JT-IF Implementation Guidelines
for
JT Validation Properties

Version 1.0, 21 June 2021

Status: Release

Contacts:

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

Sarah Giese
PROSTEP AG
Dolivostraße 11
64293 Darmstadt / Germany
sarah.giese@prostep.com

Felix Klier
PROSTEP AG
Dolivostraße 11
64293 Darmstadt / Germany
felix.klier@prostep.com

Contents

Table of Contents

Figures

Tables

Document History

Version	Date	Description
1.0	2021-06-21	Initial Public Release

1 Introduction

This document specifies implementation guidelines for the exchange of validation properties with JT. It makes use of existing default properties as well as newly introduced properties to capture certain key aspects of the model that allow for validating the success of the data transfer after importing the JT file into a target application.

Validation properties have become a well-established mechanism to ensure data transfer quality in a variety of exchange formats. This document reuses existing concepts were applicable for consistency, so that applications handling multiple formats can apply these approaches in a universal manner.

The first revision of the implementation guidelines for JT Validation Properties covers geometric validation properties as well as validation properties for Product & Manufacturing Information (PMI). This scope might be extended in the future, depending on user requests.

2 Scope

2.1 In Scope

The following are within scope of this document:

- Definition of the concept for validation properties
- Default JT properties to be used as validation properties
- General overview of property types and where they can be defined
- Geometric Validation Properties
- PMI Validation Properties

2.2 Out of Scope

The following are out of scope of this document as they are covered elsewhere:

- Definition of any other types of properties, such user-defined properties.
- Definition of geometry
- Definition of PMI

The following are out of scope for this document at the moment, because the underlying use cases and requirements have not yet been fully described:

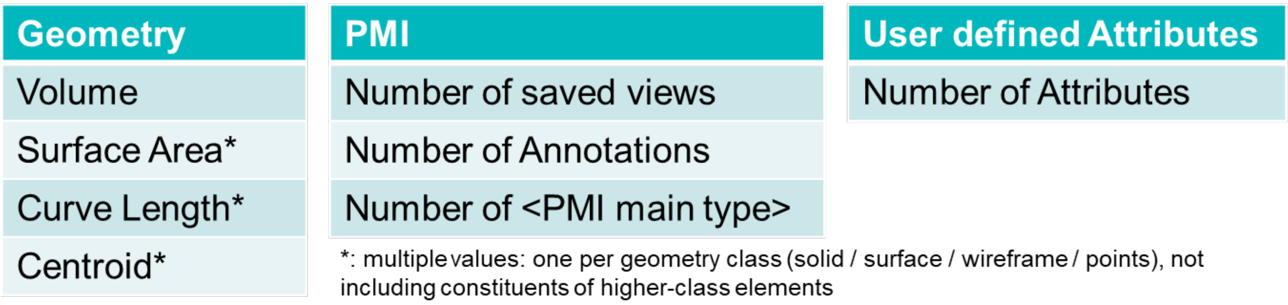
- Assembly-level validation properties
- Validation Properties for aspects of the model other than geometry or PMI.

3 Fundamentals & Concepts

3.1 What are Validation Properties?

Validation Properties are a type of meta data, as they provide information about the model they are derived from, inside the model. They are key characteristics of a model, which are deemed important for the respective use case and thus shall not be modified during translation and exchange. They serve as a kind of checksum and help to validate the level of success of importing a JT file without the use of additional third-party tools.

Typical Validation Properties are:



Geometry	PMI	User defined Attributes
Volume	Number of saved views	Number of Attributes
Surface Area*	Number of Annotations	
Curve Length*	Number of <PMI main type>	
Centroid*		

*: multiple values: one per geometry class (solid / surface / wireframe / points), not including constituents of higher-class elements

Figure 1: Typical Validation Properties

Validation Properties either carry the results of calculations performed inside the exporting systems which can be repeated in the target system based on the imported data, such as volume or surface area for instance. Another widely used type of validation properties are element counts, such as “Number of Saved Views”. The comparison of the results provides the desired feedback.

While validation properties defined following this approach do not provide any immediate benefits on export to JT, they show their full potential during import of JT files into a target application, where they allow to assess the success of the data transfer without requiring access to the originating authoring tool or the source CAD data. This makes them a valuable tool in cross-company exchange scenarios or in the context of long-term archiving.

In order to validate the export to JT, the general recommendation is using third-party check tools, which allow comparing the native CAD data with the derived JT data and in the same step often double-check the included validation property values. The validation properties defined in this document can serve as quality criteria for this export validation.

3.2 How do Validation Properties work?

The workflow for including and evaluating validation properties in a JT file is depicted by Figure 2:

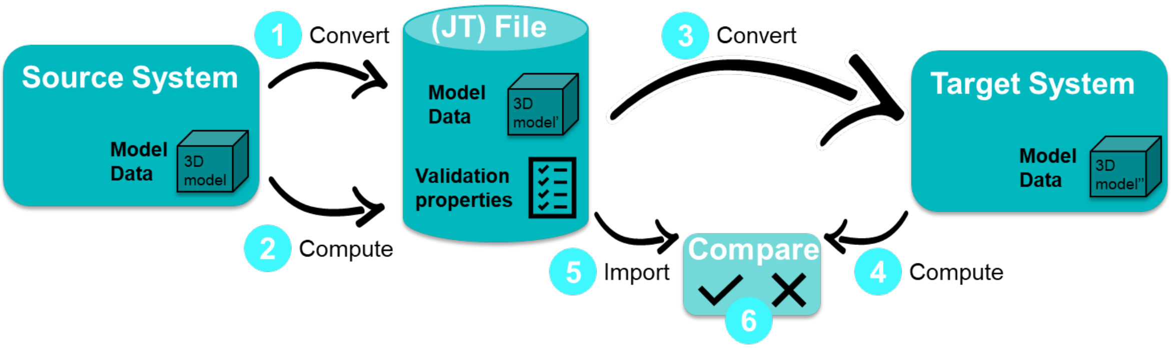


Figure 2: Concept of Validation Properties

The process follows these steps:

1. Source system converts the model to JT
2. Source system computes the validation properties and includes the respective results in the JT file
3. Target system imports the JT file and creates model from imported data
4. Target system computes validation properties from generated model
5. Target system reads validation properties from the JT file
6. Target system compares read and computed values for validation properties to determine success.

Depending on the type of validation property, the comparison is successful if the two values (read from JT and computed by target system) are within a certain tolerance for computed properties (e.g. less than 1% volume deviation), or if the two values match exactly for count properties (e.g. number of datum feature symbols).

Note that the "source system" can either be a CAD system with its built-in JT translator, or the combination of a CAD system and a third-party JT translator which calculates the validation properties using the CAD system's API. The same applies to the "target system", where the type of the target application is not limited to CAD. Many validation tools are also capable of evaluating the validation properties.

3.3 Evaluation and Reporting of Validation Properties

As described in the process above, validation properties are relevant mainly during import into a target application. To increase their level of trust, they can be double-checked during export as well.

Export Recommendations

- Validation Properties shall be included in JT files whenever supported by the exporting authoring tool and JT translator. The impact on file size and performance from adding validation properties is generally deemed negligible compared to benefits gained in process security.
- A validation of the validation properties can be done during export by using a third-party checking tool that compares the original CAD model to the derived JT file, and in that process, can perform the same calculations as in step 2 of the process given in section 3.2 above.
  - This will double-check the validation property values in the JT file.
  - The results of this check shall be included in the log file, e.g., by listing the validation property values read from the JT file and comparing them with the corresponding values calculated by the checking tool.
  - Ideally, the checking tool is provided by a different software vendor than the JT translator, to avoid that a translation error is missed due to the same error being applied during evaluations.
- Validation Properties for element types that do not occur in the model (e.g., a specific type of PMI) shall be included with a value of '0', to explicitly state this element type does not occur.
- The level of validation properties included, and the strictness of their validation during export, is a business decision with the company creating the data.

Import Recommendations

- Whenever validation properties are found in a JT file, they shall be evaluated upon import, as described in steps 4 through 6 of the process given above.
- The import log file created by the JT translator shall list for each validation property:
  - The value read from the JT file
  - The value computed in the target system
  - The deviation between these two values
  - A derived "OK" / "KO" assessment.
- For "count" validation properties ("Number of…"), an exact match is always required, while for calculated properties, thresholds are defined depending on company rules and use cases. Typical thresholds e.g. for volume deviation are 1% or 0.1%.

3.4 Storing Properties in JT

Properties in JT are stored via “Property Atom Elements”. “Property Atom Elements” are not nodes or attributes themselves but can be associated with any node or attribute. An individual property is specified as a *key/value* “Property Atom Element” pair, where the *key* identifies the type and meaning of the *value*.

Refer to the JT File Specification for details; see Annex A.

3.5 Property Types in JT

3.5.1 Property Types used by Validation Properties

String Property Atom Element

- String Property Atom Element represents a character string property atom.
- **Note** that String elements can be used to encode other data types, such as 64-bit floating point numbers (F64) for instance, or three F64 numbers separated by spaces to represent the three-dimensional coordinates of a point.

Integer Property Atom Element

- Integer Property Atom Element represents a property atom whose value is of I32 data type.

3.5.2 Other Property Types

JT does support more property types than just String and Integer, such as Floating Point, JT Object References, Dates, and other. These are, however, not required in the context of Validation Properties and thus not listed here. Check the JT File Format Specification for details.

3.6 Properties at the Geometry level (XT-BREP)

XT-BREP can store attributes on individual geometric elements. An attribute is an entity which contains data, and which can be attached to any other entity except attributes, fins, lists, transforms or attribute definitions.

An attribute has the following fields:

- **Definition:** An attribute definition is an entity which defines the number and type of the data fields in a specific type of attribute, which entities may have such an attribute attached, and what happens to the attribute when its owning entity is changed. XT data shall not contain duplicate attribute definitions. Each attribute of a given type should reference the same instance of the attribute definition for that type. It is incorrect, for example, to create a copy of an attribute definition for each instance of the attribute of that type. Only those attribute definitions referenced by attributes in the part occur in the data.
- **Owner:** The XT element the attribute belongs to
- **Fields:** These data fields consisting of one or more integers, doubles, vectors, etc.

In particular, attributes can be stored for every geometrical element inside a XT-BREP definition:

- Vertex
- Loop
- Shell
- Body
- Edge
- Face
- Region

The concept is already used for Moniker IDs and will be applied for geometric validation properties in a similar way.

4 Geometric Validation Properties (GVP)

Geometric Validation Properties (GVP) are intended to validate the shape of a part regarding completeness and position. A GVP mismatch indicates that there is an issue with the geometry in general; further investigations are most likely needed to pinpoint the problem exactly. A GVP match on the other hand provides a level of trust for the import results.

GVP shall always be defined at the part level. Depending on the use case, it is possible to define additional GVP at the geometry (XT-Brep) level to validate certain key elements of the model individually.

4.1 GVP for Solid and Surface Geometry

The following default JT properties can be used as validation properties for solid and surface geometry in a part, following the process introduced in section 3.2 above:

JT Property Key	Meaning	JT File Data Type	Valid Values
CAD_CENTER_OF_GRAVITY	Center of gravity of solids within part	String	3 * numeric
CAD_DENSITY	Density of solids within part	String	numeric
CAD_MASS	Mass or weight of solids within part	String	numeric
CAD_MOMENT_OF_INERTIA	The components of the Inertia Tensor of the part associated with the node, as calculated by the originating system.	String	6 * numeric
CAD_SURFACE_AREA	Surface area of solids and surfaces within part	String	numeric
CAD_VOLUME	Volume of solids within part	String	numeric

Table 1: Validation Properties for Solid and surface Geometry

Refer to the JT-IF Implementation Guidelines or the JT File Format Specification for details (see Annex A).

Notes:

- The properties listed in Figure 1 capture solid geometry as well as surface geometry, i.e., they also include surfaces that are not faces of a solid. Such ‘independent’ surfaces contribute to CAD_SURFACE_AREA only, since an individual surface has no volume, hence no mass, and thus doesn’t affect the center of gravity.
- To capture the volume, surface area, or center of gravity for individual bodies within a multi-body part, the respective properties must be attached to the corresponding XT-Brep ‘Body’ elements.
- For an efficient validation, using the trinity of CAD_VOLUME, CAD_SURFACE_AREA and CAD_CENTER_OF_GRAVITY is sufficient.
- It is recommended to include CAD_MOMENT_OF_INERTIA in the validation whenever available, as it provides an additional data point, in particular when materials are defined in the JT file.

4.2 GVP for Curve / Wireframe Geometry

Some CAD models contain ‘independent curves’, i.e. curves which are not edges of faces or solids. Such curves might e.g. represent the centerline of a cable or pipe. As stated before, it is worthwhile validating these elements separately from surface and solid geometry.

The following GVP values shall be used if the JT file contains curve or wireframe geometry:

JT Property Key	Meaning	JT File Data Type	Valid Values
GVP_INDEPENDENT_CURVE_LENGTH	The total length of all independent curves in the model	String	numeric
GVP_INDEPENDENT_CURVE_CENTROID	The combined center of geometry for all independent curves in the model	String	3* numeric

Table 2: Validation Properties for Curve / Wireframe Geometry

To capture the length or center of geometry for individual independent curves within a part, the respective properties must be attached to the corresponding XT-Brep ‘Edge’ or ‘Loop’ elements.

4.3 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.

The Bounding Box will be stored by providing its two opposing corner points [minX minY minZ] and [maxX maxY maxZ] per the algorithm defined in section 6.1:

JT Property Key	Meaning	JT File Data Type	Valid Values
GVP_BOUNDING_BOX	The two opposite corner points of the model bounding box per the agreed algorithm. A point should be three space separated numbers and the points should be semi-colon separated, i.e. “minX minY minZ; maxX maxY maxZ”	String	6* numeric

Table 3: Bounding Box Validation Property

These points 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).

5 PMI Validation Properties

While GVP provide the means to validate the shape of the part, PMI Validation Properties (PMI VP) aim at validating the completeness and correctness of the product and manufacturing information. These definitions are an integral part of model-based design and provide vital information about the design intent to downstream processes, thus creating detailed PMI VP is justified.

PMI VP can be applied at three different levels:

- At the Part level, to provide an overall summary
- At the Saved View level, to check subset of PMI contained in a particular Saved View
- At the PMI element level

'PMI element' will be used as a generic term in this document and refers to any kind of PMI data, such as dimensions, feature control frames, welding symbols, surface conditions or notes.

'Saved View' is the term defined by ISO 16792 and ASME Y14.5. In JT, usually the term 'Model View' is used. This document uses the terms 'Saved View', 'Model View' and 'view' synonymously.

PMI VP are either defined as count properties, providing the number of elements of a specific type in the Part or in a Saved View, or as calculated properties for certain elements.

5.1 Version Property

PMI definitions in JT evolve over time. This means that definitions get added and updated based on gathered experiences and enhanced user requirements. To ensure consistent evaluation of the validation property, the property PMI_FIDELITY_VERSION captures the version of the schema used to validate the fidelity of the PMI data in the JT file.

JT Property Key	Meaning	JT File Data Type	Valid Values
PMI_FIDELITY_VERSION	The validation version property identifies the schema used for the PMI fidelity properties. A PMI_FIDELITY_VERSION value of 3 is inclusive of definitions for 1, 2 and 3	Integer	1, 2 or 3

Table 4: PMI Fidelity Version

Note: To ensure full compatibility, this value shall always be set to 3.

Though it is unlikely to encounter JT files with a fidelity version of 1 or 2, it is recommended to gracefully reject such files as they will be incomplete in several regards.

There might be versions 4, 5 etc. in the future; if so, this document will be updated accordingly.

5.2 Part-level PMI Validation Properties

The part-level PMI VP provide an overall summary of the PMI elements defined in the entire part.

- The CAD_MV_COUNT property provides the number of Saved Views (called "Model Views" in JT) defined in the file.
- The CAD_PMI_COUNT validation property contains the total count of PMI entities belonging to the part as defined in the source CAD system.

The value of CAD_PMI_COUNT shall match the sum of all CAD_PMI_COUNT_<type> properties defined below.

Property Name	JT File Data Type	Valid Values
CAD_MV_COUNT	Integer	numeric
CAD_PMI_COUNT	Integer	numeric

Table 5: Part-Level PMI General Count Validation Properties

All further CAD_PMI_COUNT_<type> validation properties contain the total count of PMI elements of the specified type as defined in the source CAD system.

It is recommended that if a certain PMI element type does not occur in the model, to provide the corresponding CAD_PMI_COUNT_<type> property nonetheless, with a value of 0 (zero). This explicitly states that there are no elements of this kind in the file.

Property Name	JT File Data Type	Valid Values
CAD_PMI_COUNT_ATTRIBUTENOTE	Integer	numeric
CAD_PMI_COUNT_BALLOON	Integer	numeric
CAD_PMI_COUNT_BUNDLEDRESSINGNOTE	Integer	numeric
CAD_PMI_COUNT_CALLOUTDIMENSION	Integer	numeric
CAD_PMI_COUNT_CENTERLINE	Integer	numeric
CAD_PMI_COUNT_CHAMFER	Integer	numeric
CAD_PMI_COUNT_CIRCLECENTRE	Integer	numeric
CAD_PMI_COUNT_COORDINATENOTE	Integer	numeric
CAD_PMI_COUNT_CROSSHATCH	Integer	numeric
CAD_PMI_COUNT_CUTTINGPLANESYMBOL	Integer	numeric
CAD_PMI_COUNT_DATUMFEATURESYMBOL	Integer	numeric
CAD_PMI_COUNT_DATUMTARGET	Integer	numeric
CAD_PMI_COUNT_DIMENSION	Integer	numeric
CAD_PMI_COUNT_EMARKING	Integer	numeric
CAD_PMI_COUNT_FASTENER	Integer	numeric
CAD_PMI_COUNT_FEATURECONTROLFRAME	Integer	numeric
CAD_PMI_COUNT_FITDESIGNATION	Integer	numeric
CAD_PMI_COUNT_LINEWELD	Integer	numeric
CAD_PMI_COUNT_LOCATOR	Integer	numeric
CAD_PMI_COUNT_MATERIALSPECIFICATION	Integer	numeric
CAD_PMI_COUNT_MEASUREMENTPOINT	Integer	numeric
CAD_PMI_COUNT_NOTE	Integer	numeric

Property Name	JT File Data Type	Valid Values
CAD_PMI_COUNT_ORGANISATION	Integer	numeric
CAD_PMI_COUNT_PARTSPECIFICATION	Integer	numeric
CAD_PMI_COUNT_PROCESSSPECIFICATION	Integer	numeric
CAD_PMI_COUNT_REGION	Integer	numeric
CAD_PMI_COUNT_SPOTWELD	Integer	numeric
CAD_PMI_COUNT_SURFACEFINISH	Integer	numeric
CAD_PMI_COUNT_USERDEFINEDSYMBOL	Integer	numeric
CAD_PMI_COUNT_TABLE	Integer	numeric

Table 6: Part-Level PMI Type-Specific Count Validation Properties

5.3 View-level PMI Validation Properties

The view-level PMI VP provide an overall summary of the PMI elements defined for the given Saved View. Validation at this level is important since users generally don't view all PMI definitions at once, but selected manageable subsets, which are defined in views.

5.3.1 Count Properties

The CAD_PMI_COUNT validation property contains the total count of PMI entities contained in the Saved View as defined in the source CAD system. The value of CAD_PMI_COUNT shall match the sum of all subsequent CAD_PMI_COUNT_<type> properties given for the same view.

Property Name	JT File Data Type	Valid Values
CAD_PMI_COUNT	Integer	numeric

Table 7: View-Level PMI General Count Properties

All further CAD_PMI_COUNT_<type> validation properties contain the total count of PMI elements of the specified type contained in the Saved View, as defined in the source CAD system.

The types of PMI per Saved View are the same as the types of PMI at Part Level, see Table 6.

It is recommended that if a certain PMI element type does not occur in the view, to provide the corresponding CAD_PMI_COUNT_<type> property nonetheless, with a value of 0 (zero). This explicitly states that there are no elements of this kind in the current view.

5.4 Calculated Properties

Calculated properties contain validation values calculated from PMI content in the source CAD system. The allow for validation of additional aspects of a Saved View, such as the camera position or additional parameters defining the view. Detailed definitions how to calculate these properties are given in section 6.

Property Name	Valid Values	JT File Data Type
CAD_MV_STRING Unicode strings constructed of the following element: Unique identifier (VIEW) View control type identifier (perspective (PER), orthographic (ORG), view matrix (MAT), or other/unrecognizable (NA)) Model view name Background colors, space separated. Colors will be hex values, ordered in RGB. The order of the background colors will be: bottom left, bottom right, top left, top right. If there are no colors defined for a model view, the field should be blank All fields are separated by symbol, \w **Note:** See the CAD_MV_STRING examples below for the string description.	Unicode String	Defined description provided in section 6.
CAD_MV_CAMERA_POINTS A list of pairs of points The first point will represent a point in the world coordinate system. The second point will be the first point in the projection space of the model view camera Followed by that, the vertexes of a 10x10 cube in world coordinates. These 8 vertex points are to be used as the basis for point generation and will provide a common base between parts. A point should be three space separated numbers, a pair of points should be comma separated, and pairs should be semi-colon separated. **Note:** See the CAD_MV_CAMERA_POINTS example below.	Unicode String	“%.16g %.16g %.16g, %.16g %.16g %.16g; %.16g %.16g %16g, %.16g %.16g %.16g; %.16g %.16g %.16g, %.16g %.16g,%.16g; %.16g %.16g %.16g, %.16g %.16g,%.16g; %.16g %.16g %.16g, %.16g %.16g,%.16g; %.16g %.16g %.16g, %.16g %.16g,%.16g; %.16g %.16g %.16g, %.16g %.16g,%.16g; %.16g %.16g %.16g, %.16g %.16g,%.16g; %.16g %.16g %.16g, %.16g %.16g,%.16g; %.16g %.16g %.16g, %.16g %.16g,%.16g;“
CAD_MV_ASSOCIATED_PMI_ORIGINS The position of all the PMI associated with a model view will be validated using the origins of the PMI in the Model view. The PMI in a Model view is associated to that Model view.	Unicode String	“(%.16g, %.16g, %.16g); ...; (%.16g, %.16g, %.16g)” e.g. “(x1, y1, z1); … (x(n), y(n), z(n))”

Table 8: View-Level PMI Calculated Properties

5.4.1 CAD_MV_STRING example

See section 6.4 for guidelines to define the Unicode Strings for PMI.

Model View Properties	Example String
Type: Orthogonal Name: “Top” Bottom left color: 234, 232, 230 Bottom right color: 234, 232, 230 Top left color: 184, 182, 180 Top right color: 184, 182, 180	VIEW\wORG\w”Top”\weae8e6 eae8e6 b8b6b4 b8b6b4
Type: Perspective Name: “Bottom” Colors: N/A	VIEW\wPER\w”Bottom”\w
Type: View matrix Name: “Front” Bottom left color: 256, 256, 256 Bottom right color: 240, 180, 120 Top left color: 47, 168, 102 Top right color: 0, 0, 0	VIEW\wVM\w”Front”\wffffff f0b478 2fa866 000000

Table 9: CAD_MV_STRING Example

5.4.2 CAD_MV_CAMERA_POINTS example

World coordinate points	Model View / projection space points	Property value
1 2 3	2 3 1	“1 2 3, 2 3 1; 1 2 3, 0.5 0.6 0.76; 0 0 0, 0 0 0; 10 0 0, 0.25 0 0; 10 10 0. 0.25 0.25 0; 0 10 0, 0 0.25 0; 0 0 10, 0 0 0.25; 10 0 10, 0.25 0 0.25; 10 10 10, 0.25 0.25 0.25; 0 10 10, 0 0.25 0.25;”
1 2 3	0.5 0.6 0.76	
8 world vertex points	8 Model View / projection space vertex points	
0 0 0, 10 0 0, 10 10 0, 0 10 0, 0 0 10, 10 0 10, 10 10 10, 0 10 10	0 0 0, 0.25 0 0, 0.25 0.25 0, 0 0.25 0, 0 0 0.25, 0.25 0 0.25, 0.25 0.25 0.25, 0 0.25 0.25	

Table 10: CAD_MV_CAMERA_POINTS Example

5.5 Element-level PMI Validation Properties

The validation properties introduced in this section aim at validating individual PMI elements regarding completeness and correctness of the definition. They contain values calculated from PMI content in the source CAD system.

There are properties available for ‘associated’ as well as ‘attached’ geometry, which is defined as follows:

- **Associated:** this will link a single PMI to multiple faces or bodies. The association is used, for example, to link four holes with one feature control frame.
- **Attached:** Refers to the data the PMI is connected to or touching. For example, the curve or face the leader line from a feature control frame is touching out of the four it is associated with.

They can be the same face or body, and both are required.

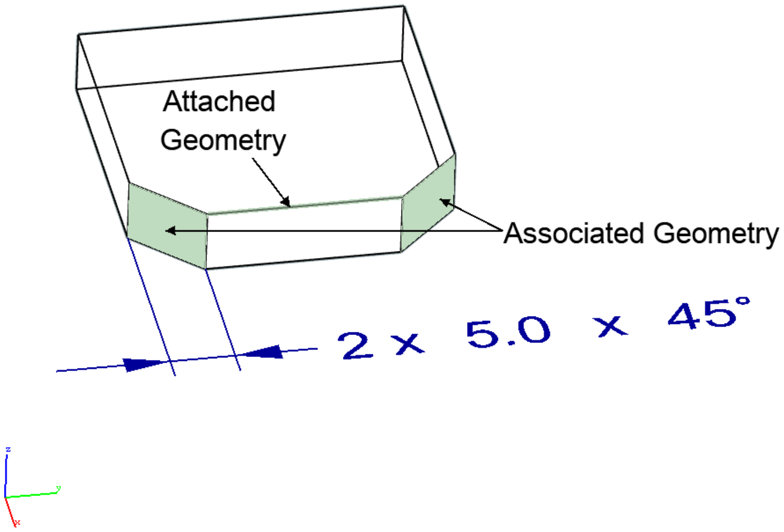


Figure 3: Associated vs. Attached Geometry Example

For information on calculating CAD_PMI_POLYLINE_LENGTH and CAD_PMI_CENTROID, see section 6 below.

__Note:__ Unit values are compliant with the ISO IS:14306 JT_PROP_MEASUREMENT_UNITS definition

__Note:__ The polyline length and centroid values provided for PMI annotations relate to the readily rendered representation of the PMI in the JT file. PMI in JT also include “building instructions” to recreate the annotation in an authoring tool (line styles, colors, font, etc.). This is a different set of data. As a result:

- If the importing application uses the readily rendered representation of the PMI annotation, the CAD_PMI_POLYLINE_LENGTH and CAD_PMI_CENTROID properties will allow validating the completeness and correct positioning of the PMI.
- If the importing application uses the “building instructions” to recreate the PMI annotation, its appearance likely changes, e.g. due to a different font being used. In this case, the CAD_PMI_STRING property can be used to validate the contents, and the CAD_PMI_CENTROID will allow for an approximate validation of the position (a recreated annotation may move slightly, but should end up on the other side of the model).
- If a PMI element imported from JT is changed in the authoring tool, it needs to be ensured that the corresponding validation properties are updated (re-calculated) when exporting the model to JT again.

Property Name	JT File Data Type	Valid Values
CAD_PMI_POLYLINE_LENGTH Total length of all the lines and circular arcs that make up the PMI. For filled fonts, the polyline length should include the outline of each of the filled characters and arrows.	Unicode String	"%.16g <unit>" *i.e “.5 millimeters”*
CAD_PMI_ATTACHED_GEOM_LENGTH The amount of curve length attached to the PMI in the source CAD system.	Unicode String	"%.16g <unit>"

Property Name	JT File Data Type	Valid Values
CAD_PMI_ASSOCIATED_GEOM_LENGTH Amount of curve length associated with the PMI in the source CAD system.	Unicode String	"%.16g <unit>"
CAD_PMI_ATTACHED_GEOM_AREA Amount of surface area of the part's B-rep attached to the PMI in the source CAD system.	Unicode String	"%.16g <unit>"
CAD_PMI_ASSOCIATED_GEOM_AREA Amount of surface area of the part's B-rep associated with the PMI in the source CAD system.	Unicode String	"%.16g <unit>"
CAD_PMI_CENTROID Mathematical centroid of the PMI presentation polylines indicated by the source CAD system	Unicode String	"%.16g %.16g %.16g" *i.e. "x1 y1 z1"*
CAD_PMI_STRING A string following the CAx Implementor Forum PMI Unicode String Specification **Note:** *The CAx-IF spec covered 4 PMI types. JT adopts the spec for these 4 PMI types and expands it to the remaining types defined in this document*	Unicode String	"String"
CAD_PMI_ASSOCIATED_PMI_ORIGINS The value of the property should be semi-colon delimited list of origin values **Note:** *Units are defined by the part units*	Unicode String	"(%.16g,%.16g,%.16g); ...; (%.16g,%.16g,%.16g)" *i.e. "(x1,y1,z1); … (x(n),y(n),z(n))"*
CAD_PMI_ATTACHED_PMI_ORIGINS The value of the property should be semi-colon delimited list of origin values **Note:** *Units are defined by the part units.*	Unicode String	"(%.16g,%.16g,%.16g); ...; (%.16g,%.16g,%.16g)"

Table 11: Part-Level PMI Calculated Properties

6 Methods for Calculating Validation Values

6.1 Bounding Box

Experience shows that the various CAD systems and validation tools determine the model extent in many different ways, thus making a comparison of these different values meaningless. Hence, the CAx Interoperability Forum agreed on a common algorithm to determine the bounding box, which will be reused here for consistency. Only bounding boxes computed per this agreement shall be exchanged as a Geometric Validation Property.

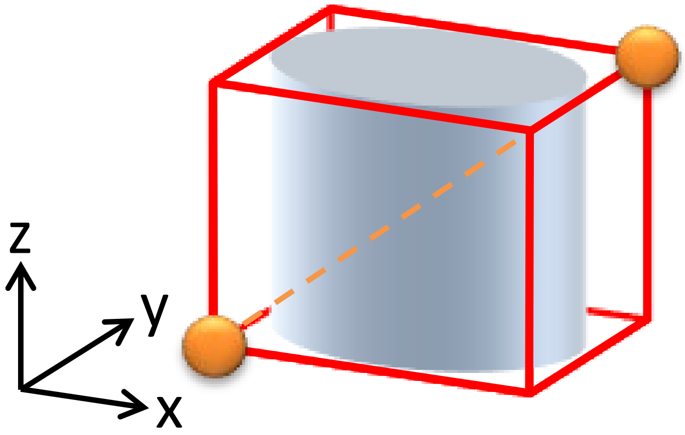


Figure 4: Bounding Box with corner points Min(x,y,z) and Max(x,y,z) for space diagonal

Agreement on the Algorithm to calculate the Bounding Box:

- Only use visible 3D elements of the geometry that would be exported to JT
- Do not include any annotations or axis systems into the calculation
- Use the vertices and edges of the XT-BRep solid, surface and wireframe geometry, or use all vertices from a tessellation of the model.
- 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)

6.2 Polylines

When calculating the value for a CAD_PMI_POLYLINE_LENGTH validation property the following rules apply.

Polylines length should be based on the following rules:

- The length of any polylines in the PMI (leader lines, non-filled text, etc.)
- The length of outlines for filled characters (filled text, etc.)
- Polyline length tests should take a percentile value as a threshold. This value will determine how far the polyline length calculated from the JT can differ from the polyline length supplied from the CAD system in the JT file. Percentile deviation should be calculated using this formula:

$$\frac{absolute(JT\ \ length - CAD\ length)}{CAD\ length}$$

6.3 Centroids

When calculating the value for CAD_PMI_CENTROID validation property the following rules apply.

Centroids are weighted centers, and are calculated based on the following algorithm:

- The centroid of a line segment is the center point of the line.
- The centroid of a circular arc is on the radius joining the center of the arc at the position:

$$p = (radius * \sin(arc\_angle/2) * 2)/(arc\_angle)$$

- For a set of line segments and arcs, the centroid is arrived at by a composition of all the element centroids, with the length of each line segment or arc as weight (similar as for a set of solids or surfaces).

6.4 Unicode Strings

PMI validation strings follow the syntax introduced in the CAx Implementor Forum’s document titled “PMI Unicode String Specification Examples and Mapping Strategies” (see Annex A), which provides a method to encode annotation contents as a Unicode string. This encoding is independent of the exchange format it is being used in and thus can be directly carried over and applied to JT files.

Annex A Bibliography

JT Implementor Forum Implementation Guidelines:

Version 3.0 – December 2018

https://www.prostep.org/fileadmin/downloads/JT_Implementation_Guidelines_v3.0.pdf

ISO JT File Specification:

ISO 14306:2017

Industrial automation systems and integration — JT file format specification for 3D visualization

https://www.iso.org/standard/62770.html

JT Industrial Application Package (JT IAP)

Version 2 – November 2016

https://www.prostep.org/fileadmin/downloads/JTIAP_V2.pdf

DIN SPEC 91383

JT Industrial Application Package (JT IAP)

https://www.beuth.de/en/technical-rule/din-spec-91383/296617117

Siemens JT File Specification:

In accordance with industry requirements, and following the recommendations given in DIN SPEC 91383, the preferred JT file format to be used in the JT-IF is JT version 10. While the prostep ivip JT IAP explicitly calls out version 10.0., the target version in focus for most user companies is Version 10.5. The JT file format specification for version 10.5 is publicly available at:

https://www.plm.automation.siemens.com/global/en/products/plm-components/jt-open-program.html

CAx Implementor Forum Unicode String Specification:

Revision J – May 2011

https://www.cax-if.de/documents/rec_prac_unicode_strings_rev_j_2011-05-25.pdf