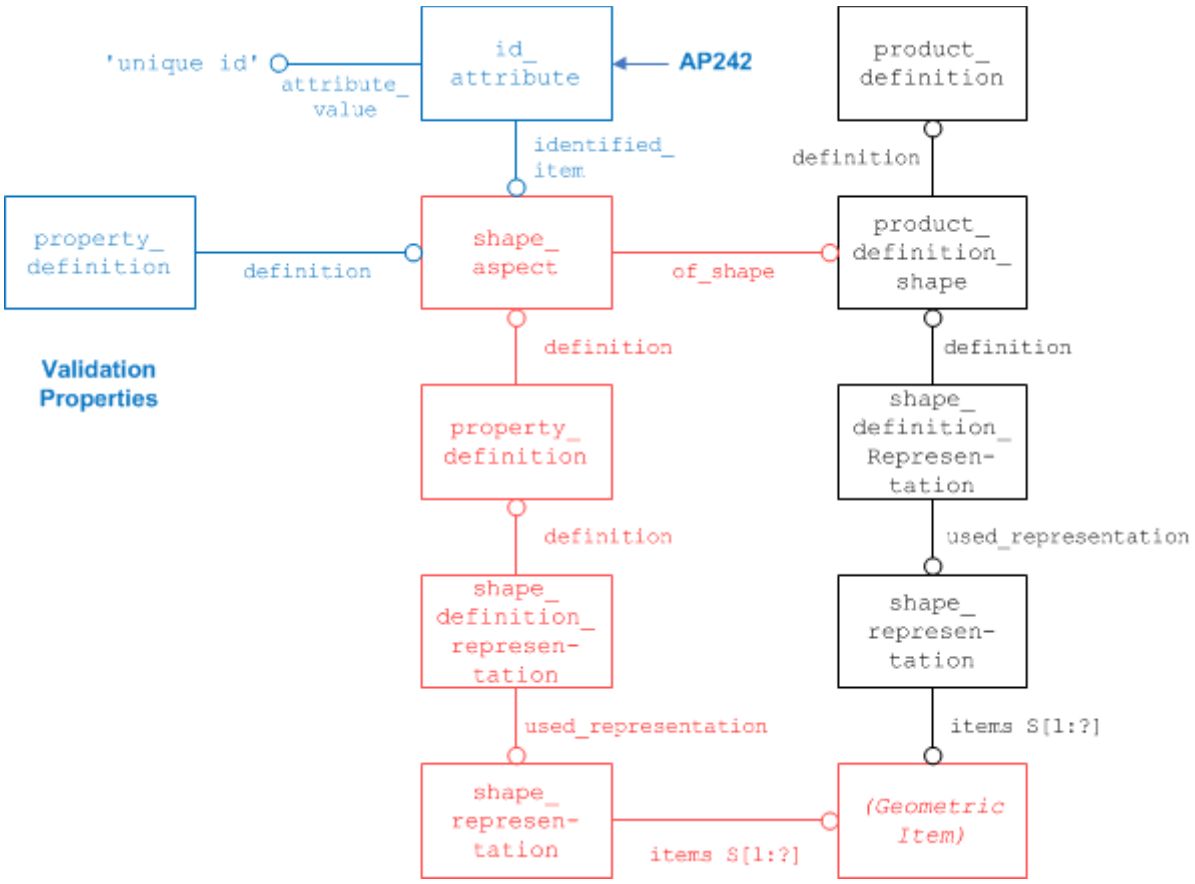

Figure 2: Validation Properties at the Geometry Level (Shape Representation)

Part 21 Example:

```
#21=PRODUCT_DEFINITION('design',$,#20,#9);
#217=MANIFOLD_SOLID_BREP('',#216);
#224=ADVANCED_BREP_SHAPE_REPRESENTATION('#224',(#217,#223),#219);
#231=PRODUCT_DEFINITION_SHAPE('',$,#21);
#232=SHAPE_ASPECT('#232','solid #217',#231,.F.);
#233=ID_ATTRIBUTE('solid #217 for #231',#232);
#246=PROPERTY_DEFINITION('','Shape for Validation Properties',#232);
#247=SHAPE_DEFINITION_REPRESENTATION(#246,#245);
#245=SHAPE_REPRESENTATION('',(#217),#219);
#961=SHAPE_DEFINITION_REPRESENTATION(#231,#224);
```

4.5.3 Geometry Assignment using Geometric Item Specific Usage

In AP203 Edition 2 and AP214 Edition 3, the new entity type GEOMETRIC_ITEM_SPECIFIC_USAGE (GISU) was introduced which was not available in earlier data models. It allows for a much more efficient implementation, as only one entity and no additional representation is needed. Since GVP are assigned at the geometry level, they are assigned to one specific element (solid, shell, face, curve), and GISU can be used with no restrictions.

Note: AP242 introduces a uniqueness rule on GISU which limits the number of GISU instances per SHAPE_ASPECT to one. Since a GISU can relate to only a single geometric item, if several geometric elements need to be associated with a SHAPE_ASPECT, the supertype ITEM_IDENTIFIED_REPRESENTATION_USAGE has to be used.

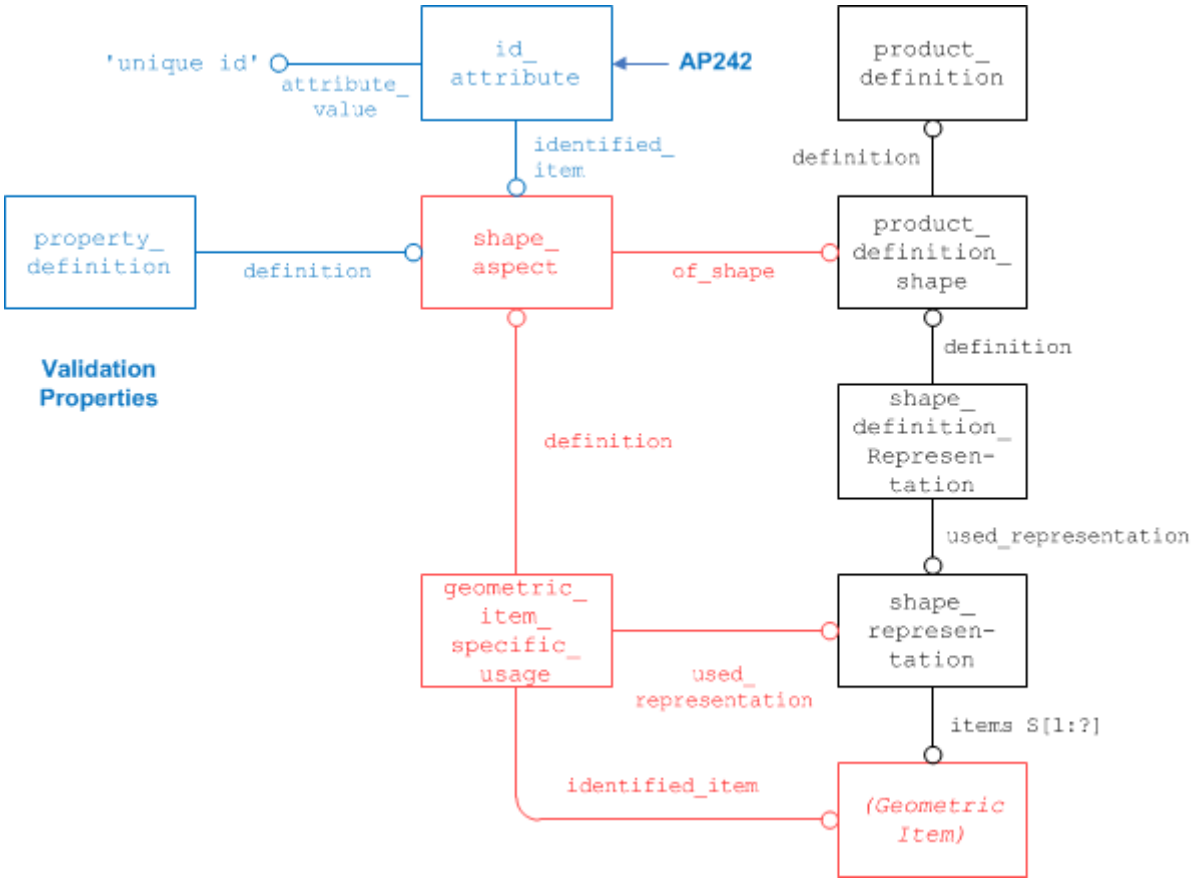

Figure 3: Validation Properties at the Geometry Level (GISU)

Part 21 Example:

```
#21=PRODUCT_DEFINITION('design',$,#20,#9);
#217=MANIFOLD_SOLID_BREP('',#216);
#224=ADVANCED_BREP_SHAPE_REPRESENTATION('#224',(#217,#223),#219);
#231=PRODUCT_DEFINITION_SHAPE('',$,#21);
#232=SHAPE_ASPECT('#232','solid #217',#231,.F.);
#233=ID_ATTRIBUTE('solid #217 for #231',#232);
#246=GEOMETRIC_ITEM_SPECIFIC_USAGE('Shape for Validation Properties','',
#232,#224,#217);
```

4.6 Validation Properties for different Classes of Geometry

It is not unusual that a geometric model contains different classes of geometry. Such a “hybrid” model contains not only solid data, but also additional surfaces, curves or points that are independent, i.e. do not take part in the definition of a higher-level element.

These independent elements are treated differently by the various CAD systems when validation properties are calculated. For instance, if a model contains both solids and independent surfaces, some systems calculate only the solid area, while others calculate the combined solid and surface area. This ambiguity can easily lead to false errors when evaluating the validation properties.

Hence, geometric validation properties are defined independently for each class of geometry:

- *Solids:* volume, surface area, and centroid
- *Surfaces:* surface area, and centroid
- *Curves:* curve length, and centroid
- *Points:* number of points, and centroid

This means that if a model contains solids, independent surfaces, independent curves, and independent points, there will be four different centroids given in the validation properties: one for each class.

4.7 Validation Properties for Solid Geometry

4.7.1 Volume

Volume specifies the amount of space occupied by the solid model as measured in cubic units. During an exchange this can be used to validate the success of creating an equivalent solid via the translation.

Figure 4 illustrates the STEP entities required to specify the volume of the original solid, or part, in the native system.

Part 21 Example:

```
#233=MEASURE_REPRESENTATION_ITEM('volume measure',
VOLUME_MEASURE(664.38055098),#226);
#234=REPRESENTATION('volume',(#233),#219);
#235=PROPERTY_DEFINITION('geometric validation property', 'volume of
#217',#232);
#236=PROPERTY_DEFINITION_REPRESENTATION(#235,#234);
```

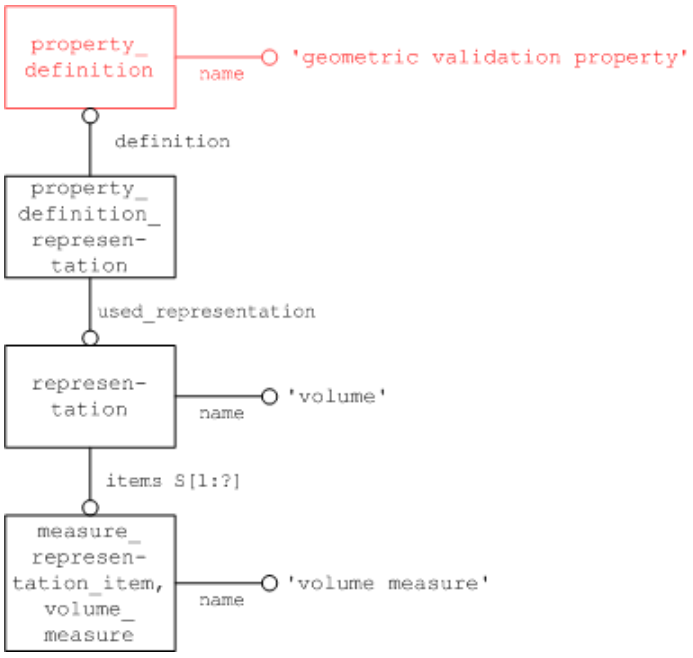

Figure 4: Geometric Validation Property "Volume"

Note: The PROPERTY_DEFINITION instance the value is attached to in this and the following diagrams is the one related to the part or the geometry the value applies to; see blue instance labeled "Validation Properties" in Figure 1 and Figure 2 respectively.

4.7.2 Surface Area

Surface area specifies the area measurement of the surface of the entire solid. By default, this will include any voids in the model. Figure 5 below illustrates the relevant entities and their mandatory attributes used in the assignment of the surface area validation property.

Note: Since CATIA calculates the 'wetted area' (i.e. voids will not be taken into account) instead of the total surface area, the validation mechanism will report a 'false error' when exchanging a model with voids in it between a CATIA-based and a non-CATIA-based system. Therefore, when exporting validation properties from a CATIA-based system, the name of the `MEASURE_REPRESENTATION_ITEM` (see Figure 5) shall be 'wetted area measure' instead of 'surface area measure'.

Part 21 Example:

```
#237=MEASURE_REPRESENTATION_ITEM('surface area measure',
AREA_MEASURE(747.16814693),#229);
#238=REPRESENTATION('surface area',(#237),#219);
#239=PROPERTY_DEFINITION('geometric validation property','area #217',#232);
#240=PROPERTY_DEFINITION_REPRESENTATION(#239,#238);
```

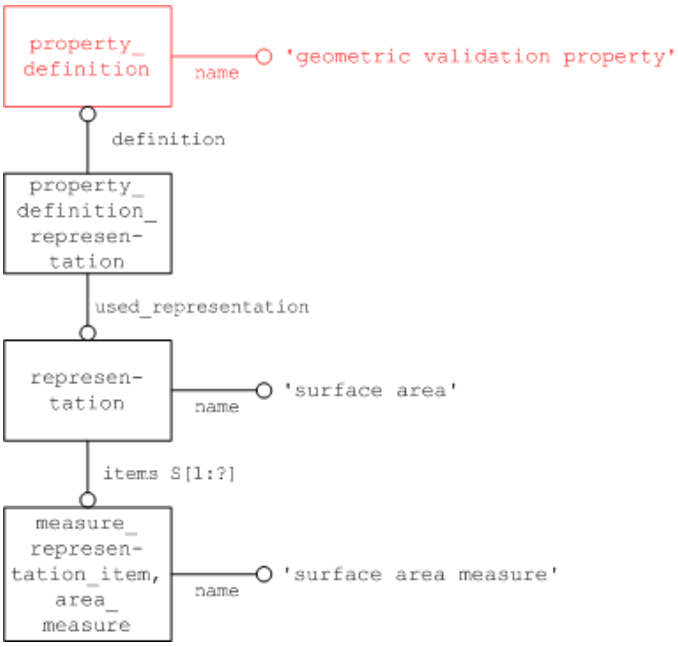

Figure 5: Geometric Validation Property "Surface Area"

4.7.3 Solid Centroid

A centroid is the center of volume of a geometric solid model. The position of the centroid is an invariant datum relative to the model origin, thus during an exchange, this can be used to validate the positional integrity of any geometric translations.

Figure 6 illustrates the relevant entities and their mandatory attributes used in the assignment of a solid centroid for validation.

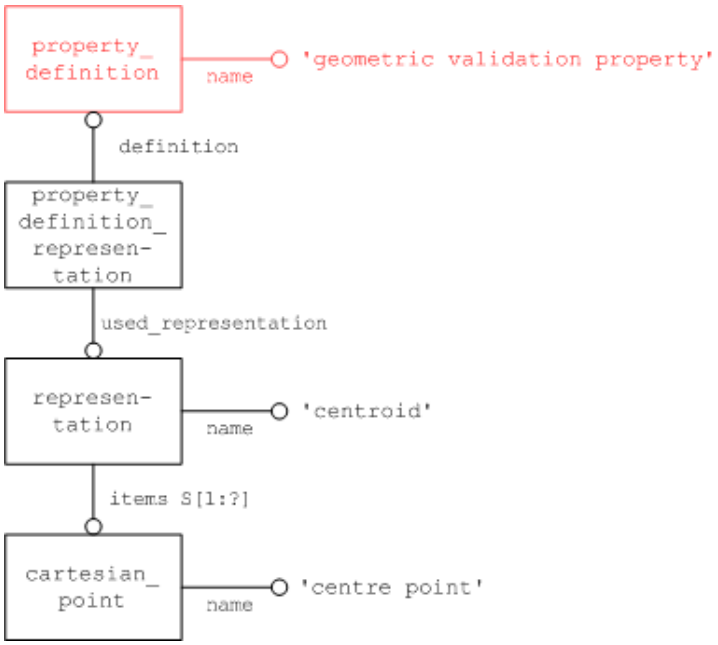

Figure 6: Geometric Validation Property "Centroid"

Part 21 Example:

```
#241=CARTESIAN_POINT('centre point',(0.0,0.0,1.5));
#242=REPRESENTATION('centroid',(#241),#219);
#243=PROPERTY_DEFINITION('geometric validation property', 'centroid of
#217',#232);
#244=PROPERTY_DEFINITION_REPRESENTATION(#243,#242);
```

4.8 Validation Properties for Surface Geometry

4.8.1 Independent Surface Area

The designation “independent” for a surface means that it is not a face of a solid. Such surfaces can occur as constituents of a surface model (open or closed shell), or as additional elements in a solid model. The total area of these surfaces in a model can be validated to ensure completeness of the data.

The instantiation follows the exact same pattern as defined in section 0, Figure 5 and the subsequent Part21 Example, but uses different magic strings:

- `REPRESENTATION.NAME` = “independent surface area”
    - **Note** that `REPRESENTATION.NAME` will be empty when combining several validation properties for one model element as defined in section 0 below)
- `REPRESENTATION_ITEM.NAME` = “independent surface area measure”

4.8.2 Independent Surface Centroid

In addition to the total area of independent surfaces (see section above), their positioning is of interest as well. This can be validated using the combined centroid of all independent surfaces in the model.

The instantiation follows the exact same pattern as defined in section 0, Figure 6 and the subsequent Part21 Example, but uses different magic strings:

- `REPRESENTATION.NAME` = “independent surface centroid”
    - **Note** that `REPRESENTATION.NAME` will be empty when combining several validation properties for one model element as defined in section 0 below)
- `CARTESIAN_POINT.NAME` = “surface centre point”

4.9 Validation Properties for Curve / Wireframe Geometry

4.9.1 Independent Curve Length

The designation “independent” for a curve means that it is not the edge curve of a surface or solid. Such curves can occur as constituents of a wireframe model, or as additional elements in a surface or solid model. The total length of these curves in a model can be validated to make sure no information was lost during transfer. The independent curve length geometric validation property shall be given as a sum at the part level, see section 0.

It can in addition be given at the geometry level, as the total length of one of these independent curves; see section 4.5. Use cases for this are electric harnesses and piping installations, were an independent curve is used as the center curve of the wire or pipe.

Figure 7 illustrates the STEP entities required to specify the total length of the independent curves in the model.

Part 21 Example:

```
#257=MEASURE_REPRESENTATION_ITEM('curve length measure',
LENGTH_MEASURE(47.1681),#249);
#258=REPRESENTATION('independent curve length',(#257),#219);
#259=PROPERTY_DEFINITION('geometric validation property','',#252);
#260=PROPERTY_DEFINITION_REPRESENTATION(#259,#258);
```

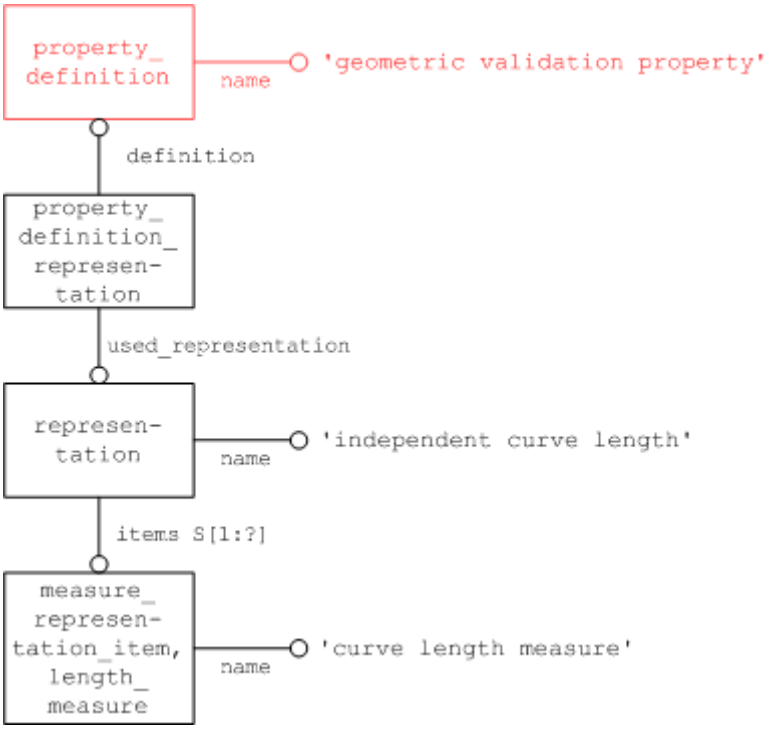

Figure 7: Geometric Validation Property "Independent Curve Length"

4.9.2 Independent Curve Centroid

In addition to the total length of independent curves in a model (see previous section), their position is of interest as well. The independent curve centroid shall store the combined centroid of all independent curves at the part level, see section 0.

In addition, it is also possible to store the curve centroid at the geometry level, to validate the position of one of these independent curves; see section 4.5. Use cases for this are electric harnesses and piping installations, were an independent curve is used as the center curve of the wire or pipe.

The instantiation follows the exact same pattern as defined in section 0, Figure 6 and the subsequent Part21 Example, but uses different magic strings:

- `REPRESENTATION.NAME` = "independent curve centroid"
    - **Note** that `REPRESENTATION.NAME` will be empty when combining several validation properties for one model element as defined in section 0 below)
- `CARTESIAN_POINT.NAME` = "curve centre point"

4.10 Validation Properties for Independent Points

4.10.1 Number of Points

The designation “independent” for a point means that it is not used in the definition of a higher-level element, i.e. as an edge vertex, as a B-Spline control point, or as origin of an axis placement. Such independent points may serve various purposes, such as tool targets or reference points, and in most cases will be classified as Supplemental Geometry.

Since it is not meaningful to define validation properties on an individual point – the validation information would just be a repetition of the point’s definition – the number of points should be given at the Part/Product level.

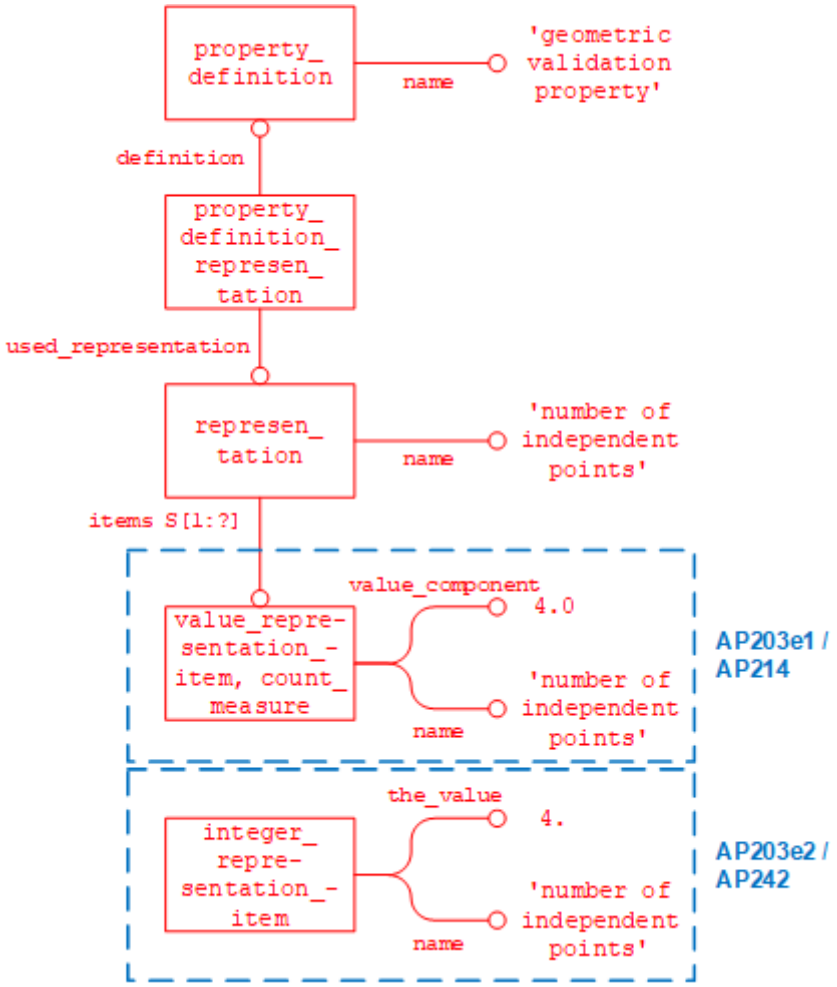

Figure 8: Geometric Validation Property “Number of Independent Points”

Note that there are two ways of implementing this. Though the first way is valid in all APs, the second way preserves the integer type of this value:

- In AP203e1 and AP214, it will be a `VALUE_REPRESENTATION_ITEM` with a `VALUE_COMPONENT` which is a `COUNT_MEASURE` (real number).
- In AP203e2 and AP242, it will be an `INTEGER_REPRESENTATION_ITEM` where `THE_VALUE` is an integer value.

Part 21 Example:

```
/* AP203e1 / AP214 */
#367=VALUE_REPRESENTATION_ITEM('number of independent points',
COUNT_MEASURE(4.0));
/* AP203e2 / AP242 */
#367=INTEGER_REPRESENTATION_ITEM('number of independent points',4.);

#368=REPRESENTATION('number of independent points',(#367),#219);
#369=PROPERTY_DEFINITION('geometric validation property','',#252);
#370=PROPERTY_DEFINITION_REPRESENTATION(#369,#368);
```

Note: The value of INTEGER_REPRESENTATION_ITEM has to be written as a REAL number, i.e. with trailing decimal point.

- In general, values of type INTEGER are represented in a Part 21 file as integers, i.e. no decimal point. The case of INTEGER_REPRESENTATION_ITEM, however, is special. INTEGER_REPRESENTATION_ITEM is a subtype of INT_LITERAL, which is a subtype of LITERAL_NUMBER. The latter defines the attribute the_value as of type NUMBER; INT_LITERAL then re-declares that to restrict it to INTEGER. Part 21 defines that in the case of a re-declared attribute, the original (more generic) type shall be used for implementation, here: NUMBER. And NUMBER maps to REAL in Part 21, hence the decimal point. So basically, INTEGER_REPRESENTATION_ITEM is an integer with an identity crisis, but from the context (name of the entity) it is clear that the decimal point shall be ignored.
- For compatibility with existing data, INTEGER_REPRESENTATION_ITEM values without decimal points shall be supported as well on import.

4.10.2 Independent Points Centroid

In addition to the number of independent points in a model (see previous section), their position is of interest as well. The independent point centroid shall store the combined centroid of all independent points at the part level, see section 0.

The instantiation follows the exact same pattern as defined in section 0, Figure 6 and the subsequent Part21 Example, but uses different magic strings:

- REPRESENTATION.NAME = “independent points centroid”
    - **Note** that REPRESENTATION.NAME will be empty when combining several validation properties for one model element as defined in section 0 below)
- CARTESIAN_POINT.NAME = “independent points centre point”

4.11 Bounding Box

The bounding box is a means of providing information about the model extent and location. It can be used as a further way of validating the position of the model by providing the space it fits into, in addition to the centroid. The Bounding Box also provides the model size, which is defined as the length of its space diagonal. This can be used to put the (absolute) deviation of the centroid in relation to the model size, see section 0 below.

Dedicated tests in the CAx-IF have shown that the various CAD systems and validation tools determined the model extent in many different ways, thus making a comparison of these values meaningless. Hence it was agreed that a common algorithm to determine the bounding box shall be given and only bounding boxes computed per this agreement shall be exchanged as a Geometric Validation Property.

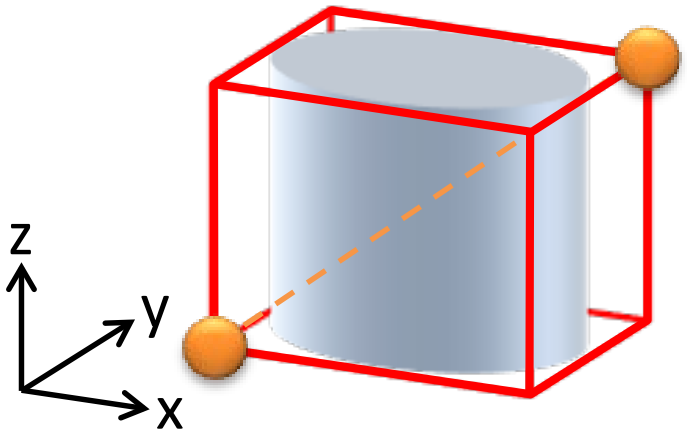

Figure 9: Bounding Box with corner points $Min_{(x,y,z)}$ and $Max_{(x,y,z)}$ for space diagonal

CAx-IF Agreement on the Algorithm to calculate the Bounding Box (see Figure 9):

- Only use visible 3D elements of the geometry that would be exported to STEP
- Do not include any annotations or axis systems into the calculation
- Use the vertices and edges of the B-Rep and Wireframe geometry, including independent points, or use all vertices from a tessellation of the model, which in many systems are being created anyway for display purposes.
- The Bounding Box has its edges parallel to the axes of the model coordinate system.
- Determine the two points
  - $Min_{(x,y,z)}$ = (minimum X, minimum Y, minimum Z) and
  - $Max_{(x,y,z)}$ = (maximum X, maximum Y, maximum Z)

The two points $Min_{(x,y,z)}$ and $Max_{(x,y,z)}$ will be stored as `CARTESIAN_POINTS` and define the Bounding Box Geometric Validation Property. They provide all information needed to recreate the box as a cuboid with axes parallel to the model coordinate system and to easily determine the model size (length of the space diagonal of the bounding box = absolute three-dimensional distance between the two points).

Part 21 Example:

```
#241=CARTESIAN_POINT('bounding box corner point',(0.5,0.5,1.5));
#242=CARTESIAN_POINT('bounding box corner point',(3.0,3.0,4.5));
#258=REPRESENTATION('bounding box',(#241,#242),#219);
#259=PROPERTY_DEFINITION('geometric validation property','',#252);
#260=PROPERTY_DEFINITION_REPRESENTATION(#259,#258);
```

Note: In the case of assemblies, it is recommended to calculate and include the Bounding Box validation property only at the part level for each component. The reason is that the Bounding Box of a positioned (rotated) component instance will be different from the original Bounding Box, except in symmetric cases. For the validation of assemblies, the recommendation is to use part-level Bounding Boxes in combination with the Assembly Validation Properties as defined in sections 5 and 7 below.

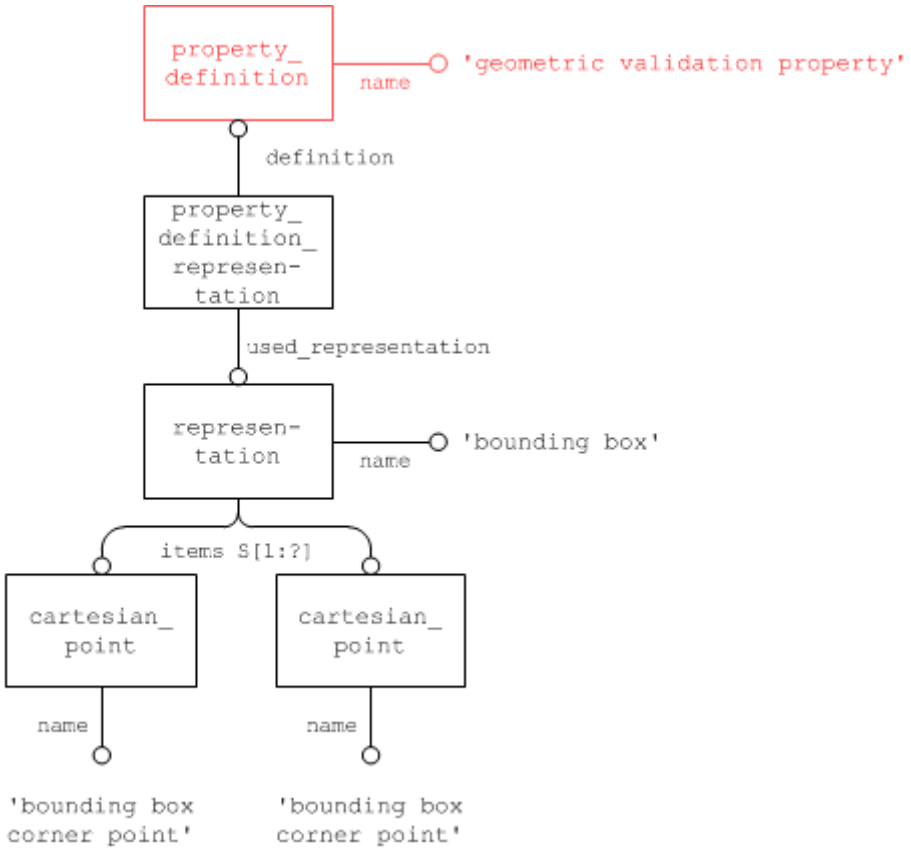

Figure 10: Geometric Validation Property "Bounding Box"

4.12 Combining Validation Properties for Efficient Implementation

Each of the Geometric Validation Properties described above follow the exact same pattern for its definition: PROPERTY_DEFINITION ← PROPERTY_DEFINITION_REPRESENTATION → REPRESENTATION → REPRESENTATION_ITEM. In a larger file, where there are many validation properties – most likely of various types – this can result in a significant number of entities, especially REPRESENTATIONS, which may have an impact on system performance.

Hence it was agreed that under specific circumstances, multiple validation properties can be combined so they share the same PROPERTY_DEFINITION, PROPERTY_DEFINITION_REPRESENTATION and REPRESENTATION. The resulting structure is illustrated in Figure 11 below.

Validation properties can be combined in this way, if they are:

- of the same type (PROPERTY_DEFINITION.NAME)
- attached to the same model element (PROPERTY_DEFINITION.DEFINITION)

In this case, the REPRESENTATION.NAME has to be empty. The individual validation properties will be distinguished by their respective REPRESENTATION_ITEM.NAME. Since each validation property is instantiated at most once per model element, this does not introduce any ambiguities.

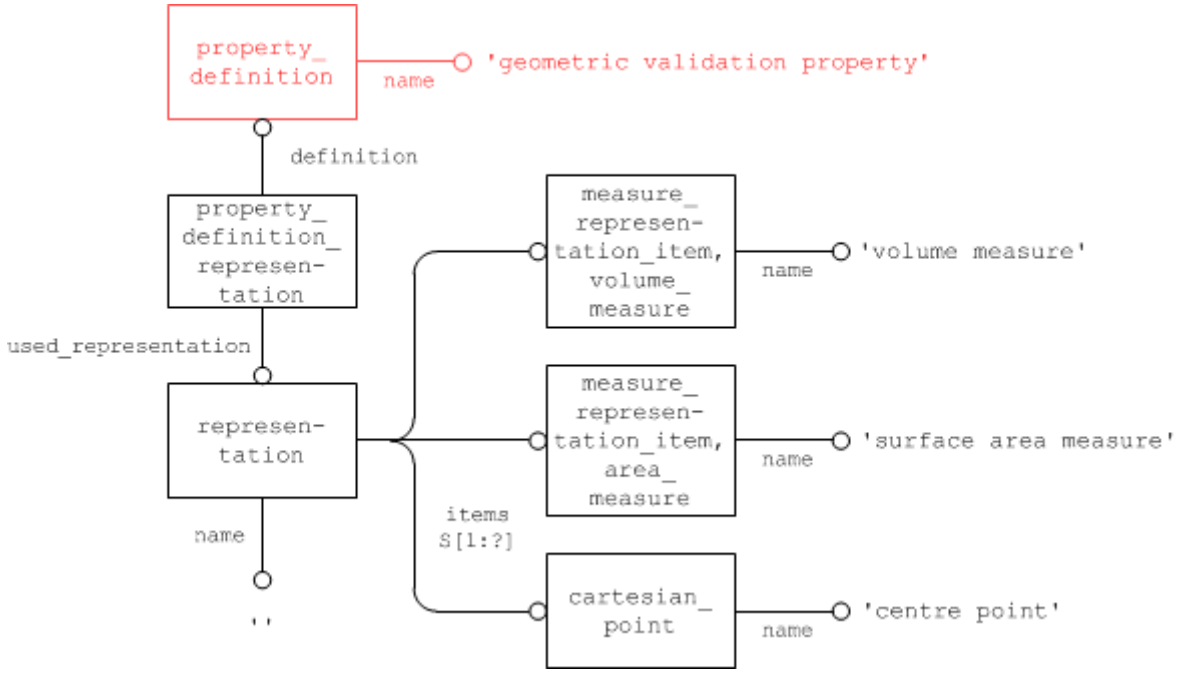

Figure 11: Combining volume, area and centroid in one property definition

Part 21 Example:

```
#233=MEASURE_REPRESENTATION_ITEM('volume measure',
VOLUME_MEASURE(664.38055098),#226);
#234=REPRESENTATION('',(#233,#237,#241),#219);
#235=PROPERTY_DEFINITION('geometric validation property', 'volume of
#217',#232);
#236=PROPERTY_DEFINITION_REPRESENTATION(#235,#234);
#237=MEASURE_REPRESENTATION_ITEM('surface area measure',
AREA_MEASURE(747.16814693),#229);
#241=CARTESIAN_POINT('centre point',(0.0,0.0,1.5));
```

Note: The only geometric validation properties that cannot be merged this way are the COPS (see section 0 below), since they use the `REPRESENTATION.NAME` to distinguish between the different types of sampling points.

Aside from this exception, this optimization can be applied similarly to all types of validation properties; i.e. it is sufficient to instantiate only one `PROPERTY_DEFINITION` ... `REPRESENTATION` per model element for each type:

- 'geometric validation property'
- 'assembly validation property'
- 'tessellated validation property'
- 'attribute validation property'
- 'pmi validation property'
- 'composite validation property'

More types might be added when specified.

4.13 Evaluation of the Geometric Validation Properties

While the basic mechanism is the same for all validation properties, their evaluation, and which thresholds are used to determine if the respective exchange was a successful, depend on user and business process requirements. In all cases, the result is obtained by comparing the value calculated in the target system from the imported model with the corresponding value provided in the STEP file as a validation property. The CAx-IF typically uses a threefold result system:

- "green" result: Successful exchange
- "yellow" result: Problems during exchange, but resulting model in target system may still be usable for some applications
- "red" result: Exchange failed.

4.13.1 Volume, Area, Centroid

For Volume, Area and Curve length validation, the thresholds usually applied are:

- "green" result: less than 1% deviation
- "yellow" result: between 1% and 10% deviation
- "red" result: more than 10% deviation

4.13.2 Centroid

For Centroid validation, there are two possible ways for evaluation. The basic method is by calculating the absolute three-dimensional distance (in millimeters) between the Centroid in the target system and the Centroid as transferred by the validation property:

- "green" result: less than 1mm deviation
- "yellow" result: between 1mm and 5mm deviation
- "red" result: more than 5mm deviation

However, this does not consider the actual model size. User feedback showed that while precise placement is required for small parts, larger tolerances may be acceptable for bigger parts. Therefore, the absolute centroid deviation will be scaled (i.e. divided) by the model size, which is defined as the length of the space diagonal of the three-dimensional bounding box enclosing the entire model (see section 0 above). This also eliminates the need for unit conversion.

- "green" result: less than 0.1% deviation
- "yellow" result: between 0.1% and 1% deviation
- "red" result: more than 1% deviation

There is one limitation to this evaluation method: it does not work for small parts. Assuming a model accuracy of 0.02mm, the minimum model size is 0.02mm / 0.1% = 20mm. Parts with a space diagonal below 20mm must be checked with the absolute value, as the scaled Centroid deviation is no longer meaningful due to rounding errors caused by the model accuracy.

4.13.3 Bounding Box

Similar to the Centroid, an absolute and a relative error can be calculated for the Bounding Box. The recommended procedure is as follows, using the same thresholds as given above:

- Calculate the absolute errors for the two corner points as defined in section 0: Error[$Min_{(x,y,z)}$] and Error[$Max_{(x,y,z)}$]
- Define the absolute error as the maximum between Error[$Min_{(x,y,z)}$] and Error[$Max_{(x,y,z)}$]
- Define the relative error as a ratio between the absolute error and the length of the diagonal of the bounding box, i.e. the distance between $Min_{(x,y,z)}$ and $Max_{(x,y,z)}$.

The recommendation is to always evaluate the absolute error, as this allows detecting not only size errors, but also position errors. The relative error can be evaluated in addition.

5 Extended Validation Properties

A shortcoming of the (basic) Geometric Validation Properties is that it fails to identify the component at fault when a correct geometric exchange has taken place, but the component has been incorrectly positioned within an assembly.

In order to avoid this, the Extended Validation Properties mechanism was agreed upon. This provides centroid information for an occurrence of a component or sub-assembly in the context of its parent, i.e. the centroid it would have in the parent part, if correctly positioned. This information allows the post-processor to traverse up the tree from the leaf parts, determining if they have been correctly positioned in their respective parents, and so identify where, if any, positional errors have occurred.

The extended validation properties require the instantiation of “centroid” validation properties, which are attached to the assembly definition in the following manner:

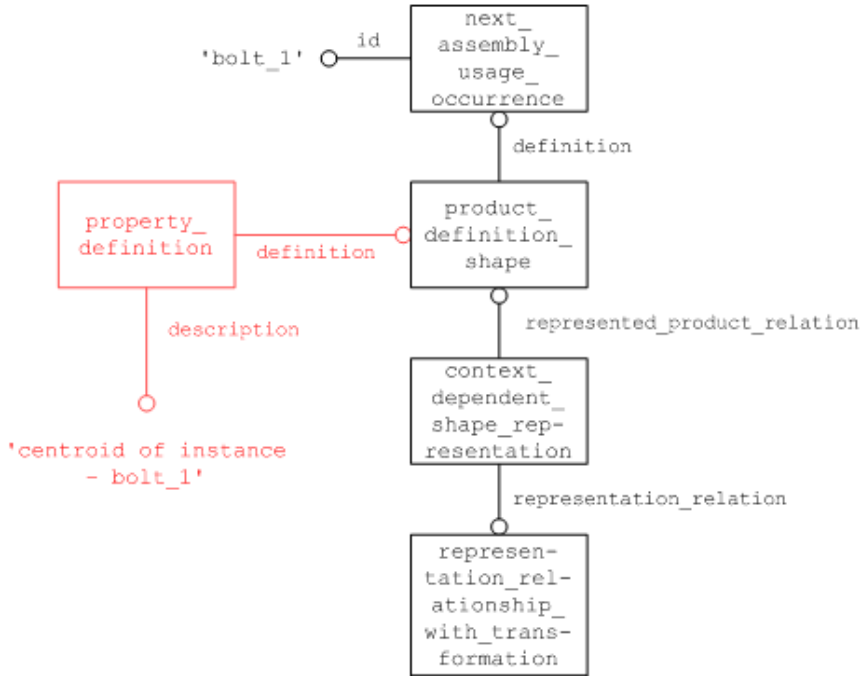

Figure 12: Extended Validation Property

Part 21 Example:

```
#419=(REPRESENTATION_RELATIONSHIP('#419','part44_nutbolt : part44_bolt',
#371,#417)REPRESENTATION_RELATIONSHIP_WITH_TRANSFORMATION(#411)
#972=NEXT_ASSEMBLY_USAGE_OCCURRENCE('PART44_BOLT','','PART44_BOLT',#33,
#27,'PART44_BOLT');
#973=PRODUCT_DEFINITION_SHAPE('PART44_BOLT',$,#972);
#974=CONTEXT_DEPENDENT_SHAPE_REPRESENTATION(#419,#973);
#977=PROPERTY_DEFINITION('geometric validation property', 'centroid of
instance - PART44_BOLT',#973);
```

Depending on which classes of geometry are contained in the subordinate sub-assemblies and leaf node parts, up to four centroids can be attached to the `property_definition` shown above:

- At least one solid → Solid Centroid (4.7.3)
- At least one independent surface → Independent Surface Centroid (4.8.2)
- At least one independent curve → Independent Curve Centroid (4.9.2)
- At least one independent point → Independent Points Centroid (4.10.2)

This provides the same level of validation at the assembly as is given at the part level. However, this also creates redundancy, because if a component gets misplaced during an exchange, then all (up to) four centroids will be affected in the same way, and any one centroid will be sufficient to trace the error.

Therefore, an alternative approach to using the Extended Validation Properties as described here is to use the full set of validation properties at the part level as defined in sections 4.6 – 4.10, in combination with the Assembly Validation Properties as defined in section 7 below. The “notional solid” introduced in section 7.2 will validate the correct positioning of the components independent of which classes (solid, surface, curve, point) and types (B-Rep, tessellated) of geometry are contained. The “number of children” (7.1) ensures the completeness of the assembly. The combined use of GVP and AVP is deemed to be more general and more efficient than the EVP.

6 Cloud Of PointS (COPS) Validation Properties

The “Cloud Of PointS” checking mechanism gives a set of sampling points which lie upon the face within the originating system. By checking the deviation of these points from the resulting surface, post STEP translation, a measure of confidence can be achieved that the face has not deviated from its original position or shape by an unacceptable amount.

6.1 Requirements and Distribution

To ensure reliable and meaningful results, the sampling points defined for this type of validation need to comply with the following requirements:

- Accuracy: Each point must be evaluated by the native system precisely on the surface or curve, as explained below.
- Active region: Each point must be inside or on the boundary of the corresponding face or edge, as explained below.
- Coverage: Sampling points must be evaluated inside a face surface and on its boundary.

A good balance needs to be found for the number of sampling points. Too many points drastically increase the STEP file size, while on the other hand too few points decrease the sensitivity of the mechanism. Hence, the following distribution parameters should be considered:

- Minimum number of points per face: Ensures adequate coverage of tiny faces (which are most likely to change during translation).
- Maximum spacing between points: Ensures reasonable distribution. This can be specified relative to model extents to enable the same parameter to be used for parts of different sizes.
- Chordal deviation tolerance: Ensures proper coverage of faces with high curvature (which are more likely to change during translation).

There is no “best” algorithm to distribute the sampling points, as “best” depends on a variety of parameters. However, experience shows that algorithms used to create FEM meshes for computer-based simulations render good distributions for the COPS points.

6.2 Validation Guidelines

For the evaluation of the cloud of points, two different classes of sampling points have to be considered:

- Smooth points must be evaluated on the surface and lie inside the domain of the face or on a smooth boundary.
- Sharp points must be evaluated on the edge curve within the active region of the sharp edge.

Each distinct section of the boundary of a face, corresponding to each edge entity, must be classified as either smooth or sharp and only have sampling points assigned to either the smooth or the sharp set, not both.

The following recommendations are given for the evaluation itself:

Smooth points must be evaluated on the surface and lie inside the domain of the face or on a smooth boundary. Edge curves as well as parameter space curves may be used to calculate smooth boundary points, but either way the points must satisfy this requirement, i.e. smooth edge curve points must be projected into the active region of the face surface and parameter space curves satisfy this requirement by definition.

Smooth points must be projected to the nearest active face surface region in the target model. This allows “arbitrary” edges to move along the smooth profile of the model’s geometry, or be eliminated altogether, in order to satisfy the topology requirements of the target system.

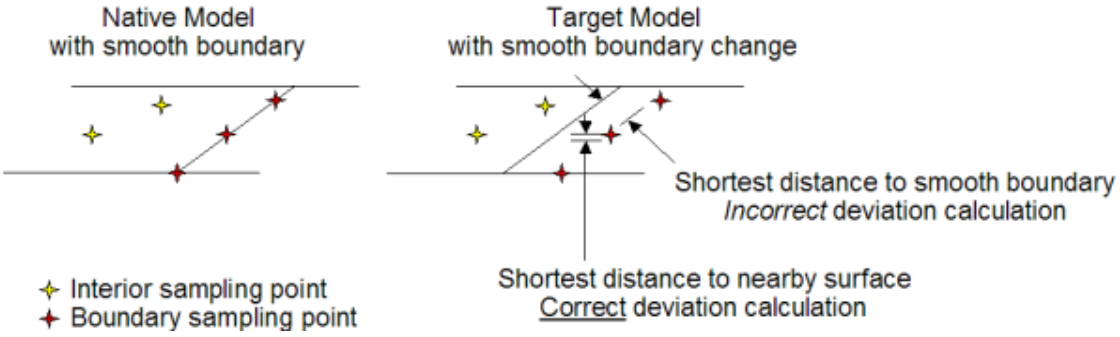

Figure 13: COPS – Smooth Sampling Points

Sharp points must be evaluated on the edge curve within the active region of the sharp edge. An edge is defined as sharp if it is non-manifold (not connected to only two faces) or if it is manifold and any surface normal angle calculations within the active region of the edge are non-tangent.

Although different recommendations exist concerning the maximum threshold for the angle, a 1.0 degree limit seems to be practical. The risk in defining this threshold too low is that edges will be classified as sharp which the target system would consider smooth and false negative sharp point deviations will be reported on edges which the target user feels are arbitrary and should be allowed to move along the smooth profile of the model’s geometry.

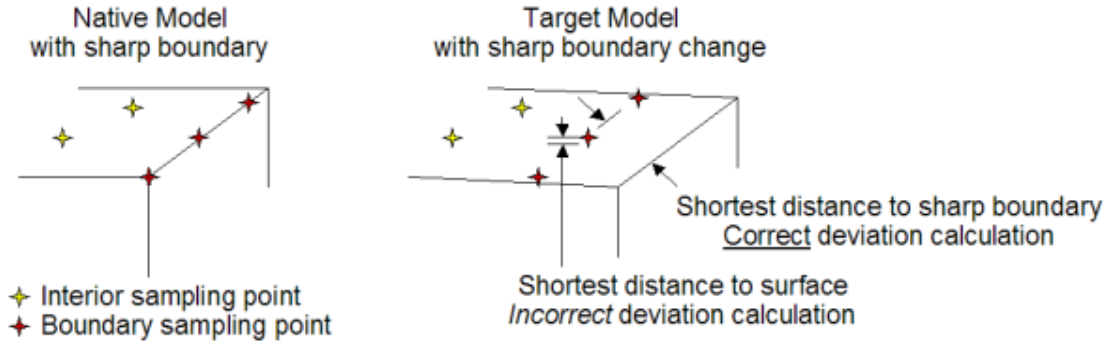

Figure 14: COPS – Sharp Sampling Points

Sharp points must be projected to the nearest active sharp edge curve region in the target model. This precisely identifies any deviations in the “real” edges that often have a direct relationship to the design intent of the model.

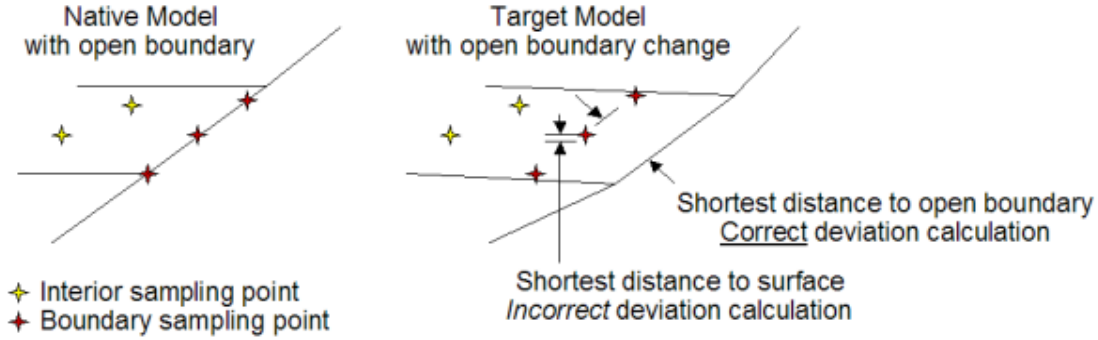

Figure 15: COPS – Sampling Points for an Open Boundary

These recommendations are internally consistent: smooth points are evaluated on surfaces and projected to surfaces; sharp points are evaluated on curves and projected to curves.

6.3 Instantiation

Figure 16 in the next page illustrates the relevant entities and their mandatory attributes used in the assignment of “Cloud Of PointS” for validation. It also displays the link between the sampling points and the face they are defined for.

All `CARTESIAN_POINTS` of the same class of sampling points (smooth / sharp) for a face shall be gathered in the same `REPRESENTATION`. Those in turn shall be linked through the same `PROPERTY_DEFINITION` to the `SHAPE_ASPECT` identifying the face to be validated.

Note that since the `REPRESENTATION.NAME` is used here to distinguish between the two classes of sampling points, COPS cannot be combined with other geometric validation properties as described in section 0 above.

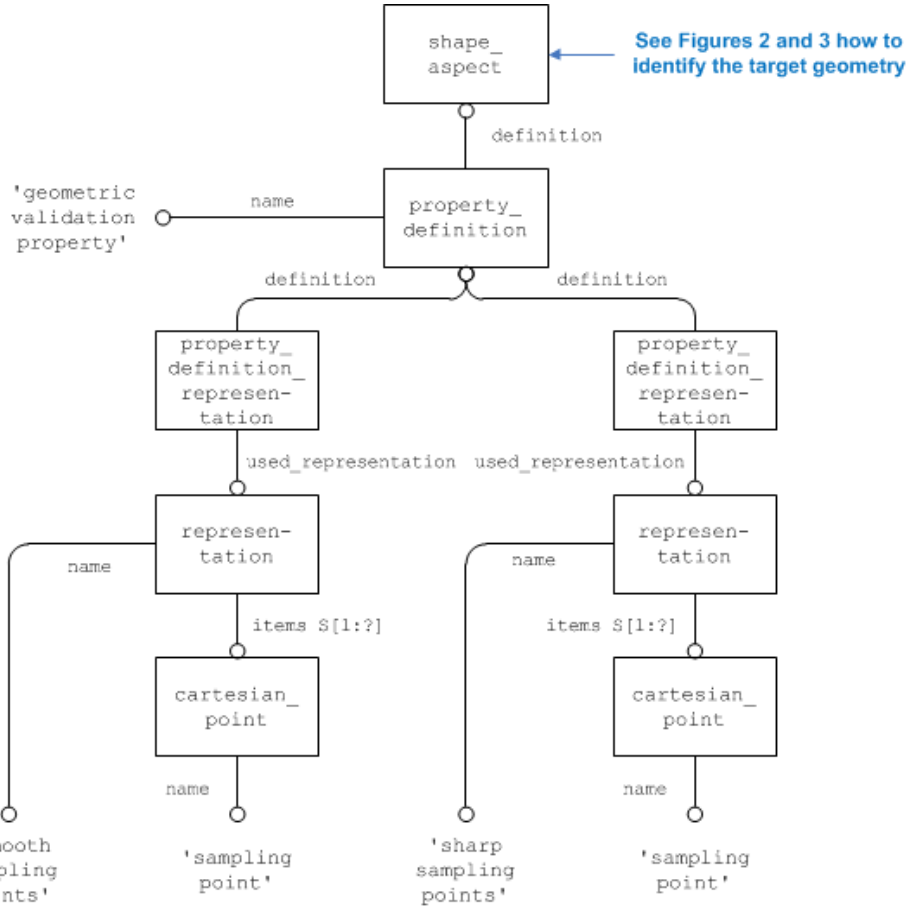

Figure 16: Cloud Of PointS Validation Property

Part 21 Example:

```
#251=ADVANCED_FACE('',(#236,#239,#242,#245),#250,.T.);
#471=CARTESIAN_POINT('#471',(13.33333333,10.,1.03163017));
#937=SHAPE_ASPECT('#937','aspect of #251',#419,.F.);
#938=REPRESENTATION('smooth sampling points',(#471,#472,#473,#474,#475);
#939=PROPERTY_DEFINITION('geometric validation property', 'surface
validation data for #251',#937);
#940=PROPERTY_DEFINITION_REPRESENTATION(#939,#938);
#942=PROPERTY_DEFINITION('','shape for validation properties',#937);
#943=SHAPE_DEFINITION_REPRESENTATION(#942,#941);
#941=SHAPE_REPRESENTATION('',(#251),#407);
```

7 Assembly Validation Properties

The extended validation properties for assembly data as defined in section 5 provides a suitable mechanism where geometrical data is provided along with the product structure. However, for incremental exchange of STEP files, where the product structure is included but the geometry of the component parts may be omitted, this is not applicable. Similarly, if the product structure is broken down into smaller sub-assemblies (“nested” external references), the component data is not available in the sub-assembly files, so again this is not applicable.

The additional Assembly Validation Properties introduced in this section will provide a verification capability where geometry is not present. This will make it possible for the exchange of assembly data to be verified in two ways. The first will ensure that the number of instances found at each node is correct. The second will ensure that the position and orientation information for each instance is correct.

The following two concepts are defined:

Number of Children

The first of these allows the pattern of the product structure to be verified, i.e. each node has the correct number of instances or branches.

Notional Solids Centroid Position

The second of these allows the positional information for each instance in the product structure to be verified, i.e. the coordinate systems for each child node is positioned and oriented correctly relative to its parent. Note that this condition is not mathematically guaranteed by this methodology, but the chance of an incorrect position and orientation combining to give the correct result is extremely small.

7.1 Number of Children

Each node which is the parent node of at least one instance will have a property attached to enumerate the actual number of child instances of that node. This number will be defined by the sending system. On receipt of the data, the system which post-processes the STEP file will check that the translated assembly data has the correct number of child instances at each node.

This property will be defined by STEP entities in the following way:

- Where a `PRODUCT_DEFINITION` entity is used as `RELATING_PRODUCT_DEFINITION` by one or more `NEXT_ASSEMBLY_USAGE_OCCURRENCE` entities, it will have a `PROPERTY_DEFINITION` for which the name will be “assembly validation property”.
- The `REPRESENTATION` linked to the `PROPERTY_DEFINITION` by a `PROPERTY_DEFINITION_REPRESENTATION` will have the name “number of children”.
- The single `REPRESENTATION_ITEM` for this `REPRESENTATION` will have the name “number of children”. There are two ways of implementing this. Though the first way is valid in all APs, the second way preserves the integer type of this value:
  - In AP203e1 and AP214, it will be a `VALUE_REPRESENTATION_ITEM` with a `VALUE_COMPONENT` which is a `COUNT_MEASURE` (real number).
  - In AP203e2 and AP242, it will be an `INTEGER_REPRESENTATION_ITEM` where `THE_VALUE` is an integer value.
- The value defined will represent the number of `NEXT_ASSEMBLY_USAGE_OCCURRENCE` entities for which the `RELATING_PRODUCT_DEFINITION` is the `PRODUCT_DEFINITION` for which the property is defined, as illustrated below:

Note: The value of INTEGER_REPRESENTATION_ITEM has to be written as a REAL number, i.e. with trailing decimal point. See explanation in section 4.10.1 for details.

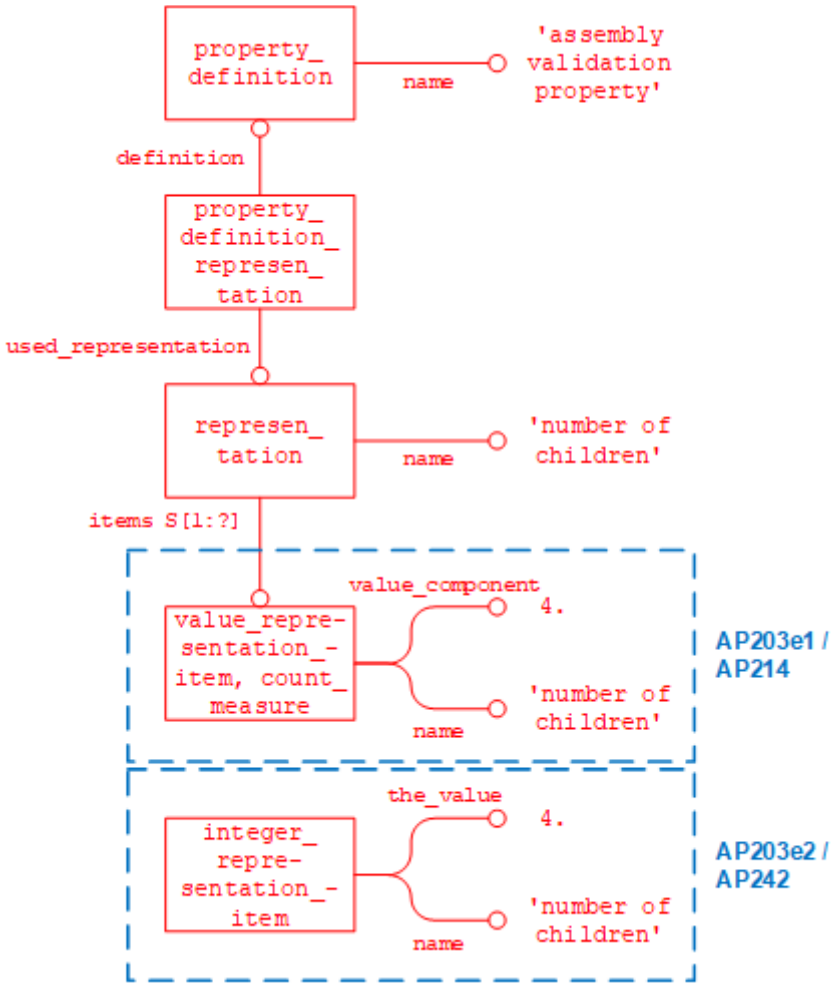

Figure 17: "Number of Children" Assembly Validation Property

Part 21 Example:

```
#10=PRODUCT('as1','as1',$,(#8));
#14=PRODUCT_DEFINITION_FORMATION('v0','v0 for as1',#10);
#15=PRODUCT_DEFINITION('design',$,#14,#9);
#333=PROPERTY_DEFINITION('assembly validation property','',#15);
#335=REPRESENTATION('number of children',(#334),#266);
#336=PROPERTY_DEFINITION_REPRESENTATION(#333,#335);

/* AP203e1 / AP214 */
#334=VALUE_REPRESENTATION_ITEM('number of children',COUNT_MEASURE(4.0));
/* AP203e2 / AP242 */
#334=INTEGER_REPRESENTATION_ITEM('number of children',4.);
```

Where the following instance definitions occur within the STEP file which have #15 as the RELATING_PRODUCT_DEFINITION:

```
#294=NEXT_ASSEMBLY_USAGE_OCCURRENCE('PART44_PLATE','','PART44_PLATE',#15,#2
1, 'PART44_PLATE');
#315=NEXT_ASSEMBLY_USAGE_OCCURRENCE('PART44_LBRACKASS_1','',
'PART44_LBRACKASS',#15,#51,'PART44_LBRACKASS');
#318=NEXT_ASSEMBLY_USAGE_OCCURRENCE('PART44_LBRACKASS_2','',
'PART44_LBRACKASS',#15,#51,'PART44_LBRACKASS');
#330=NEXT_ASSEMBLY_USAGE_OCCURRENCE('PART44_RODASS','','PART44_RODASS',#15,
#63,'PART44_RODASS');
```

7.2 Notional Solids Centroid Position

This property is similar to the geometric validation property “centroid” (cp. section 0) in that a property representing a location is defined for each sub-assembly. However, in this case the property is not calculated based on any real geometry defined for that product.

For the top node and each intermediate node of a product structure, a notional solid is assumed within the child node of each child instance of that node. Using the positional and orientation relationship for each child instance, a centroid position can be calculated for the combined set of notional solids within the set of child instances.

The notional solid will be a cube of size 10 x 10 x 10. The notional solid will be positioned with its centroid at (10.0, 10.0, 10.0) of the coordinate system of the child node. Note that the actual size and shape of the notional solid will not, in fact, affect the overall result. The key data is the centroid position and the assumption that the volume of the notional solid in each child node is the same. Mathematically, the calculated point is the mean of the set of points at (10.0, 10.0, 10.0) within the child nodes.

Note that in contrast to an actual solid, the notional solid itself is not in the STEP file – it is just a convention. Thus, it has to be ensured that the correct geometrical context is used for the notional solids centroid position, in order to guarantee that the units are applied correctly.

The child node may be a leaf node of the overall assembly or another intermediate node within the sub-assembly. Each case is treated in the same way. Even though the child node might be an intermediate node with no actual geometry defined, a notional solid will be assumed for the purpose of this calculation.

The notional centroid for each sub-assembly is influenced only by the notional solid in each of its directly instanced child nodes.

This property will be defined by STEP entities in the following way:

- Where a `PRODUCT_DEFINITION` entity is used as the `RELATING_PRODUCT_DEFINITION` by one or more `NEXT_ASSEMBLY_USAGE_OCCURRENCE` entities, it will have a `PROPERTY_DEFINITION` for which the name will be “assembly validation property”.
- The `PROPERTY_DEFINITION` description will be “notional solids centroid”.
- The `REPRESENTATION` linked to the `PROPERTY_DEFINITION` by a `PROPERTY_DEFINITION_REPRESENTATION` will have the name “notional solids centroid”.
- The single `REPRESENTATION_ITEM` for this `REPRESENTATION` will have the name “centre point”. It will be a `CARTESIAN_POINT` defining the calculated centroid for the notional solids assumed for each child node. The child nodes are those `PRODUCT_DEFINITIONS` defined as a `RELATED_PRODUCT_DEFINITION` in a `NEXT_ASSEMBLY_USAGE_OCCURRENCE` entity for which the `RELATING_PRODUCT_DEFINITION` is the `PRODUCT_DEFINITION` for which the property is defined.

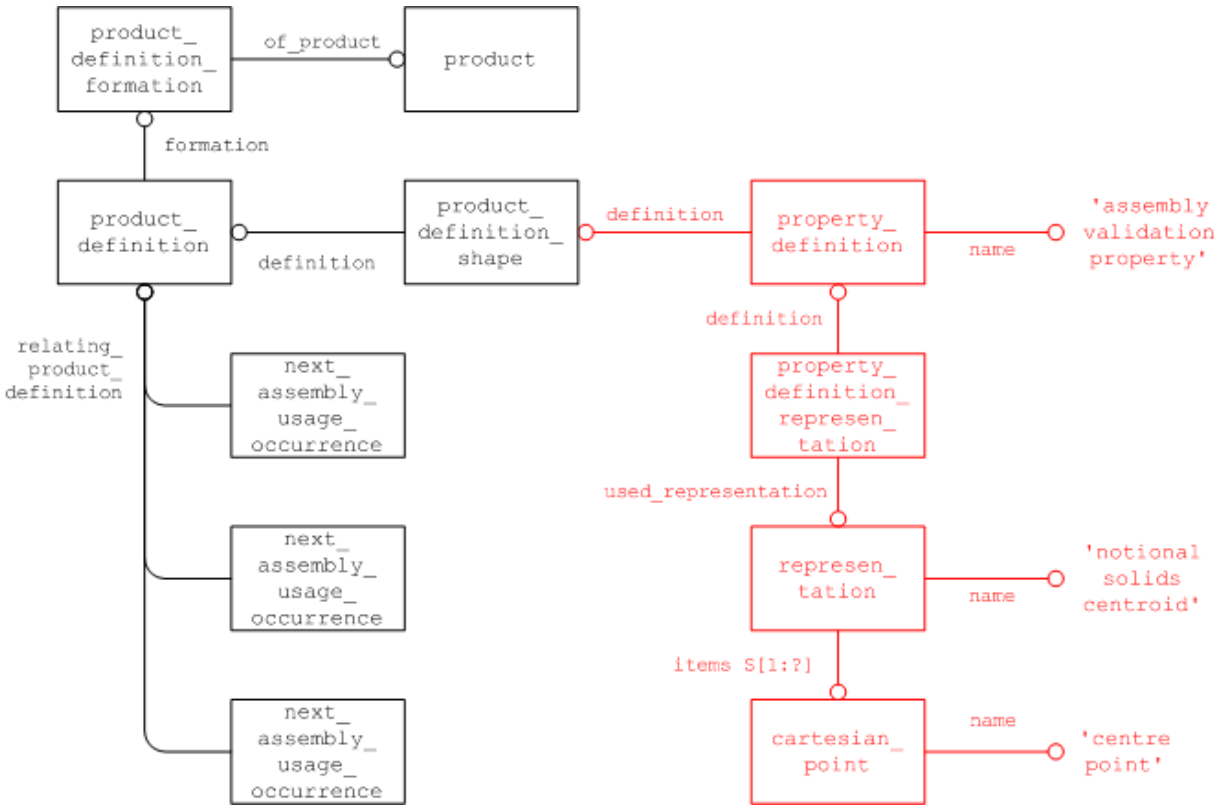

Figure 18: "Notional Solids Centroid Position" Validation Property

Part 21 Example:

```
#40=PRODUCT('part44_nutbolt','part44_nutbolt',$,(#8));
#44=PRODUCT_DEFINITION_FORMATION('v0','v0 for part44_nutbolt',#40);
#45=PRODUCT_DEFINITION('design',$,#44,#9);
#286=PRODUCT_DEFINITION_SHAPE('',$,#45);
#345=PROPERTY_DEFINITION('assembly validation property',
'notional solids centroid',#286);
#346=CARTESIAN_POINT('centre point',(10.,10.,12.));
#347=REPRESENTATION('notional solids centroid',(#346),#172);
#348=PROPERTY_DEFINITION_REPRESENTATION(#345,#347);
```

Where the following instance definitions occur within the STEP file which have #45 as the RELATING_PRODUCT_DEFINITION:

```
#300=NEXT_ASSEMBLY_USAGE_OCCURRENCE('PART44_BOLT','','PART44_BOLT',#45,
#33,'PART44_BOLT');
#303=NEXT_ASSEMBLY_USAGE_OCCURRENCE('PART44_NUT_1','','PART44_NUT',#45,
#39,'PART44_NUT');
```

8 Summary of Imposed Attribute Values

The following constraints on attribute values are imposed by this recommended practice:

Validation Property	**Applicable Geometry Class**	**PROPERTY_DEFINITION .NAME**	**REPRESENTATION .NAME** (*)	**REPRESENTATION_ITEM *Subtype***	**REPRESENTATION_ITEM.NAME**
Total Volume	Solids	'geometric validation property'	'' / 'volume'	VOLUME_MEASURE	'volume measure'
Surface Area (**)	Solids	'geometric validation property'	'' / 'surface area'	AREA_MEASURE	'surface area measure'
Centroid	Solids	'geometric validation property'	'' / 'centroid'	CARTESIAN_POINT	'centre point'
Independent Surface Area	Surfaces	'geometric validation property'	'' / 'independent surface area'	AREA_MEASURE	'independent surface area measure'
Independent Surface Centroid	Surfaces	'geometric validation property'	'' / 'independent surface centroid'	CARTESIAN_POINT	'surface centre point'
Independent Curve Length	Curves	'geometric validation property'	'' / 'independent curve length'	LENGTH_MEASURE	'curve length measure'
Independent Curve Centroid	Curves	'geometric validation property'	'' / 'independent curve centroid'	CARTESIAN_POINT	'curve centre point'
Number of Independent Points	Points	'geometric validation property'	'' / 'number of independent points'	VALUE_REPRES._ITEM (COUNT_MEASURE) *(AP203e1 / AP214)* INTEGER_REPRES._ITEM *(AP203e2 / AP242)*	'number of independent points'
Independent Points Centroid	Points	'geometric validation property'	'' / 'independent points centroid'	CARTESIAN_POINT	'independent points centre point'
Bounding Box	Solids, Surfaces, Curves	'geometric validation property'	'' / 'bounding box'	CARTESIAN_POINT *(2x)*	'bounding box corner point'
COPS	Solids, Surfaces	'geometric validation property'	'smooth sampling points', 'sharp sampling points'	CARTESIAN_POINT *(many)*	'sampling point'

Validation Property	Applicable Geometry Class	PROPERTY_DEFINITION .NAME	REPRESENTATION .NAME(*)	REPRESENTATION_ITEM *Subtype*	REPRESENTATION_ITEM.NAME
Number of Children	-/-	'assembly validation property'	'' / 'number of children'	VALUE_REPRES._ITEM (COUNT_MEASURE) *(AP203e1 / AP214)* INTEGER_REPRES._ITEM *(AP203e2 / AP242)*	'number of children'
Notional Solids Centroid Position	-/-	'assembly validation property'	'' / 'notional solids centroid'	CARTESIAN_POINT	'centre point'

Figure 19: Table of Imposed Attribute Values

Explanation of footnotes for Figure 19:

(*): If several validation properties of the same type for the same model element are combined into a single property as defined in section 0, the REPRESENTATION.NAME will be an empty string. That means:

- An empty REPRESENTATION.NAME string means that several properties can be expected in the REPRESENTATION.
- A populated REPRESENTATION.NAME means that the representation includes only value for the stated validation property.
- The only exception to this rule are COPS, because for these, the REPRESENTATION.NAME is used to distinguish between 'smooth' and 'sharp' sampling points; i.e. COPS cannot be combined with other geometric validation properties.

(**): For CATIA-based systems, which calculate "wetted" surface instead of total surface (i.e. voids are not taken into account), the REPRESENTATION_ITEM.NAME shall be 'wetted area measure' to avoid reporting of false errors

Annex A Availability of implementation schemas

A.1 AP214

The AP214 schemas support the implementation of the capabilities as described. The schemas can be retrieved from:

- IS Version (2001) – http://www.cax-if.de/documents/ap214_is_schema.zip
- 3rd Edition (2010) – http://www.cax-if.de/documents/AP214E3_2010.zip

A.2 AP203 2nd Edition

The long form EXPRESS schema for the second edition of AP203 can be retrieved from:

- http://www.cax-if.de/documents/part403ts_wg3n2635mim_lf.exp

Note that the first edition of AP203 is no longer support in the Recommended Practices.

A.3 AP242

The long form EXPRESS schema for the first edition of AP242 can be retrieved from:

- http://www.cax-if.de/documents/ap242_is_mim_lf_v1.36.zip