Theorem Solutions

JT Validation Utility

USER GUIDE

Document Revision: 4.0

Issued: 02/08/2018

Theorem Solutions Limited,
Theorem House, Marston Park,
Bonehill Road, Tamworth,
Staffordshire,
B78 3HU,
England

Telephone: +44 (0) 1827 305 350
Fax: +44 (0) 1827 692 63
E-mail: sales@theorem.com

Website: http://www.TheoremSolutions.com

Document Control:

Status:	Final
Revision:	4.0
Date of issue:	02/08/2018

Amendment Summary:

Issue	Date	Commentary	Initials
1.0		Final for customer release	
2.0		Updated document for 19.3 product release	TL
3.0		Updated document format	TL
4.0-	02/08/18	Updated for Theorem 21.x release	TL

Table of Contents

INTRODUCTION

The Theorem Solutions JT Validation Utility extends the functionality of the output created by the Siemens JTInspector tool. The output from the utility is written to a structured XML based report file.

In addition to reporting on the contents of the JT file the utility also supports the following addition functionality;

- Data verification based upon point cloud surface checking points input from a data file.
- Mass property comparison based upon data entered via an input data file
- A new JT output file will be created – the format of this file can be controlled by using a JT configuration file. Alternatively default parameters will be used if a configuration isn't specified.

Command Line Syntax

jt_validation.cmd <input_jt_filename> <output_jt_filename> [Optional Arguments]

Note the output JT file should be a different file to the input file.

Optional Arguments		Description
check_points_file	***<filename>***	*See "Example Cloud of Points File" below for details.*
mass_props_file	***<filename>***	*See "Example Mass Property File" below for details.*
config_file	***<filename>***	*Configuration file used when exporting the file referenced by "output_jt_filename" (If omitted then Siemens defaults are applied)*
error_file	***<filename>***	*Adds a line to the supplied csv file when a part has body checking errors. To be used in conjunction with the "mend_bodies" option to generate a summary spreadsheet for a set of JT files. Each line contains details of the errors, before and after the attempted healing. See below for details.*
mend_bodies		*Invokes Parasolid healing functions on any bodies that have body checking errors*
progress		*Turns on progress file output that is normally suppressed. Intended to be used when diagnosing problems*

Example Output

Note the items marked in Red are additional information reported by the Theorem jt_validation utility compared to the output generated by the Siemens JT Inspector. All output would be consistent with the JT Inspector report.

```
<JTInspector>
   <Summary>
      <ReportingUnits>
         <Length value = "m"/>
         <Mass value = "kg"/>
      </ReportingUnits>
      <JTContent>
         <Geometry>
            <XTB-Rep> - a count of each topological and geometric entity in the part, plus
                        details of the NURBs data.
            </XTB-Rep>
            <JTB-Rep> - a count of each topological and geometric entity in the whole part,
                        plus details of the NURBs data.
            </JTB-Rep>
            <ULP> - empty tag
            </ULP>
            <Wireframe> - count of wireframe edges
            </Wireframe>
         </Geometry>
         <Tessellation> - details of LODs and average tessellation parameters
         </Tessellation>
         <PMI> - counts of PMI entities
         </PMI>
         <MetaProperties value = "24"/> - Total number of properties attached to nodes
         <XTB-RepPhysicalProperties> - Mass Property totals
         </XTB-RepPhysicalProperties>
      </JTContent>
      <Geometry> - Totals for parts passing/failing the Parasolid body checks at the 4 levels.
      </Geometry>
      <PMIValidity> - "Validity" of the PMI including counts of polylines, polygons, persisted ids and
                      associations
      </PMIValidity>
      <Fidelity> - Checks on calculated mass properties against standard attributes containing mass
                   properties e.g. CAD_MASS.
      </Fidelity>
   </Summary>
   <Details> - Each node in detail in hierarchical form
      <Assembly/Part  value = "Assembly Name">
         <JTContent>
            <PMI/> - PMI Details
            <MetaProperties value = "7"> - Property details
            </MetaProperties>
         </JTContent>
         <PMIValidity> - PMI Validity details
         </PMIValidity>
         <Part value = "Part Name">
            <JTContent>
               <Geometry>
```

```
        <XTB-Rep>  - a count of each topological and geometric entity in the part, plus
                     details of the NURBs data
        </XTB-Rep>
    </Geometry>
    <BoundingBox> - Bounding box stored in part
        <Minimum value="( 1.540869, -4.605161,  3.988309) m" />
        <Maximum value="( 6.720977, -2.452496,  6.938706) m" />
    </BoundingBox>
    <CalculatedBoundingBox> - Bounding box calculated using Parasolid
        <Minimum value="( 1.540869, -4.605161,  3.988309) m" />
        <Maximum value="( 6.720977, -2.452496,  6.938706) m" />
    </CalculatedBoundingBox>
    <Tessellation> - LOD and tessellation parameter details
    </Tessellation>
    <PMI>
    </PMI>
    <MetaProperties value = "10"> Print of node properties
    </MetaProperties>
    <XTB-RepPhysicalProperties> - Calculated mass properties
    </XTB-RepPhysicalProperties>
</JTContent>
<Geometry>  - Body checking details. I've left the fail types in there so you can see which
              errors  in which level.  It colours each body green, and then colours any bad
              faces red. Corrupt bodies are coloured all red. It also adds error locations as
              points into the JT file if Parasolid returns the information.
    <Level0>
        <BodyStructureCorrupt/>
    </Level0>
    <Level1>
        <CurveGeometryInvalid/>
        <DegenerateCurveGeometry/>
        <MissingEdgeGeometry/>
        <CloseKnotValuesonB-Curve/>
        <SelfIntersectingCurveGeometry/>
        <SurfaceGeometryInvalid/>
        <NonG1CurveGeometry/>
        <DegenerateSurfaceGeometry/>
        <MissingFaceGeometry/>
        <CloseKnotValuesonB-Surface/>
        <SelfIntersectingSurfaceGeometry/>
        <NonG1SurfaceGeometry/>
    </Level1>
    <Level2>
        <VertexNotonCurve/>
        <BadWire/>
        <EdgeOrientationReversed/>
        <OpenCurveonRingEdge/>
        <EdgeNotonSurface/>
        <MissingVertexonSingularity/>
        <BadEdgeOrder/>
        <SelfIntersectingFace/>
        <VerticesTouchEdge/>
```

```
                    <InconsistentLoops/>
                  </Level2>
                  <Level3>
                    <BadShells/>
                  </Level3>
                </Geometry>
                <Fidelity> - Mass properties stored as CAD_MASS etc.
                </Fidelity>
                <PMIValidity> - PMI Validity details.
                </PMIValidity>
              </Part>
            </Assembly>
          </Details>
          <Result value = "Passed"/>
        </ JTInspector>
```

Example Mass Property File

The following data is an example Mass Property file that can be used as an input the validation process.

```
<?xml version="1.0" encoding="UTF-8"?>
<MassProps>
<Part value = "eng_oilp">
<SurfaceArea value = "185129.0"/>
<MassUnits value = "Kilograms"/>
<Mass value = "0.000564"/>
<Volume value = "564000.0"/>
<Density value = "0.000000001"/>
<CenterofGravity value = "-405.923 -22.929 -14.861"/>
<Part/>
<MassProps/>
```

Example Cloud of Points File

The following data is an example Cloud of Points file that can be used as an input to the validation process.

```
<?xml version="1.0" encoding="UTF-8"?>
<CheckPoints>
<Part value = "eng_oilp">
<Body value = "0">
<Face ident = "5">
<Point x = "-456.3" y = "-160.3" z = "-94.55"/>
<Point x = "-421.2" y = "-124.5" z = "-9.5"/>
<Point x = "-371.5" y = "125.5" z = "87"/>
<Face/>
<Body/>
<Body value = "1">
<Face ident = "325">
<Point x = "-456.3" y = "-160.3" z = "-94.55"/>
<Point x = "-421.2" y = "-124.5" z = "-9.5"/>
<Point x = "-371.5" y = "125.5" z = "87"/>
<Face/>
<Face ident = "125">
<Point x = "-456.3" y = "-160.3" z = "-94.55"/>
<Point x = "-421.2" y = "-124.5" z = "-9.5"/>
<Point x = "-371.5" y = "125.5" z = "87"/>
<Face/>
<Body/>
<Part/>
<CheckPoints/>
```

Example Error File Output

A line of output in the Error File consists of five or six comma separated entries:-

1. The File Name
2. The Part Name
3. The number of body checking errors before healing
4. The Parasolid Tokens for the errors
5. The number of body checking errors after healing
6. The Parasolid Tokens for the errors (missing if 5 is zero)

Example:-

```
JT01-SDPlate-0018-AX01--.jt,Solid,2, PK_BODY_state_invalid_ident_c
PK_FACE_state_self_int_c,0
```

Additionally there will be extra output in the XML file if the “mend_bodies” option is in force, for example:-

```
<PreMendFaults value = "10"/>
<Mergen value = "Entities merged via MERGEN on tag 93"/>
<Menden value = "Successful mend on body 93 : 6 entities fixed"/>
<PostMendFaults value = "5"/>
```