Visualize 3D for STEP - JT

Product Release Version 23.1

USER GUIDE

Revision: 1.0

Issued: 18/06/2020

Contents

Overview of Visualize 3D

About Theorem

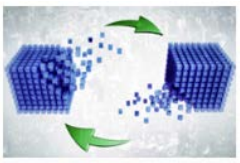
Theorem Solutions is a world leader in the field of Engineering Data Services and Solutions. This leadership position stems from the quality of our technology and the people in the company. Quality comes not only from the skills and commitment of our staff, but also from the vigorous industrial use of our technology & services by world leading customers.

We are proud that the vast majority of the world's leading Automotive, Aerospace, Defense, Power Generation and Transportation companies and their Supply chains use our products and services daily. Working closely with our customers, to both fully understand their requirements and feed their input into our development processes has significantly contributed to our technology and industry knowledge.

Theorem Solutions is an independent UK headquartered company incorporated in 1990, with sales and support offices in the UK and USA. Theorem has strong relationships with the major CAD and PLM vendors, including; Autodesk, Dassault Systemes, ICEM Technologies (a Dassault company), PTC, SolidWorks, Spatial Technology and Siemens PLM Software. These relationships enable us to deliver best in class services and solutions to engineering companies worldwide.

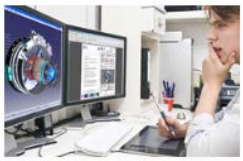
What is Visualize 3D?

Visualize 3D is one of 5 core Theorem brands which consist of:



CADverter

Direct translation of 3D data to or from an alternate CAD, Visualization or Standards Based format



Multi-CAD

Interactive integration of non-native 3D data formats into the native CAD system

Visualize 3D

Direct translation of 3D data for the purpose of Visualization

Publish 3D

The creation of documents enriched with 3D content

Process Automation

Applications to automate any Data Exchange and collaboration processes

The STEP Bi-directional JT Translator

The translator may be installed on a number of machines each accessing a central network-floating license.

Theorem's Visualize 3D product for STEP to JT is a direct converter between STEP files and Siemens JT file format. It enables the user to convert all forms of mechanical design geometry, as well as assembly and attribute information, between these two systems.

The STEP-JT Visualize 3D product can be purchased as a uni-directional, STEP to JT, or JT to STEP product, or as a bi-directional product. It can be used interactively or in a batch mode, from a standard GUI Interface, offering combined viewing, data filtering and translation capabilities.

Primary Product Features

- Converts all types of geometry, wire frame, surfaces, trimmed surfaces (faces) and solid models
- Converts assembly structure between the systems
- Converts attribute data including colour and layer information
- The conversion process can be run Interactively or in Batch mode
- Data can be filtered by layer and entity type
- Geometry can be filtered and selectively processed

Primary Product benefits?

- Direct conversion between STEP and JT reduces processing time, simplifies integration and retains accuracy of the model
- The integrated viewing capability enables visual verification, pre and post translation
- The integrated data filtering options allows selected data ONLY to be processed, enabling optimisation of translations and time savings

- By converting all forms of geometry no data is lost, eliminating the time required to recreate missing data

- With over 20 years industrial use, Theorem's product robustness and quality is well proven, reducing your business risk

This document will focus specifically on guidance for the use of the Visualize 3D for STEP – JT product. For information regarding any of Theorem's product ranges please contact sales@theorem.com

Getting Started

Documentation

The latest copy of this documentation can be found on our web site at:

http://www.theorem.com/Documentation

Each product has a specific link that provides user documentation in the form of PDF and Tutorials.

Installation Media

The latest copy of Theorem software can be found via our web site at:

http://www.theorem.com/Product-Release-Notes

Each product has a specific link to the Product Release Document, which contains a link to the download location of the installation CD.

Alternatively, you can request a copy of the software to be shipped on a physical CD.

Installation

The installation is run from the CD or ZIP file download provided.

Currently, there are 2 distinct installation stages that are required.

To install the translator, select the ***Setup.exe*** file and follow the installation process. For a full guide to the process, please see our 'Translator Installation Process' demonstration video located here.

In addition, the Theorem Unified Interface will also need to be installed. The installation process is the same as for the Translator. For a full guide to the process, please see our 'Translator Installation Process' demonstration video located here.

License Configuration

In order for the translation to run successfully, the Theorem license file provided to you needs to be configured using FlexLM. For a full guide to this process, please see our 'FlexLM License Set Up and Configuration' demonstration video located here.

Running the Product

Once configured and licensed, the product is ready to be run.

There are 2 distinct ways of running the translator:

- Via the Theorem Unified Interface

- The Unified Interface offers a Desktop Environment that allows CAD and Visualization data to be viewed pre and post translation

- Via the Command Line

- The Command Line Interface provides a direct method of invoking the translator. It can be used via a DOS shell or called via a third party application as part of a wider process requirement.

Using the Product

Default Translations

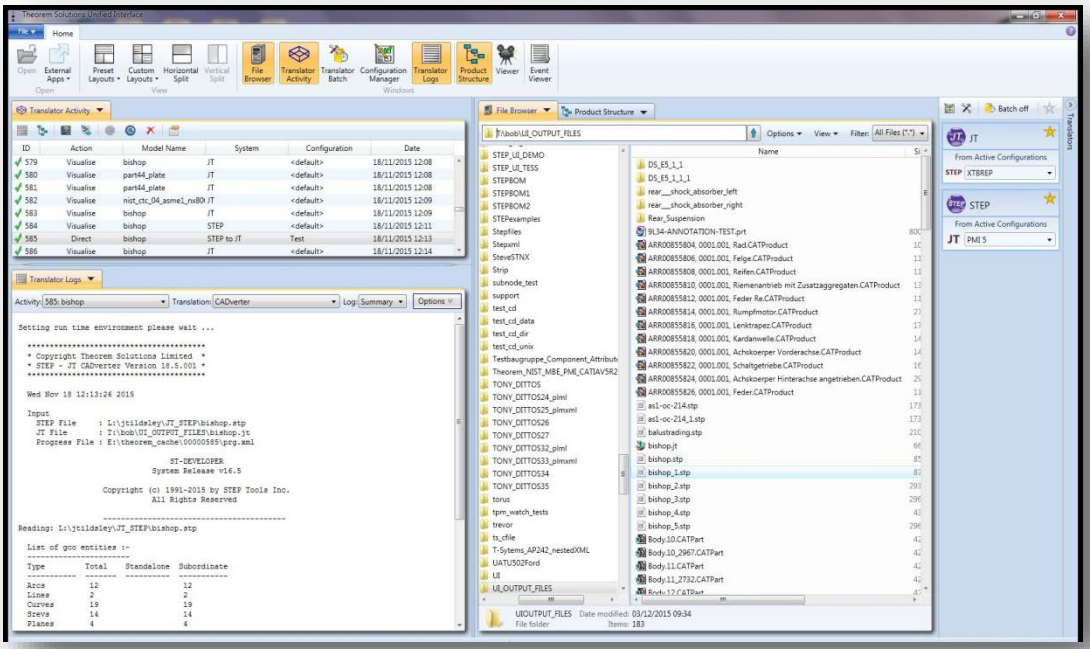
Default Translation – via the Unified Interface

The Unified Interface can be started via the Start Menu – if a shortcut was added during installation.

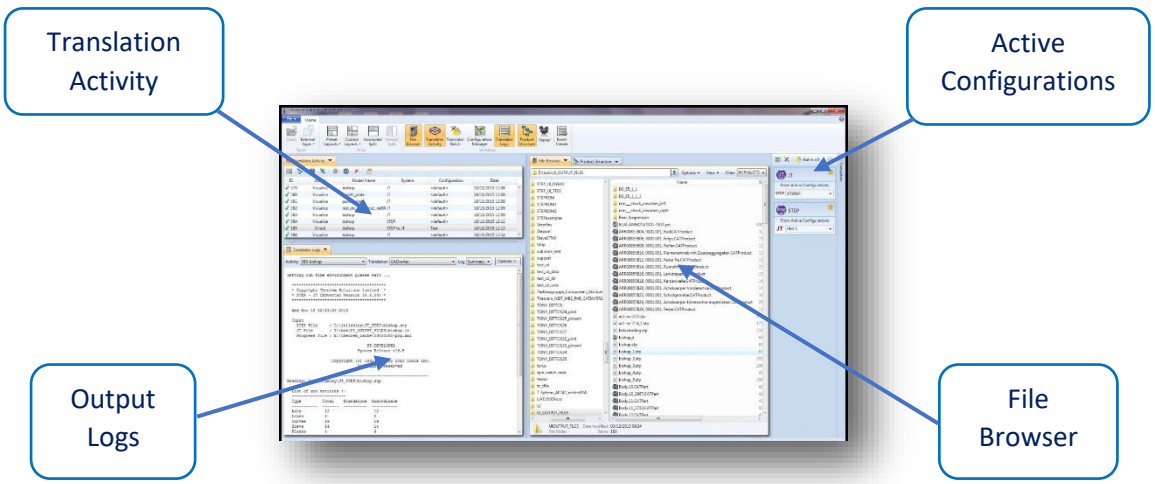
Alternatively, the Unified Interface can be run via a Windows Explorer selection in:

<UI_installation_directory>\bin\Unified_Interface.cmd

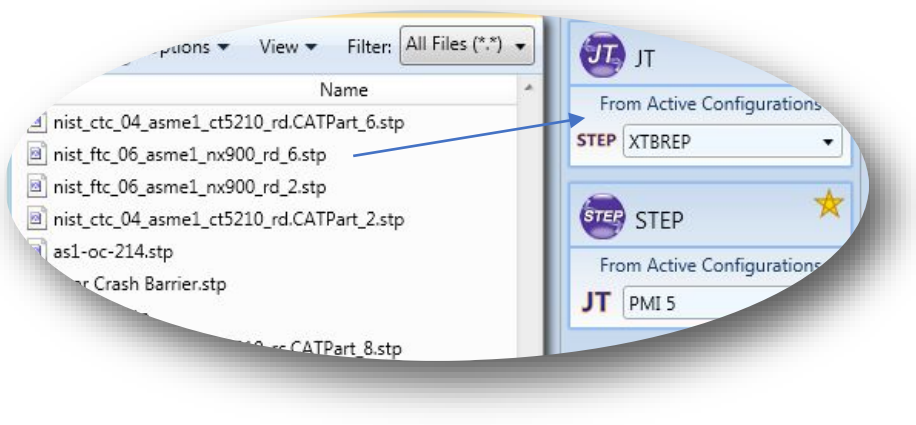
The following interface will be launched:



The default layout is split into 4 primary areas, which can be altered to the users preference:

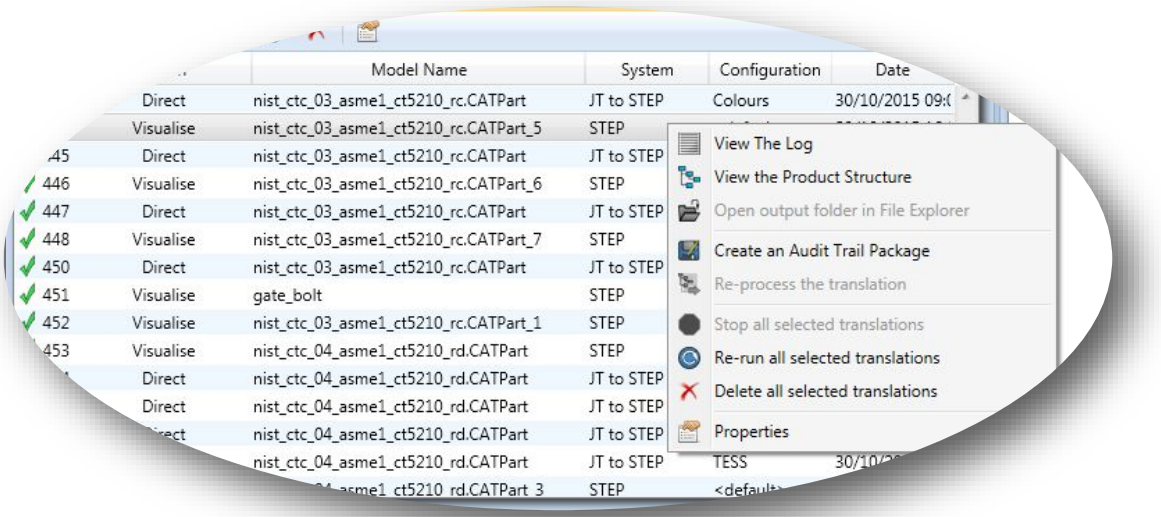


The simplest way to translate from STEP or JT is to drag a file from the file Browser Pane on to the Active Configurations for the translation you require.



On completion, the Unified Interface will display the activity information and details from the log file created during the translation, if requested, in the Translation Activity and Output Log panes, respectively.

The generated output data can be located by selecting the translation from the Activity pane and opening the output folder:



Default Translation – via the Command Line

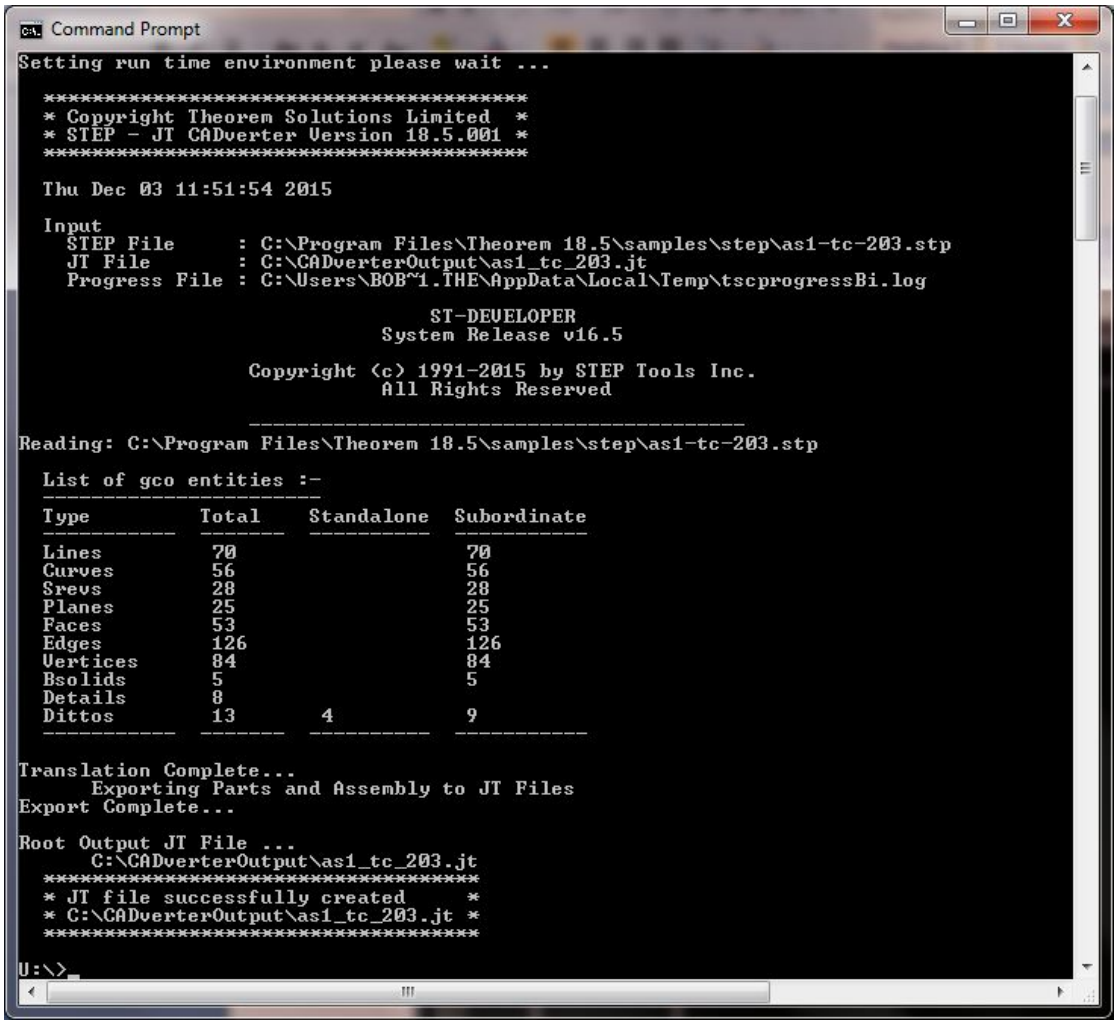
Running a translation via the command line can be carried out via the ***cad_run.cmd*** file located in the ***<installation_directory>\bin*** directory. The format of the command is as follows when translating from STEP to NX:JT

<Translator_installation_directory>\bin\cad_run.cmd STEP_JT –i <input_file> -o <output_file>

The format of the command is as follows when translating from JT to STEP:

<Translator_installation_directory>\bin\cad_run.cmd JT_STEP –i <input_file> -o <output_file>

The example above will translate a STEP sample file provided within the installation and produce the following screen output:



The file will be output to the target location. In this case:

C:\CADverter_Output\ as1-tc-203.jt

Translator Customization

Visualize 3D allows the information that is read from the source system and written to the target system to be tailored via a set of user specified arguments. Commonly used arguments are supported via the Unified Interface, with Advanced Arguments being described within this document for use in the Unified Interface or via the Command Line invocation.

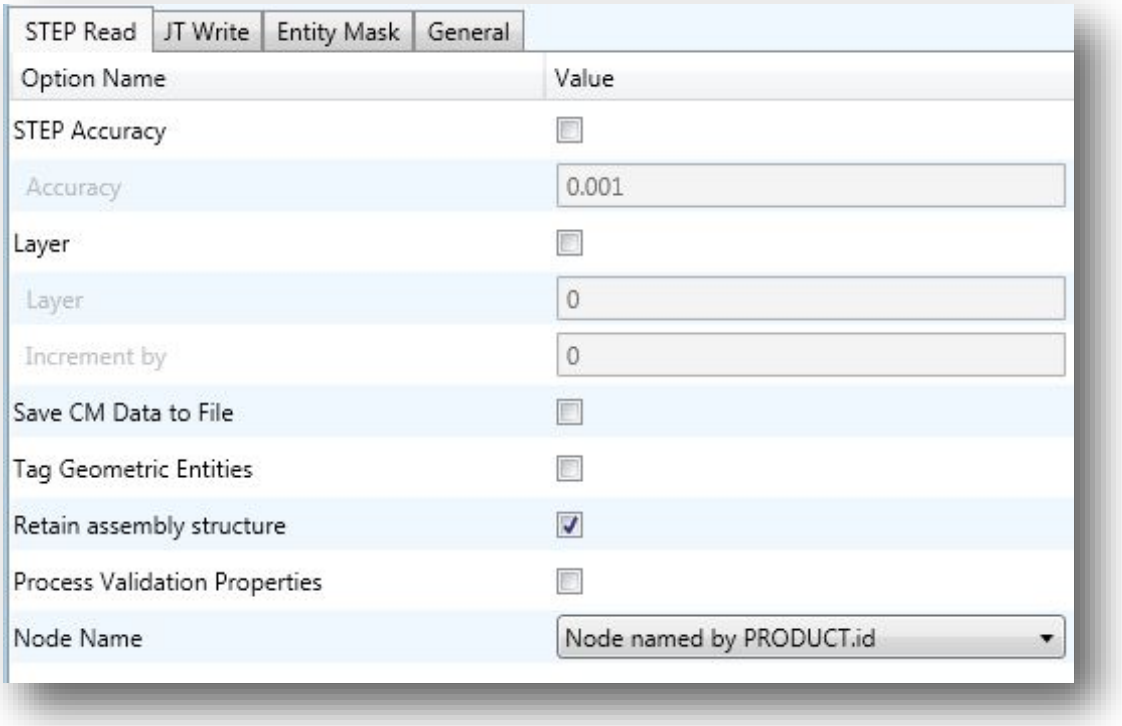
Common Options for STEP to JT

Within the Configuration Manager pane of the Unified Interface, arguments that can be specified when publishing STEP data into JT are grouped into 4 areas:

- STEP Read – Those arguments that affect how data is read from STEP
- JT Write – Those arguments that affect how the data is written to JT
- Entity Mask – Those arguments that allow specific read entities to be masked
- General – Those arguments that are common to ALL Publishing activities regardless of source data

STEP Read Arguments

The image below shows the STEP Read arguments that are available, with their default settings:



Each of these options is described below:

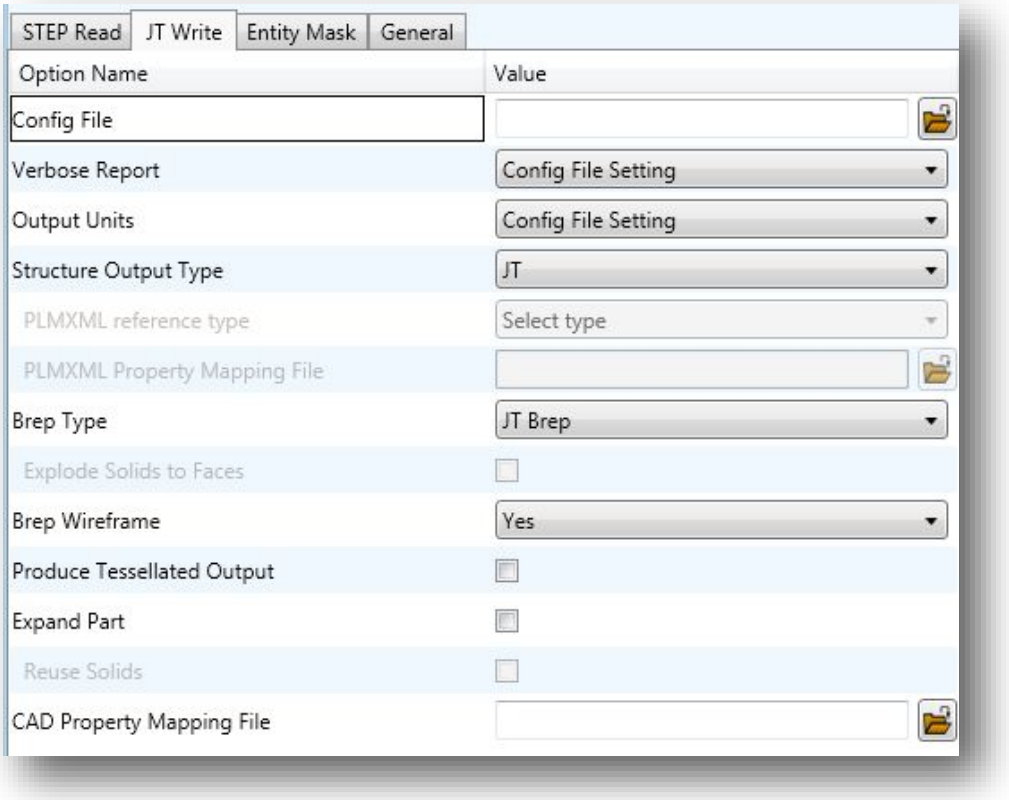
Option	Description
STEP Accuracy	This option is used to specify the tolerance value used by the translator to decide whether a curve or surface is closed, or a surface is degenerate. For most cases, the default value of 0.001 mm will be adequate. o Command Line Syntax: ▪ *step_accur 0.001*
Layer	This option may be used to define layer information, if the input STEP File does not contain layering, for processing into the receiving system. You can specify either the one and only layer that the STEP data is to be put on to (if the Incremental Layer option is not used), or the start layer if other than zero is required (if the Incremental Layer option is used). For example, if a layer of 3 is specified and the Incremental Layer option is not used, then all geometry will be written to layer 3. If you specify a layer of 3 and set a value in the Incremental Layer

	option, then the geometry from each Shape Representation encountered in the STEP file will be put on a new layer starting from layer 3, incremented by the value entered each time. **NOTE: -** If a STEP AP203 or STEP AP214 file containing Colours and Layer Modular extension data is read, the values in the file will take precedence over the values entered by this option. o Command Line Syntax: ▪ *layer "number"* ▪ *layer_incr "number"*
Save CM Data to File	STEP files contain more than just geometric data. In addition it can contain Configuration Management Data (also known as CM Data, or PDM information), such as the names of people and organisations fulfilling various roles (design creator etc.). By using this option this non-geometric data will be printed out into a CM Data File, which conforms to the Theorem Solutions Ltd. CM Data File format. The CM Data File will be named ***CAD_filename*.cmdata** where ***CAD_filename*** is the name of the CAD output file being created. o Command Line Syntax ▪ *print_cmdata*
Tag Geometric Entities	Many CAD systems have the capability to apply TAG values to geometric entities. This option instructs the translator to use the value found in the 'name' field of the Geometric Representation Item as this TAG value. o Command Line Syntax ▪ *tag_name*
Retain Assembly Structure	If the Input STEP File contains any assembly structure, then by default assembly structure will be created in the Output CAD format. Default is ON. Deselection of this option (or the command line option *'offditto'*) causes the assembly structure to be "exploded" into a flat single component file. o Command Line Syntax ▪ *offditto*
Process Validation Properties	If the STEP File contains Geometric Validation Property data, conforming to the appropriate modular extension, then this flag activates processing of this data. This data is used to provide a check on the success of the translation. The method used for this check depends upon the target system, some will automatically check the data (e.g.

	Unigraphics) Default is OFF. o Command Line Syntax ▪ *print_val*
Node Name	This option gives the choice of which PRODUCT entity attribute to use for naming the structure nodes in the target system, when reading in an assembly from STEP. Default is PRODUCT.id o Command Line Syntax ▪ *use_prod_name* or *use_prod_id*

JT Write Arguments

The image below shows the JT Write arguments that are available, with their default settings:



Each of these options is described below:

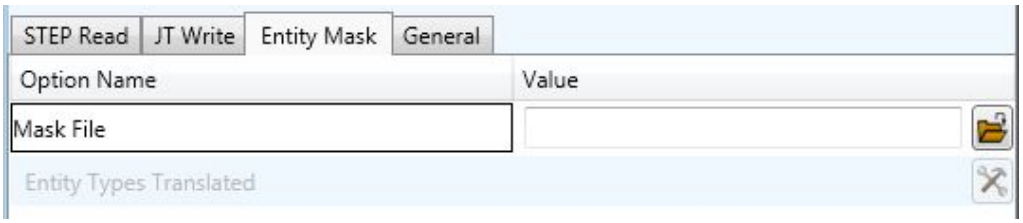
Option	Description
Config File	Allows a JT configuration file to be specified. Please see Appendix B for a full description of the JT config file format. o Command Line Syntax ▪ *-z [path_to_file]*
Verbose Report	Defines the report. Default is *'Config File setting'* o Command Line Syntax ▪ *Config File Setting: Default* ▪ *No: -VerboseReporting false* ▪ *Yes: -VerboseReporting true*
Output Units	Output unit definition. Default is *'Config File setting'* o Command Line Syntax ▪ *Config File Setting: Default* ▪ *As Input: -OutputUnits inputUnits*

	▪ Millimeters: *-OutputUnits mm* ▪ Centimeters: *-OutputUnits cm* ▪ Meters: *-OutputUnits m* ▪ Inches: *-OutputUnits inches* ▪ Feet: *-OutputUnits feet* ▪ Yards: *-OutputUnits yards*
Structure Output Type	Specifies the type of assembly structure to be output. Default is JT. • Selectable options are: *JT: Default* *PLMXML (with external references to JT files)* *STEP BOM (with external references to JT files)* o Command Line Syntax ▪ JT: *Default* ▪ PLMXML: *<See PLMXML Ref Type>* ▪ STEP BOM*: write_stepbom*
PLMXML Reference Type	Only active if a Structure Type of PLMXML is selected. • Selectable options are: 1. *PLMXML referencing JT Parts in the same folder* 2. *PLMXML in addition to the JT Assembly File* o Command Line Syntax ▪ 1: *plmxml_only* ▪ 2: *write_plmxml*
PLMXML Property Mapping File	Allows a Property Mapping file to be selected. Only active if a Structure Type of PLMXML is selected. *(See Appendix C for the file format)* o Command Line Syntax ▪ *plmxml_prop_map_file [File]* ***Note!*** *When this option is unset, the file "plmxml_property_mapping.txt" in the 'data\jt folder' will be used as the mapping file. This file contains lines with mappings that are mandatory for certain downstream applications and mappings to remove attributes used solely in the translation process. Therefore it is a good idea to start with a copy of this file when creating a new mapping file.*
Brep Type	Specifies the BREP type in the resultant JT Files. Default is JT Brep. • Selectable options are: *JT Brep: Default* *XT Brep (Theorem)* *JT Brep (JT Open)* o Command Line Syntax ▪ *JT Brep: Default* ▪ *XT Brep (Theorem): xt_brep no_fixup* ▪ *JT Brep (JT open): jt_xt_brep*

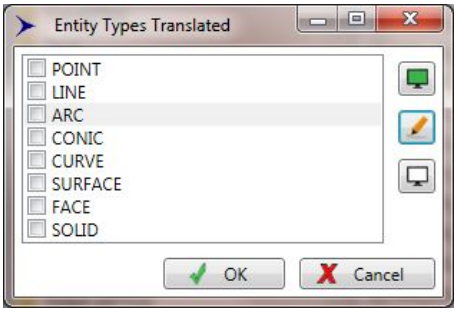
Explode Solids to Faces	A secondary option enabled when XT Brep (Theorem) output is specified. Explodes solids to faces. Default is OFF. o Command Line Syntax ▪ *split_brep*
Brep Wireframe	Store wireframe in the JT Brep. Default is YES. • Selectable options are: *Yes: Default* *No* o Command Line Syntax ▪ *no_brep_wire : No (Tessellated)*
Produce Tessellated Output	Convert Brep data directly to facetted data. Default is OFF. o Command Line Syntax ▪ *tess_output*
Expand Part	Process multi-solid parts as an assembly. Default is OFF. o Command Line Syntax ▪ *expand_part*
Reuse Solids	A secondary option enabled when Expand Part is set to re-use existing solids. Default is OFF. o Command Line Syntax ▪ *reuse_solids*
CAD Property Mapping File	A file containing a list of CAD properties and information on how they are mapped to the JT file. Not used by default. *(See Appendix C for the file format)* o Command Line Syntax ▪ *cad_prop_map_file [File]* ▪ ***Note!*** *An example mapping file is provide in the 'data\jt' folder. This file provides a good starting point when creating a new mapping file.*

STEP to JT Entity Masking Arguments

The image below shows the Masking arguments that are available, with their default settings:

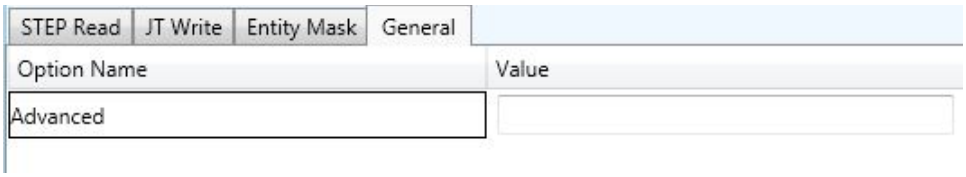


Each of these options is described below:

Option	Description
Mask File	Specifies the Mask File to be written to, that can be referenced by future translations. A Mask file MUST be specified if masking is required. The first line in this file is OFF ALL ENT: o Command Line Syntax: ▪ *Mask <filename>*
Entity Types Translated	Specifies a selection list (see picture) from which to select which entity types are to be processed. The following types are available: "POI","LIN","ARC","CON","CUR","SUR","FAC","SOL" o Command Line Syntax: ▪ *Add any of the above to the specified mask file, one entry per line prefixed by the word ON, e.g.:* ***ON POI*** *to ensure they are considered in the translation* 

STEP to JT General Arguments

The image below shows the General arguments that are available, with their default settings:



Each of these options is described below:

Option	Description
Advanced	Allows any of the Command Line Advanced arguments documented below to be passed to the Unified Interface invocation

STEP to JT Advanced Arguments

Theorem's STEP to JT translator has been configured with default settings that optimise the translation process. However, there are times when a satisfactory result cannot be obtained, so it may be required to deploy one or more Advanced Arguments to improve the translated result.

The following table describes useful Advanced Arguments that can be entered into the General Tab -> Advanced field:

Option	Description
Parasolid Tolerant Modelling	A secondary option to be used when *Brep Type = XT Brep (Theorem)* output is specified. Enables Parasolid tolerant modelling. Default is ON o Command Line Syntax ▪ *nopstolmodel – to turn off*
Factor	Specify the factor level of Parasolid Tolerant Modelling when turned on. Default is 3. o Command Line Syntax ▪ *pstolmodel 3*
Sew Parasolid Bodies	A secondary option to be used when *Brep Type = XT Brep (Theorem)* output is specified. Enables the sewing of Parasolid

	bodies. Default is ON o Command Line Syntax ▪ *nosew – to turn off*
Tolerance	Specify the tolerance for the sew command above. Default is 0.01. o Command Line Syntax ▪ *pssew 0.01*
Incremental Sewing	Enables incremental sewing when used with Sew Parasolid Bodies. Default is ON. o Command Line Syntax ▪ *no_sew_increm – to turn off*
Split Discontinuous Surfaces	A secondary option to be used when *Brep Type = XT Brep (Theorem)* output specified. Splits discontinuous surfaces. Default is OFF. o Command Line Syntax ▪ *brep_prep – to turn on* ▪ *no_brep_prep – to turn off*
Force body creation (No check of Parasolid entities)	A secondary option enabled when XT Brep (Theorem) output specified. Removes the checking of Parasolid entities. Default is ON. o Command Line Syntax ▪ *nocheck* – (force body creation without checking = Default) ▪ *check* – (doesn't force the body creation - Parasolid checking is enabled)
Fix Degenerate Edges	A secondary option to be used when *Brep Type = XT Brep (Theorem)* output specified. On face create failure, check and fix any degenerate edges. Default is ON. o Command Line Syntax

	▪ *fix_degen* ▪ *no_fix_degen – to turn off*
Specify a Face Edge Tolerance	A secondary option to be used when *Brep Type = XT Brep (Theorem)* output specified. Specify an edge tolerance to be used when creating faces. Default is ON. o Command Line Syntax ▪ *Please see Edge Tolerance below*
Edge Tolerance	A secondary option used with Specify a Face Edge Tolerance where the tolerance value is assigned. Default is 0.000006. o Command Line Syntax ▪ *face_edge_tol 0.000006*
Fix small features in solids	A secondary option to be used when *Brep Type = XT Brep (Theorem)* output specified. Remove small edges, sliver and spike faces from solid bodies. Default is OFF. o Command Line Syntax ▪ *ps_fix_small – to turn on* ▪ *no_ps_fix_small - default*
Fix small features in open solids	A secondary option to be used when *Brep Type = XT Brep (Theorem)* output specified. Remove small edges, sliver and spike faces from open solids. Default is OFF. o Command Line Syntax ▪ *ps_fix_osol – to turn on* ▪ *no_ps_fix_osol - default*
Simplify Geometry	A secondary option to be used when *Brep Type = XT Brep (Theorem)* output specified. Simplify Geometry. Default is OFF. o Command Line Syntax ▪ *simplify_solids – to turn on*

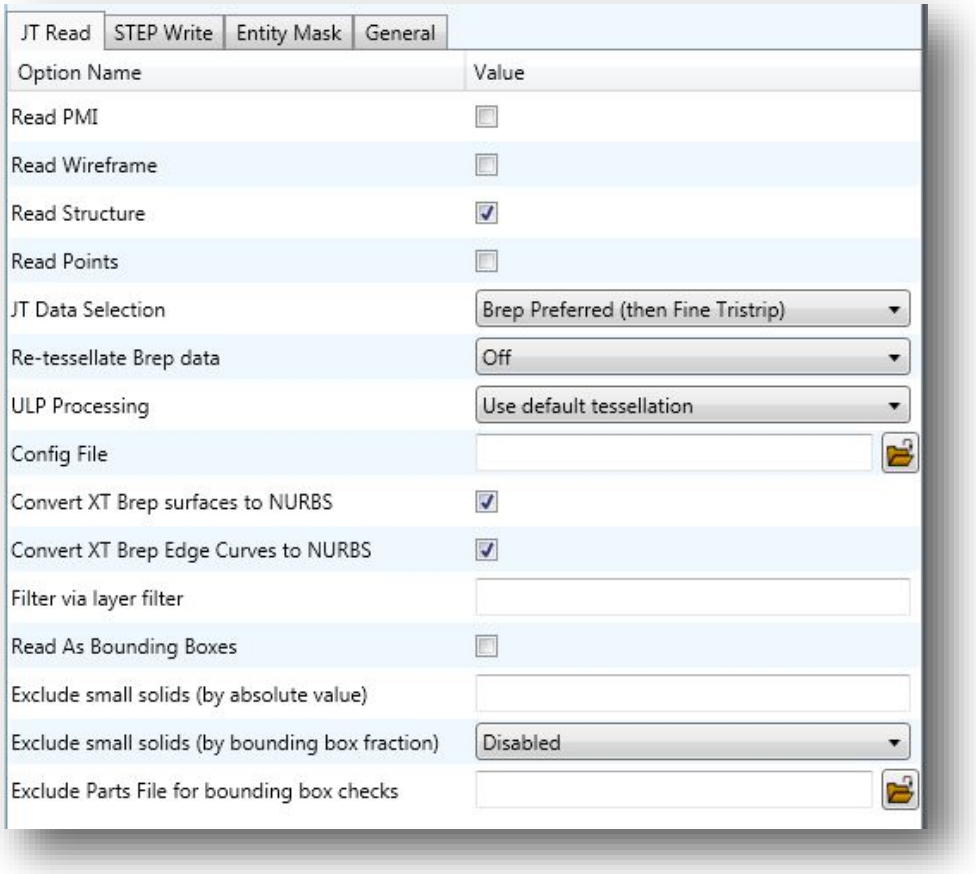
Common Options for JT to STEP

Within the Configuration Manager pane of the Unified Interface, arguments that can be specified when publishing JT Data into STEP are grouped into 4 areas:

- JT Read – Those arguments that affect how data is read from JT
- STEP Write – Those arguments that affect how the data is written to STEP
- Entity Mask – Those arguments that allow specific read entities to be masked
- General – Those arguments that are common to ALL Publishing activities regardless of source data

JT Read Arguments

The image below shows the JT Read arguments that are available, with their default settings:



Each of these options is described below.

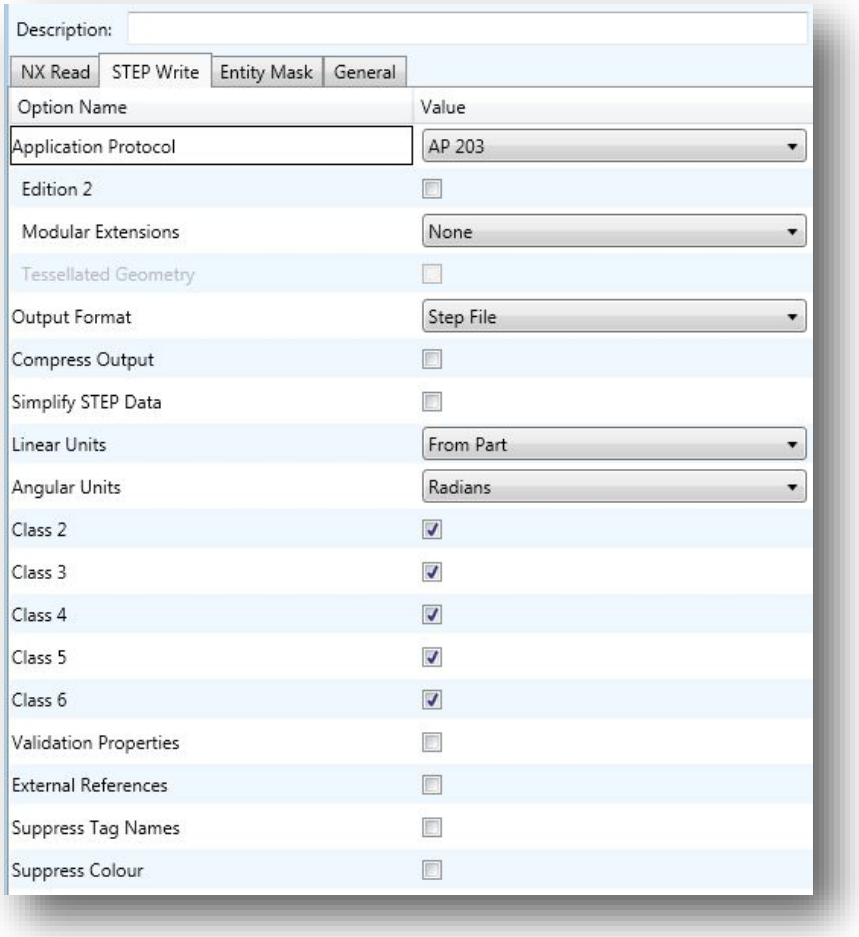
Option	Description
Read PMI	Reads 3D PMI. Default is OFF. o Command Line Syntax ▪ *read_pmi*

Read Wireframe	Read JT wireframe data. Default is OFF. o Command Line Syntax ▪ *read_wire_frame*
Read Structure	Read assembly structure. Default is ON. o Command Line Syntax ▪ *structure* ▪ *no_structure – to turn off*
Read Points	Read JT Point data. Default is OFF. o Command Line Syntax ▪ *read_points*
JT Data Selection	Select Brep or tessellated data read. Default is 'Brep Preferred (then fine tristrip)'. o Command Line Syntax ▪ *Brep Preferred (then Fine Facet): brep_pref* ▪ *Brep preferred (then Fine Tristrip): brep_pref_tri* ▪ *Brep Only: brep_only* ▪ *Fine Facet: fine_facets* ▪ *Fine Tristrip: fine_tristrips* ▪ *Coarse Facet: coarse_facets* ▪ *Coarse Tristrip: coarse_tristrips*
Re-tessellate Brep Data	Allows Brep data to be re-tessellated to create a link between the faces in the Brep and the tessellated representation. There is a choice between processing just the tessellated form or both the tessellated and Brep forms with the links between them. Default is 'Off'. o Command Line Syntax ▪ *On – Re-tessellate Brep Data: tess_brep* ▪ *On – Re-tessellate and Read Brep Data: tess_and_read_brep*
ULP processing	Allows control of Ultra Lightweight Precise data tessellation. Default is 'Use default tessellation' which tessellates the ULP data as the JT file is imported into the translator using default tessellation parameters. o Command Line Syntax ▪ *Tessellate ULP Data: tess_ulp*
Config File	Allows a JT config file to be specified that will contain tessellation parameters to be used to tessellate any Brep or ULP data. N.B. Only the LOD 0 parameters in the file are used. Please see Appendix B for a full description of the JT config file format. o Command Line Syntax ▪ *-z [path_to_file]*
Convert XT Brep surfaces to NURBS	Read XT Brep surfaces as NURBS surfaces (else read in native form). Default is ON. o Command Line Syntax ▪ *noprep – to turn off*
Convert XT Brep Edge Curves to NURBS	Read XT Brep edge curves as NURBS curves (else read in native form. Default is ON. o Command Line Syntax ▪ *rd_native_edge – to turn off*
Filter via layer filter	Supply layer filter(s) separated by commas and double quoted. Default is OFF.

	○ Command Line Syntax ▪ *layer_filter*
Read As Bounding Box	This option allows the user to read the JT file as a very light weight bounding box representation, with each solid having its own bounding box. This option may be useful when the part's details are not of interest, as much as the physical space, the part occupies. ○ Command Line Syntax ▪ *bounding_box*
Exclude small solids (by absolute value)	Exclude solids that have bounding boxes smaller than value. Default is OFF. ○ Command Line Syntax ▪ *bb_exclude_value [value]*
Exclude Small Solids (by bounding box fraction)	This option allows the user to exclude small solids from the JT read, the excluded solids being smaller than a specified fractional size of the overall bounding box of the part. ○ Command Line Syntax ▪ *bounding_box_frac [value]*
Exclude Parts File for bounding box checks	This option allows the user to supply an input file containing a list of JT parts, one per line. The parts specified in the file will be excluded from the checks to remove small solids. This option is used in conjunction with the "Exclude Small Solids" options. ○ Command Line Syntax ▪ *bb_exclusion_file '[path_to_file]*

STEP Write Arguments

The image below shows the STEP Write arguments that are available, with their default settings:



Each of these options is described below:

Option	Description
Application Protocol	Specifies the type of STEP file. Default is AP 203. o AP 203, *Configuration controlled 3D designs of mechanical parts and assemblies*. o AP 214, *Core data for automotive mechanical design processes* o AP 242 *Managed model based 3d engineering* o Command Line Syntax ▪ *AP 203* or *214* or *242*
Edition 2	For AP 203 STEP Files, specifies that edition 2 of the standard is to be used. o Command Line Syntax ▪ *edition2*

Modular Extensions	Allows modular extensions to be used in AP 203 files. The options are:- o None (Default) Do not use modular extensions in the AP 203 file o Colours Layers Extension – use the colours and layers modular extensions in the AP 203 file o Command Line Syntax ▪ *clg* o 3D Text Colours Layers – use the colours, layers and 3D text modular extensions in the AP 203 file. o Command Line Syntax ▪ *clgwis*
Tessellated Geometry	Specifies that tessellated representation is to be used for tessellated/facetted solids in the output AP 242 file. The alternative is to use the older facetted representation. o Command Line Syntax ▪ *fsol2 write_tess*
Output Format	Specifies the text format of the STEP file. Default is Part 21. o Part 21, *Conventional text*. o Part 28, *Formatted using the XML standard* o Command Line Syntax ▪ *part 21* or *28*
Compress Output	Specifies that the output STEP file is to be compressed o Command Line Syntax ▪ *compress*
Simplify STEP Data	Any BSpline geometry from the Input file will be checked to see if it can be accurately represented by analytical geometry (e.g. Arcs or spheres). If this is the case, the BSpline geometry will be converted to the appropriate analytical curve or surface, thus reducing the overall size of the resultant STEP File. o Command Line Syntax ▪ *simplify*
Linear Units	Specifies to units to be used in the output STEP files. The translator will write the STEP File in identical units of length to those used in the Input file. If different units of length are required (i.e. to send an Inch part from a supplier in the USA, to a European company modelling in Millimetres), then this option allows for the specification of a different unit of length to be used for data written to the STEP File. o Command Line Syntax ▪ *length_measure mm* or *inch*
Angular Units	By default, the angular units written to a STEP File are Radians. This is the most accurate method for Data Exchange as the majority of CAD Systems use Radians for their internal unit of Angular Measure. If it is required to use a different angular unit then, by applying this option, the translator will write angular units of the specified type. o Command Line Syntax ▪ *angle_ measure radian* or *degree*

Class 2
Class 3
Class 4
Class 5
Class 6

This instructs the translator as to which Conformance Class of data the user wishes to create in the STEP file. For example, the original CAD file may contain solids data, however the user needs it represented in the STEP file as a collection of trimmed surfaces. This would be achieved by selecting Class 4 for output. The translator will then interpret the data in the context of the specified class.
By default, all classes are on, so the resulting STEP File would maintain the data as it appears in the input file.

Conformance Classes
STEP groups its entities into Conformance Classes 1 to 6. The definitions of the Classes are given below:-

Class	Definition
1	Configuration-controlled design information without shape
2	Class 1 plus shapes represented by geometrically bounded wireframe models, surface models or both
3	Class 1 plus shapes represented by wireframe models with topology
4	Class 1 plus shapes represented by manifold surface models with topology
5	Class 1 plus shapes represented by facetted B-rep solids
6	Class 1 plus shapes represented by advanced B-rep solids

Class 1 alone is not supported as it is non-geometric.

- Command Line Syntax
  - *class 2* or *3* or *4* or *5* or *6*

Validation Properties

If this option is selected, then the trabslator will write Validation Data (Volume, Surface Area and Centroid information) into the STEP File.
If a suitable STEP post-processor is used to read the STEP File, these values can be used to automatically validate that the exchange has been a success.

- Command Line Syntax
  - *val*

External References

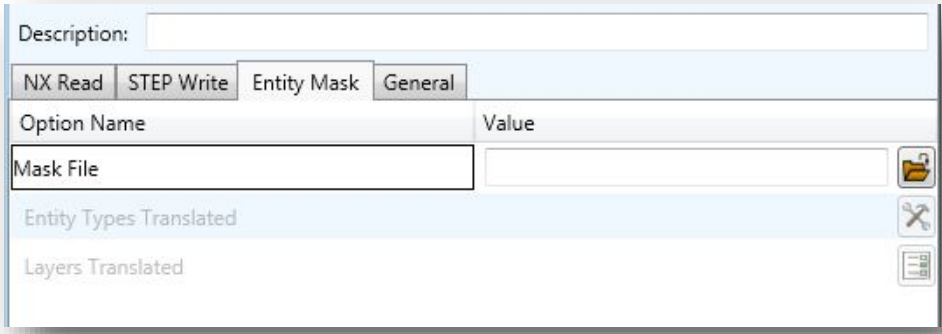
The translator has the capability to create a STEP File containing Structure and PDM information, with references to the Geometry files as either Native CAD models or STEP Files.
This capability is useful for exchanging large structures in multiple files rather than creating a single monolithic STEP File with both Structure and Geometry embedded in it.

- Command Line Syntax
  - *ext_ref*

Suppress Tag Names	Switches off the processing of TAG names from the STEP File. Some STEP Files contain TAG Names which are duplicates of those applied to other STEP geometric entities (e.g. all Edge Curves may be tagged as "Edge_Curve"). Since most CAD systems require TAG names to be unique, this can lead to extremely long processing times, as each TAG must be altered to avoid name clashes. o Command Line Syntax ▪ *tag_name default_null*
Suppress Colour	Prevents colour being added to the entities in the output STEP file o Command Line Syntax ▪ *no_style*

JT to STEP Entity Masking Arguments

The image below shows the Masking arguments that are available, with their default settings:

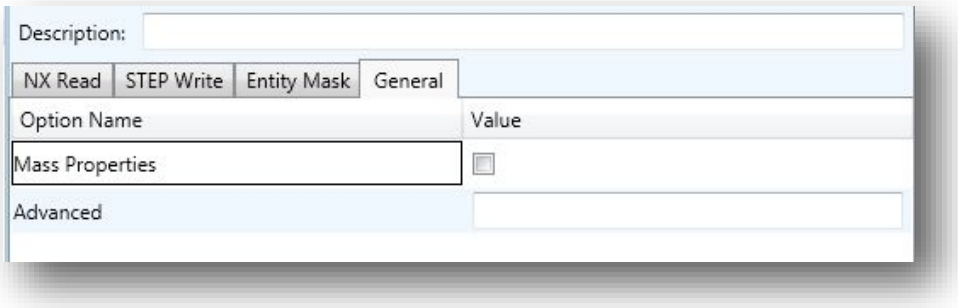


Each of these options is described below:

Option	Description
Mask File	Specifies the Mask File to be written to, that can be referenced by future translations. A Mask file MUST be specified if masking is required. The first line in this file is OFF ALL ENT: o Command Line Syntax: ▪ *Mask <filename>*
Entity Types Translated	Specifies a selection list from which to select which entity types are to be processed. The following types are available: "POI",”LIN”,”ARC”,”CON”,"CUR",”SUR”,”FAC”,"SOL" o Command Line Syntax: ▪ *Add any of the above to the specified mask file, one entry per line prefixed by the word ON, e.g.:* ***ON POI*** *to ensure they are considered in the translation*
Layers Translated	Specifies a selection list from which to select which layers are to be processed. o Command Line Syntax: ▪ *A single entry of **ON ALL LAY** Must precede any Layer Mask command.* ▪ *Add a list or range of numbers representing layer to be processed to the specified mask file to ensure they are NOT considered in the translation e.g.:* ***OFF LAY 114,149,166,167,168***

JT to STEP General Arguments

The image below shows the General arguments that are available, with their default settings:



The option is described below:

Option	Description
Mass Properties	This causes the translator to calculate and print mass properties information for the created solid bodies. The printed output appears in the progress file. This needs to be selected if validation properties are required in the output STEP file. o Command Line Syntax ▪ *mprops*
Advanced	Allows any of the Command Line Advanced arguments documented below to be passed to the Unified Interface invocation

JT to STEP Advanced Arguments

Theorem's JT to STEP translator has been configured with default settings that optimises the translation process. However, there are times when a satisfactory result cannot be obtained, so it may be required to deploy one or more Advanced Arguments to improve the translated result.

The following table describes useful Advanced Arguments that can be entered into the General Tab -> Advanced field:

Option	Description
pmi_attr	Adds validation properties for polyline PMI into the STEP file.
assy_val_props	Adds validation properties for assembly entities into the STEP file.

Appendix A – A GUIDE TO THE CM-DATA DEFINITION FILE

To conform to the published standard, a STEP file must contain some Configuration Management (CM) data and may also contain a geometrical representation of a part or Assembly. Whilst the Theorem translators are supplied to allow the use of STEP AP203, STEP AP214 and STEP AP242 with designed geometry there is a minimum subset of CM data which must be included in the STEP file in order to comply with the standard.

In addition to the minimum subset to satisfy the rules for the AP, there is also a level of support for data within the Units of Functionality (UoF) defined which must be provided to allow the processor to be fully conformant for those UoFs. The minimum level of support implies that the user has a means of defining the data items for those UoFs rather than that the processor will populate them with default values.

This document defines the format of a text file by which this CM data can be supplied to the translator for inclusion with geometry in a STEP part 21 file.

The data required is that which is additional to the geometrical definition and structure and which is necessary to allow the STEP part 21 file created to contain the minimum subset of Configuration Management data in order to satisfy all of the rules of the AP.

Optional input allows the processor to provide full support for the UoFs "identification" and "authorization".

The remainder of this document deals mainly with the CM Data input for the AP203 translator, as the AP214 and AP242 standards do not require the same level of minimum data. For AP214 and AP242, the requirement is only for the Product, Version and Definition data to be present.

This document is divided into sections each of which will give a different perspective on the data to be provided.

The first section provides an overview of the data and file construction.

The second section defines the structure of the file in more detail and describes each entity and attribute. Note that the descriptions given imply a certain interpretation of the AP203 schema and structure and might be described or used differently according to the internal culture and terminology of any enterprise.

The third section defines the mapping to the AP203 schema entities and attributes at AIM (Application Interpreted Model) level.

The fourth section defines the mapping to the AP203 ARM (the Application Reference Model). Not all of the entities and attributes in the file are defined in the ARM but all are required to satisfy the AP203 rules related to the ARM information used.

The fifth section is a sample file as used within the PDES Stepnet testing process.

The sixth section defines the defaults used for each entity where the relevant data is not provided.

Overview

The data pertains to a PART and to a specific VERSION and DEFINITION of that PART. The PART must have an owner (the design owner). The VERSION must have been created, supplied, approved and classified. The classification must have an approver. The DEFINITION or the specific design of this version must have a creator and must be approved.

Thus the basis for the data is a series of actions (which includes the required actions of approving other actions). Actions require a "performer" and may have a "date" of action attached.

A "performer" must be defined as a person and the organization to which they belong. Thus the "performer" is defined within the context of a person and organization combination.

A "date" here is defined as the combination of a date and time, the time being specified relative to the appropriate local time zone.

The data is therefore split into 2 sections.

The first section allows basic units of information which might be required more than once within the part related data to be defined. These are organizations, people and time zones. This is referred to as as the library section below.

The second section allows the specific part information to be supplied and uses the data from the first section.

Structure and Mapping

The data is differentiated into header records and attribute records. Header records begin with a # and define the start of a block of attribute information. The block name defines the type of data following and is in upper case characters (eg #ORG, #PERSON, #CLASSIFICATION).

Attribute records begin with a $ which is followed by the attribute name in lower case characters. This name is followed by a colon which is then followed by the data. The data may contain blank characters. Where multiple items are data are defined (eg $date) the items are delimited by a comma

e.g. *$person_id : Person-1 $purpose : Because I want to $date : edt,1993,7,17,13,45,20*

The majority of the information defined is textual. Where it is other than text data this is specified in the description for the attribute in this section.

Where AP203 restrictions apply the relevant possible list of options is given. If the data is text information in AP203 then the user may choose to use an alternative value and no

check is made that the restriction rule has been satisfied. NB for make_or_buy the data within the AP203 schema is defined with an enumeration and so only the values listed may be used.

Where appropriate, recommendations from the AP203 Recommended Practices document have been given.

The base information for the CM data consists of people, organizations, addresses and date - times.

The format allows for the input of a number of organizations, each one initiated via the #ORG record and for which a unique $org_id must be provided.

Where people are referred to this is done within the context of an organization so the organizations are defined first. Subsequently each person block is defined with a reference via org_id back to the relevant organization.

The persons are initiated via a #PERSON record for which a $org_id record must be used to identify the relevant organization and also for which a unique $person_id record must be provided which identifies the person and organization combination.

Date and time items are defined within the context of a time zone so the time zone(s) data is defined first. For each time zone, initiated with a #TIME_ZONE record, a unique name must be defined. Time zone descriptions below are done in relation to Greenwich Meant Time (GMT). For the purposes of this data GMT is deemed to be equivalent to Coordinated Universal Time which is the formal reference time for AP203.

Persons and organizations may have addresses. Addresses must be defined with an address_id before they can be referred to within a person or org definition. They are linked via the address_id.

Following the input of the base information the data relevant to the specific product may be defined. The #DATA record delineates between organization, person and time_zone data and the product data.

Each set of data for the product may refer back to persons via the person_id. Date items refer to the relevant time zone via the given name.

Each item used in this file is flagged as mandatory (m) or optional (o). Note that in most cases the omission of a mandatory attribute will cause an error message to be given but a default value will be assigned to the attribute to allow processing to continue.

The library section of the file contains #ORG, #PERSON and #TIMEZONE blocks.

Mandatory (o) Optional(o)	***#ADDRESS***
m	$address_id address identifier
o	$internal_location
o	$street_number

o	$street
o	$postal_box
o	$town
o	$region
o	$postal_code
o	$country
o	$facsimile_number
o	$telephone_number
o	$electronic_mail_address
o	$telex_number

NB! at least one optional parameter must be set

Mandatory (o) Optional(o)	***#ORG***	***Description***
m	m $org_id	unique organization identifier
m	m $org_name	the name of the organization
o	o $org_description	a description of the organization
o	o $address_id	organization address identifier

Mandatory (o) Optional(o)	***#PERSON***	***Description***
m	m $org_id	the organization to which the person belongs in this context
m	m $person_id	unique person id
m	m $first_name	person's first name
m	m $last_name	person's last name
o	o $middle	person's middle names if any exist
o	o $prefix	person's prefix titles if any exist
o	o $suffix	person's suffix titles if any exist
o	o $address_id	person address identifier

#TIME_ZONE

$name&offset	**the time zone in which the activity dates are defined is specified relative to GMT. A name for the time zone by which the dates provided will refer to it is given together with the hours difference from GMT.** **Positive hour offset values apply to time zones which are ahead (ie east) of the meridian. Negative time zones refer to time zones which are behind the meridian (ie west of the meridian)** **Optionally a minute offset can be added.** **Format is :** **name,hour** **or** **name,hour,minute** **name - may not contain commas.** **hour - is an integer number.** **minute - is an integer number.**

The end of the library section and start of the data section is defined with a block header #DATA

When dates are defined in this section it is actually a date and time which has to be provided. The syntax for a date record is as follows.

$date : time-zone-name,year,month,day,hour,minute,second

Mandatory (o) Optional(o)	***#DATE***	***Description***
m	time-zone-name	name of the time zone defined in the library section
m	year	the year (integer number)
m	month	the month of the year (integer number)
m	day	the day of the month (integer number)
m	hour	the hour of the day (integer number)
o	minute	the minute of the hour (integer number)
o	second	the second of the minute (real number)

Each field is delimited with a comma.

The data in this section can be considered in 3 categories:

1) Information about the part
2) Information about version of the part
3) Information about definition of the part

The data section can specify information related to the part via the following block headers grouped according to these 3 categories:

#PRODUCT
#DESIGN_OWNER

#PRODUCT_VERSION
#PRODUCT_SOURCE
#DESIGN_SUPPLIER
#VERSION_CREATION
#VERSION_APPROVAL
#CLASSIFICATION
#CLASSIFICATION_APPROVAL

#PRODUCT_DEFINITION
#DESIGN_CREATOR
#PRODUCT_APPROVAL

Definitions and attributes for these are as follows:

Mandatory (o) Optional(o)	***Data***	***Description***
	#PRODUCT	defines the part
m	$product_name	part name
m	$product_id	part number
o	$description	part description
o	$product_type	part_type be assembly, inseparable_assembly, detail or customer_furnished_equipment
o	$product_class	part_classification must be assembly, inseparable_assembly, detail, customer_furnished_equipment cast, coined, drawn, extruded, forged, formed, machined, molded, rolled or sheared
o	$context_name	name for product_context (AP203 mechanical_context)
o	$category_name	optional additional product category Products may have additional category information. Each category may have a description. The additional category will be related to the product with a product_category_relationship which may have a name and description. A category name introduces a new category definition. Subsequent category information will pertain to that category until another category name is found. Note for standard part indicator set as standard_part
o	$category_desc	product_category.description for preceding category
o	$category_rel_name	product_category_relationship name for

		preceding category linking to product_related_product_category
o	$category_rel_desc	product_category_relationship.description for preceding category_rel

	#DESIGN_OWNER	defines the owner of the part
m	$person_id	the id of the person defined in the library section
	#PRODUCT_VERSION	defines the version of the part
m	$version	the revision letter
o	$description	a description for this version
	#PRODUCT_SOURCE	defines whether the version is made or bought.
m	$make_or_buy	may be MADE, BOUGHT or NOT_KNOWN

	#DESIGN_SUPPLIER	defines the supplier of the part version
m	$person_id	the id of the person defined in the library section

	#VERSION_CREATION	defines the person who initiated the part version
m	$person_id	the id of the person defined in the library section

	#VERSION_APPROVAL	approval for this version
m	$person_id	the approver
m	$status	status of the approval must be one of: • Approved • not_yet_approved • disapproved • withdrawn
m	$purpose	the purpose of the action
m	$date	the date when the action took place

	#CLASSIFICATION	defines the security classification for the version
m	$person_id	the classification officer
m	$status	the classification level must be one of • unclassified • classified • proprietary • confidential • secret • top_secret
m	$purpose	the purpose of the classification
m	$name	the name for the classification if the version is defined with a status of classified, this could be the company name for the

		classification eg "secret restricted"
m	$date	the date of classification

	#CLASSIFICATION_APPROVAL	approval for the security classification
m	$person_id	the approver
m	$status	status of the approval must be one of • approved • not_yet_approved • disapproved • withdrawn
m	$purpose	the purpose of the action
m	$date	the date when the action took place

	#PRODUCT_DEFINITION	defines the actual design definition of this version
m	$description	a description for this definition
o	$id	identifier for product_definition
o	$life_cycle_stage	for AP203 must be set as design
o	$context_name	design discipline context name
o	$shape_name	name for product definition shape
o	$shape_description	description for product definition shape

	#DESIGN_CREATOR	defines the person who created this definition
m	$person_id	the id of the person defined in the library section
m	$date	the date when the design creation took place

	#PRODUCT_APPROVAL	approval for this product definition
m	$person_id	the approver
m	$status	status of the approval must be one of • approved • not_yet_approved • disapproved • withdrawn
m	$purpose	the purpose of the action
m	$date	the date when the action took place

The file is terminated with an #END record

1. Mappings to the AIM (The Schema)

Mandatory (o) Optional(o)	**#ORG**	***Description***
m	$org_id :	organization.id
m	$org_name :	organization.name
o	$org_description :	organization.description

Mandatory (o) Optional(o)	**#PERSON**	
m	$org_id : $organization.id :	person_and_organization
m	$person_id :	person.id
m	$first_name :	person.first_name
m	$last_name :	person.last_name
o	$middle :	person.middle_names
o	$prefix :	person.prefix_titles
o	$suffix :	person.suffix_titles

Mandatory (o) Optional(o)	**#ADDRESS**	NB! at least one optional parameter must be set
o	$internal_location	address.internal_location
o	$street_number	address.street_number
o	$street	address.street
o	$postal_box	address.postal_box
o	$town	address.town
o	$region	address.region
o	$postal_code	address.postal_code
o	$country	address.country
o	$facsimile_number	address.facsimile_number
o	$telephone_number	address.telephone_number
o	$electronic_mail_address	address.electronic_mail_address
o	$telex_number	address.telex_number

Mandatory (o) Optional(o)	**#TIME_ZONE**	***Description***
m	$name&offset :	file-id-name,hour,minute file-id-name is the name used within this file
m	coordinated_universal_time_offset.hour_offset	The sign of this value defines ahead_or_behind+ve values -> ahead-ve values -> behind (so Eastern Standard Time = -5)
o	coordinated_universal_time_offset.minute_offset	where for any date below
	$date :	
m	file-id-name,year,month,day,hour,minute,second	date
m	file-id-name -> time_zone-> coordinated_universal_time_offset	
m	year : calendar_date.year_component	year

	month : calendar_date.month_component	month
m	day : calendar_date.day_component	day
m	local_time.hour_component	hour
o	local_time.minute_component	minute
o	local_time.second_component	second

#DATA

Mandatory (o) Optional(o)	**#PRODUCT**	***Description***
m	$product_name :	product.name
m	$product_id :	product.id
o	$description	product.description
o	$product_type :	product_related_product_category.name must be: • Assembly • inseparable_assembly, detail • customer_furnished_equipment
o	$product_class:	product_related_product_category.name must be one of: • product_type list • cast • coined • drawn • extruded • forged • formed • machined • molded • rolled or sheared
o	$context_name	product_context.name (AP203 mechanical_context.name)
o	$category_name	product_category.name. Note for standard part indicator set as standard_part
o	$category_desc	product_category.description for preceding category
o	$category_rel_name	product_category_relationship.name
o	o $category_rel_desc	product_category_relationship.description

Mandatory (o) Optional(o)	**#DESIGN_OWNER**	***Description***
m	$person_id :	cc_design_person_and_organization_assignment.assigned_person_and_organization where: cc_design_person_and_organization_assignment.role ->

		person_and_organization_role.name = "design_owner" cc_design_person_and_organization_assignment.items[0] = product

Mandatory (o) Optional(o)	**#DESIGN_VERSION**	***Description***
m	$version :	product_definition_formation_with_specified_source.id
o	$description	product_definition_formation_with_specified_source.description

Mandatory (o) Optional(o)	**#DESIGN_SOURCE**	***Description***
m	$make_or_buy :	product_definition_formation_with_specified_source.make_or_buy NB must be BOUGHT, MADE or NOT_KNOWN

Mandatory (o) Optional(o)	**#DESIGN_SUPPLIER**	***Description***
m	$person_id :	cc_design_person_and_organization_assignment.assigned_person_and_organization where: cc_design_person_and_organization_assignment.role -> person_and_organization_role.name = "design_supplier" cc_design_person_and_organization_assignment.items[0] = product_definition_formation_with_specified_source

Mandatory (o) Optional(o)	**#VERSION_CREATION**	***Description***
m	$person_id :	cc_design_person_and_organization_assignment.assigned_person_and_organization where: cc_design_person_and_organization_assignment.role -> person_and_organization_role.name = "creator" cc_design_person_and_organization_assignment.items[0] = product_definition_formation_with_specified_source

Mandatory (o) Optional(o)	**#VERSION_APPROVAL for approval**	***Description***
	product_definition_formation_with_specified_source	

m	$person_id :	approval_person_organization.person_organization
m	$status :	approval_status.name
m	$purpose :	approval.level
m	$date :	approval_date_time.date_time and cc_design_date_and_time_assignment.assigned_date_and_time where: cc_design_date_and_time_assignment.role -> date_time_role.name = "sign_off_date" cc_design_date_and_time_assignment.items[0] = approval_person_organization for VERSION_APPROVAL

Mandatory (o) Optional(o)	**#CLASSIFICATION**	***Description***
m	$person_id :	cc_design_person_and_organization_assignment.assigned_person_and_organization where: cc_design_person_and_organization_assignment.role -> person_and_organization_role.name= "classification_officer" cc_design_person_and_organization_assignment.items[0] = security_classification
m	$status :	security_classification_level.name
m	$purpose :	security_classification.purpose
m	$name :	security_classification.name
m	$date :	cc_design_date_and_time_assignment. assigned_date_and_time where: cc_design_date_and_time_assignment.role date_time_role.name = "classification_date" .items[0] = security_classification

Mandatory (o) Optional(o)	**#CLASSIFICATION_APPROVAL approval for security_classification**	***Description***
m	$person_id :	approval_person_organization.person_organization
m	$status :	approval_status.name
m	$purpose :	approval.level
m	$date :	approval_date_time.date_time and cc_design_date_and_time_assignment. assigned_date_and_time where: cc_design_date_and_time_assign

		ment.role -> date_time_role.name = "sign_off_date" cc_design_date_and_time_assignment.items[0] = approval_person_organization for CLASSIFICATION_APPROVAL

Mandatory (o) Optional(o)	**#PRODUCT_DEFINITION**	***Description***
m	$description :	product_definition.description
o	$id :	product_definition.id
o	$life_cycle_stage	product_definition_context.life_cycle_stage using product_definition_context subtype design_context
o	$context_name	product_definition_context.name
o	$shape_name	product definition shape.name
o	$shape_description	product definition shape.description

Mandatory (o) Optional(o)	**#DESIGN_CREATOR**	***Description***
m	$person_id :	cc_design_person_and_organization_assignment.assigned_person_and_organization where: cc_design_person_and_organization_assignment.role -> person_and_organization_role.name = "creator" cc_design_person_and_organization_assignment.items[0] = product_definition
m	$date :	cc_design_date_and_time_assignment.assigned_date_and_time where: cc_design_date_and_time_assignment.role -> date_time_role.name = "creation_date" cc_design_date_and_time_assignment.items[0] = product_definition

Mandatory (o) Optional(o)	**#PRODUCT_APPROVAL approval for product_definition**	***Description***
m	$person_id :	approval_person_organization.person_organization
m	$status :	approval_status.name
m	$purpose :	approval.level
m	$date :	approval_date_time.date_time and cc_design_date_and_time_assignment. assigned_date_and_time where:

		cc_design_date_and_time_assignment.role -> date_time_role.name = "sign_off_date" cc_design_date_and_time_assignment.items[0] = approval_person_organization for PRODUCT_APPROVAL

#END

2. Mappings to the ARM

Note that there are many attributes which don't map directly to the ARM of AP203 but need to be defined to satisfy the entity definitions or AP203 rules based on other ARM requirements.

Only those elements of the file format definition which relate directly to the ARM are listed here.

The data is indented in order to denote a dependency on a previous block. eg DESIGN_OWNER is dependent on PRODUCT, CLASSIFICATION_APPROVAL is dependent on CLASSIFICATION.

ARM mappings are as follows.

Mandatory (o) Optional(o)	**#ORG**	***Description***
	#ORG	PERSON_ORGANIZATION.organization
	#PERSON	PERSON_ORGANIZATION.person
m	$org_id :	
m	$person_id :	PERSON_ORGANIZATION.person_organization_id
	#ADDRESS	Address
Mandatory (o) Optional(o)	**#PRODUCT**	***Description***
m	$product_name :	PART.part_nomenclature
m	$product_id :	PART.part_number
o	$product_type :	PART_TYPE
o	$product_class :	PART_CLASSIFICATION
o	$category_name :	Standard part indicator = 'standard_part'
Mandatory (o) Optional(o)	**#DESIGN_OWNER Is The Owner Of PART**	***Description***
m	$person_id :	PERSON_ORGANIZATION
	#PRODUCT_VERSION	
m	$version :	PART_VERSION.revision_letter
	#PRODUCT_SOURCE	PART_VERSION.make_or_buy_code

	#DESIGN_SUPPLIER Missing From ARM	
m	$person_id :	PERSON_ORGANIZATION
	#VERSION_CREATION Is The Creator Of PART_VERSION	
m	$person_id :	PERSON_ORGANIZATION
	#VERSION_APPROVAL Approves PART_VERSION	
m	$person_id :	PERSON_ORGANIZATION (APPROVAL is Authorized By...)
m	$status :	APPROVAL.status
m	$purpose :	APPROVAL.purpose
m	$date :	APPROVAL.date
	#CLASSIFICATION	
m	$person_id :	PERSON_ORGANIZATION (Classification Officer)
m	$status :	PART_VERSION.security_code
	#CLASSIFICATION_APPROVAL approval for security_classification	
m	$person_id :	PERSON_ORGANIZATION(CLASSIFICATION is approved by)
	#PRODUCT_DEFINITION	
m	$description :	DESIGN_DISCIPLINE_PRODUCT_DEFINITION.description
o	$life_cycle_stage	discipline_id life cycle stage
o	$context_name	discipline_id name
	#DESIGN_CREATOR Is The Creator Of DESIGN_DISCIPLINE_PRODUCT_DEFINITION	
m	$person_id :	PERSON_ORGANIZATION
	$date :	DESIGN_DISCIPLINE_PRODUCT_DEFINITION.creation_date
	#PRODUCT_APPROVAL Approves DESIGN_DISCIPLINE_PRODUCT_DEFINITION	
m	$person_id :	PERSON_ORGANIZATION (APPROVAL is Authorized By...)

3. Sample File

In this sample file empty records have been used to provide easier reading but the blank lines are not required for an actual file and would be ignored.

#ADDRESS

$address_id : NVI-Address
$region : Michigan
$country : USA

#ADDRESS
$address_id : White-Address
$town : Detroit
$country : USA
$electronic_mail_address : KWhite@NVI.com

#ORG
$org_id : NVI-Michigan
$org_name : New Ventures, Inc.
$address_id : NVI-Address

#ORG
$org_id : CDI
$org_name : Contract Design, Inc

#PERSON
$org_id : NVI-Michigan
$person_id : 333-003
$last_name : White
$first_name : K.
$middle : H.
$address_id : White-Address

#PERSON
$org_id : NVI-Michigan
$person_id : 555-005
$last_name : Johnson
$first_name : Stephen
$prefix : Mr.

#PERSON
$org_id : NVI-Michigan
$person_id : 666-006
$last_name : Spock
$first_name : Robert
$middle : S.

#PERSON
$org_id : NVI-Michigan
$person_id : 222-002
$last_name : Pierre
$first_name : Gabrielle

#PERSON
$org_id : NVI-Michigan
$person_id : 777-007
$last_name : Black
$first_name : James

#PERSON
$org_id : CDI
$person_id : CDI-2

$last_name : Smith
$first_name : B.

#TIME_ZONE
$name&offset : edt -5

#DATA

#PRODUCT
$product_name : Fillet Boot
$product_id : 22222

#PRODUCT_VERSION
$version : A

#PRODUCT_SOURCE
$make_or_buy : BOUGHT

#DESIGN_OWNER
$person_id : 333-003

#DESIGN_SUPPLIER
$person_id : CDI-2

#DESIGN_CREATOR
$person_id : 222-002
$date : edt,1993,7,19,19,46,55

#VERSION_CREATION
$person_id : 222-002

#PRODUCT_DEFINITION
$description : detailed drawing as planned for STEP conformance testing

#VERSION_APPROVAL
$person_id : 777-007
$status : approved
$purpose : Approved as initial STEP test case part
$date : edt,1993,7,17,13,29,52

#PRODUCT_APPROVAL
$person_id : 777-007
$status : approved
$purpose : Approved as STEP conformance testing drawing
$date : edt,1993,7,19,19,47,51

#CLASSIFICATION_APPROVAL
$person_id : 666-006
$status : approved
$purpose :Approved as unclassified STEP test data
$date : edt,1993,7,17,13,47,28

#CLASSIFICATION
$person_id : 555-005
$status : unclassified
$purpose : open availability of STEP test data

```
$date : edt,1993,7,17,13,45,20
$name : 1993-C1

#END
```

4. Defaults Definition

This describes the full set of defaults used for the data attributes which may be defined. However, once an organization has been defined then that organization becomes the default. Once a person has been defined then that person becomes the default person (and organization).

Any unset date will be defaulted to the current time and date, defined as YYYY,MM,DD,HH below

```
#ORG
$org_id : Default_oid
$org_name : Default_org
#PERSON
$org_id : Default_oid
$person_id : Default_pid
$last_name : Doe
$first_name : John
#TIME_ZONE
$name&offset : tsc_gmt,0
#DATA
#PRODUCT
$product_name : product name
$product_id : product id
$product_type : detail or assembly dependent on data
#PRODUCT_VERSION
$version : version 0
#PRODUCT_SOURCE
$make_or_buy : NOT_KNOWN
#DESIGN_OWNER
$person_id : Default_pid
#DESIGN_SUPPLIER
$person_id : Default_pid
#DESIGN_CREATOR
$person_id : Default_pid
$date : tsc_gmt,YYYY,MM,DD,HH
#VERSION_CREATION
$person_id : Default_pid
#PRODUCT_DEFINITION
$description : unknown
#VERSION_APPROVAL
$person_id : Default_pid
$status : not_yet_approved
$purpose : unknown
$date : tsc_gmt,YYYY,MM,DD,HH
#PRODUCT_APPROVAL
$person_id : Default_pid
$status : not_yet_approved
$purpose : unknown
$date : tsc_gmt,YYYY,MM,DD,HH
```

#CLASSIFICATION_APPROVAL
$person_id : Default_pid
$status : not_yet_approved
$purpose :unknown
$date : tsc_gmt,YYYY,MM,DD,HH
#CLASSIFICATION
$person_id : Default_pid
$status : unclassified
$purpose : unknown
$date : tsc_gmt,YYYY,MM,DD,HH
#END

Appendix B – JT Configuration File

Introduction

A configuration file contains the settings for your translations. The configuration file can be specified using the command line option –config or -z.

If this is not supplied the following config file will be used:-

tessGCO.config in TS_INST\etc directory (TS_INST = Installed directory)

The JT configuration file contains various sections, each containing different settings based on the section.

The Setup Section

The setup options in the configuration file define how your files are translated. The setup section is the first part of the configuration file and contains a series of standard translator options.

To edit setup options

1. Open an existing configuration file with a text editor.
2. Edit the configuration file options listed in the table below.
3. Save the configuration with a .config extension

Option name	Keywords	Example
EAITranslator	EAITranslator {	EAITranslator {
OutputDirectory	"*path to directory*"	OutputDirectory = "/home/<user>/"
CommonPartsPath	"*path to directory*"	CommonPartsPath= "/myaccount/jtparts/"
chordalOption	"RELATIVE" "ABSOLUTE"	chordalOption = "RELATIVE"
structureOption	"PER_PART" "MONOLITHIC" "FULL_SHATTER"	structureOption = "MONOLITHIC"
WriteWhichFiles	"ALL" "ASSEMBLY_ONLY" "PARTS_ONLY"	WriteWhichFiles = "ALL"
compression	true TRUE false FALSE	compression = true
triStripOpt	true TRUE false FALSE	triStripOpt = false
seamSewing	true TRUE false FALSE	seamSewing = true
seamSewingTol	*any integer*	seamSewingTol = 0.001
includeBrep	true TRUE	includeBrep = false

	false FALSE	
brepPrecision	"SINGLE" "DOUBLE"	brepPrecision = "SINGLE"
autoNameSanitize	true TRUE false FALSE	autoNameSanitize = true
updateChangedPartsOnly	true TRUE false FALSE	updateChangedPartsOnly = false
verboseReporting	true TRUE false FALSE	verboseReporting = false
writeAsciiAssembly	true TRUE false FALSE	writeAsciiAssembly = false
singlePartsNoAssem	true TRUE false FALSE	singlePartsNoAssem = false
smartLODgeneration	true TRUE false FALSE	smartLODgeneration = true
autoLowLODgeneration	true TRUE false FALSE	autoLowLODgeneration = true
numLODs	*any integer*	numLODs = 3
JtFileFormat	64,70,80,81,82,90, 91,92,93,94,95,100	JtFileFormat = “95”
includeULP	PASSTHROUGH TRUE FALSE	includeULP = “PASSTHROUGH”
ulpPrecision	Real Value	ulpPrecision = 0.001
close brace	}	}

The Level of Detail Section

The level of detail section of the configuration file contains the tessellation and simplification information for each level of detail in the file.

This section consists of several sets of level of detail (LOD) information, and the number of these sets depends on the number you specified on the numLODs line in the configuration file.

To edit level of detail options

1. Open an existing configuration file in a text editor.
2. Edit the configuration file options listed below.
3. Save the configuration with a .config extension

Option name	Keywords	Example
LOD	LOD "*lod number*" {	LOD "1" {
Level	*any integer*	Level = 1
Chordal	*any number*	Chordal = 0.001
Angular	*any number*	Angular = 25
Length	*any number*	Length = 1
Label	*any string*	"ud_FINE"
FeatureSuppression	*any integer*	FeatureSuppression = 0
Simplify	*any number*	Simplify = 0.60
AdvCompressionLevel	*any number*	AdvCompressionLevel = 0.0
ULP	true false	
close brace	}	}

The Filter Section

The filter section of the configuration file contains the filename and metadata filtering information. Edit this section if you want to change how the translator sanitizes filenames and filters metadata keys.

To edit filter options

1. Open an existing configuration file with a text editor.
2. Edit the configuration file options from the table below.
3. Save the configuration with a .config extension

Option name	Keywords	Example
Filter	Filter {	Filter {
FilenameSanitizeSet	*"string of*	FilenameSanitizeSet = "abc123."

	characters"	
FilenameSanitizeSetAdd	*"string of characters"*	FilenameSanitizeSetAdd = "4l"
FilenameSanitizeSetDelete	*"string of characters"*	FilenameSanitizeSetDelete = "c"
MetadataKey	*"string of characters"*	MetadataKey = "*metadata key to exclude*"
close brace	}	}

The Metadata section

The metadata section sets which metadata to attach to all parts, assemblies and nodes of the model.

Note: Be sure to add these options to the configuration file in pairs: one line to define the metadata key and one line to define the metadata value.

To edit metadata options

1. Open an existing configuration file (.CONFIG) in a text editor.
2. Edit the configuration file options shown in the table below.
3. Save the configuration with a .config extension

Option name	Keywords	Example
Metadata	Metadata {	Metadata {
AddToParts	*"string of characters"*	AddToParts = "<metadata key>" AddToParts = "<metadata value>"
AddToAssemblies	*"string of characters"*	AddToAssemblies = "<metadata key>" AddToAssemblies = "<metadata value>"
AddToAllNodes	*"string of characters"*	AddToAllNodes = "<metadata key>" AddToAllNodes = "<metadata value>"
close brace	}	}

The Special Section

The special section of the configuration file contains lines that are unique to this translator.

To edit special options

1. Open an existing configuration file with a text editor.
2. Edit the configuration file options shown in the table below.
3. Save the configuration with a .config file extension.

Option	Keyword	Example	Default Value
GcoOptions	GcoOptions {	GcoOptions {	
ReportFilename	Full system file path	Windows example ReportFilename = P:\caddata\translation\result\part55	Windows system C:%TEMP%\tscprogressyi
OutputUnits	mm millimetres cm centimetres	OutputUnits = mm	inputUnits

	m metre metres inches feet yards inputUnits		
StructureOutputType	JT PLMXML PLMXMLJT	StructureOutputType = JT	JT
PLMXMLPropertyMappingFile	*File Name*	PLMXMLPropertyMappingFile = “mapping_file.txt”	“”
brepType	XT JT XTJT	brepType = XT	JT
ParasolidTolerantModelling	true/TRUE false/FALSE	ParasolidTolerantModelling = true	true
ParasolidTolerantModellingFactor	*Any positive integer*	ParasolidTolerantModellingFactor = 3	3
SewParasolidBodies	true/TRUE false/FALSE	SewParasolidBodies = true	true
SewParasolidBodiesTol	*Any number*	SewParasolidBodiesTol = 0.01	0.01
IncrementalSewing	true/TRUE false/FALSE	IncrementalSewing = true	true
IncrementalSewingNoOfIterations	true/TRUE false/FALSE	IncrementalSewingNoOfIterations = 5	5
ExplodeSolidstoFaces	true/TRUE false/FALSE	ExplodeSolidstoFaces = false	false
SplitDiscontinuousSurfaces	true/TRUE false/FALSE	SplitDiscontinuousSurfaces = true	false
ForceBodyCreation	true/TRUE false/FALSE	ForceBodyCreation = true	true
FixDegenerateEdges	true/TRUE false/FALSE	FixDegenerateEdges = true	true
FaceEdgeTol	*Any number*	FaceEdgeTol = 0.000006	0.000006
FixSmallFeaturesSolids	true/TRUE false/FALSE	FixSmallFeaturesSolids = false	false
FixSmallFeaturesOpenSolids	true/TRUE false/FALSE	FixSmallFeaturesOpenSolids = false	false
SimplifyGeometry	true/TRUE false/FALSE	SimplifyGeometry = false	false

BrepWireframe	true/TRUE false/FALSE	BrepWireframe = true	true
ProduceTessellatedOutput	true/TRUE false/FALSE	ProduceTessellatedOutput = false	false
ExpandPart	true/TRUE false/FALSE	ExpandPart = false	false
ReuseSolids	true/TRUE false/FALSE	ReuseSolids = false	false
CADPropertyMappingFile	*File Name*	CADPropertyMappingFile = “mapping_file.txt”	“”
SavedViewsViewSetName	*"string of characters"*	SavedViewsViewSetName	"SavedViews"
AnnotationPlanesViewSetName	*"string of characters"*	AnnotationPlanesViewSetName	"AnnotationPlanes"
close brace	}	}	

Appendix C – Property Mapping Files

Property mapping files are required for CAD property Mapping and PLMXML Property Mapping

A Property Mapping File is a comma separated text file containing information of how CAD properties from the source system will be mapped into the target file.

The format is as follows:

- Lines beginning with a "#" are treated as comment lines and are ignored.
- Any space characters will be treated as part of the item
- Lines containing a mapping must contain 6 items separated by 5 commas
- The six items are :-

Item	Description
Source name	The attribute name in the Source System
Target name	The attribute name in the Target File
Data derived from	0 - Do not convert 1 - Use the source value as given 6 - Use the source value as given and hide the property ***Note!*** *Value 6 For CAD Mapping Files ONLY (Not PLMXML)*
Default Value	Not currently used
Value Type	Not currently used
Default Units	Not currently used

An Example of a mapping file is shown below:-

```
# Mapping from input attribute name to Target property name
#
# Line Format:-
# Source name,Target name,Data derived from,Default Value,Value Type,Default Units
#    Data derived from:-
#    0 - Do not convert
#    1 - Use the source values as given
#    6 - Use the source value as given and hide the property
#
_ActivateBOM,NULL,0,0,,
_LastModifier,NULL,0,0,,
_Maturity,NULL,0,0,,
_PrdVersion,NULL,0,0,,
_ReferenceTimeStamp,NULL,0,0,,
_Responsible,NULL,0,0,,
COG M,ud_CAD_CENTER_OF_GRAVITY,1,0,,
COMPONENTS PRINCIPAL AXES ,NULL,0,0,,
```

DENSITY Kg/M^3,NULL,0,0,,
INERTIA MATRIX KgM2,ud_CAD_MOMENT_OF_INERTIA,1,0,,
INERTIA VOLUME M^3,ud_CAD_VOLUME,1,0,,
INERTIA WET AREA M^2,ud_CAD_SURFACE_AREA,1,0,,
MASS Kg,ud_CAD_MASS,1,0,,
PRINCIPAL MOMENTS KgM^2,NULL,0,0,,
FILESAVETIME,File Last Modified,1,0,,
LOCALE,LOCALE,1,0,,
Masterdata Version,Masterdata Version,1,0,,
Material Details,Material Details,1,0,,
PART_NUMBER,PART_NUMBER,1,0,,
MPARTNAME,Source Model Name,1,0,,
Source,SourceName,1,0,,