i-UR CityGML Geometry Validator

1. Introduction:

This FME workspace has been developed to validate LoD1 and LoD2 solid building geometries. The workspace performs several geometry checks and outputs a json formatted report. In addition to geometry checks the workspace will also perform an overlap check for each building feature in the tested dataset(s).

The workspace only supports checks on solid primitives. Other primitives are not currently supported. 2D Overlap checks are only performed on Building features (including their sub-components e.g., Building Parts and Building Installations).

FME is a proprietary software tool which specializes in Geodata processing. For more information about FME including free training and community support please visit safe.com.

The *CityGMLValidation.fmw* contains process specific information to help understand the process. For more information on the process itself and how it works please open *CityGMLValidation.fmw* in FME Workbench.

In addition, a wrapper workspace, *runner.fmw*, has also been included to help with batch processing – read more in Section 4 below.

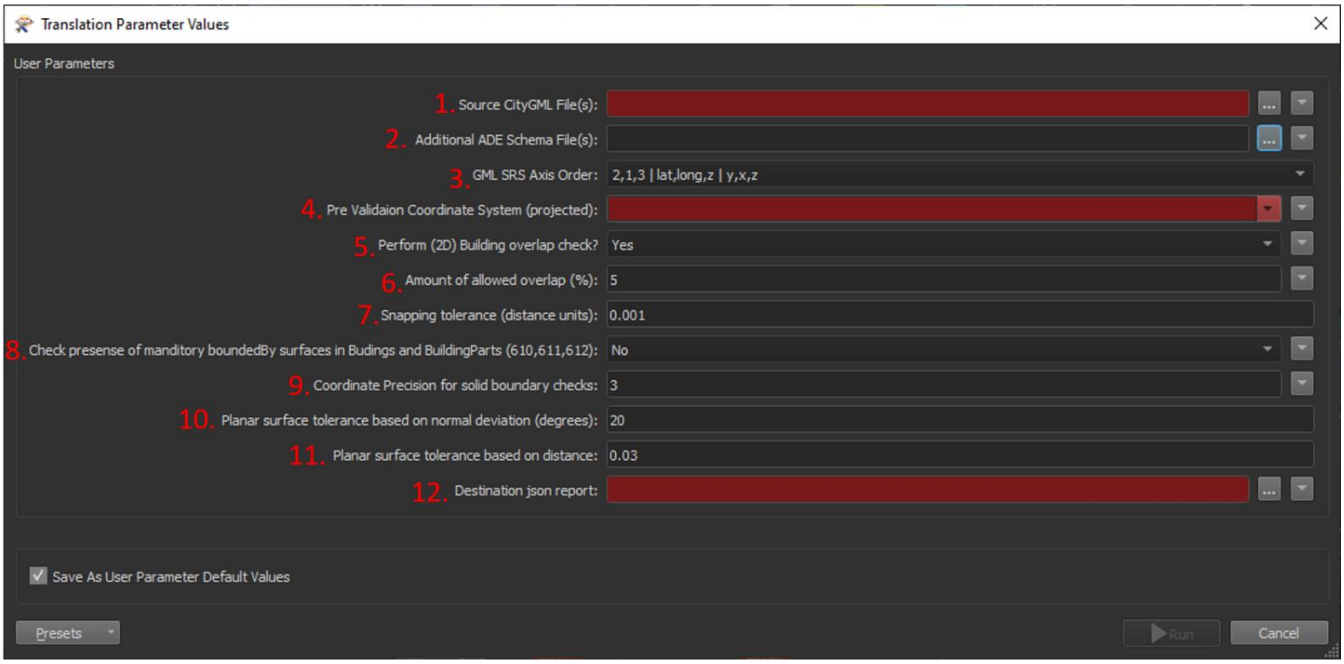
2. Requirements:

FME Desktop 2020.1+ (to get a trial license please visit safe.com)

3. Usage and input:

The workspace can be run in several ways including from within the FME Workbench application itself and via the command line. For more information on how to run an FME Workspace with FME Desktop please refer to this page of the FME documentation. More detailed instructions on how to run FME from the command line can be found here;

The workspace has several required inputs which need to be set for each run of the workspace which are defined below.


Translation Parameter Values

User Parameters

1. Source CityGML File(s):

2. Additional ADE Schema File(s):

3. GML SRS Axis Order: 2,1,3 | lat,long,z | y,x,z

4. Pre Validaion Coordinate System (projected):

5. Perform (2D) Building overlap check? Yes

6. Amount of allowed overlap (%): 5

7. Snapping tolerance (distance units): 0.001

8. Check presense of manditory boundedBy surfaces in Budings and BuildingParts (610,611,612): No

9. Coordinate Precision for solid boundary checks: 3

10. Planar surface tolerance based on normal deviation (degrees): 20

11. Planar surface tolerance based on distance: 0.03

12. Destination json report:

☑ Save As User Parameter Default Values

Figure 1: Input FME parameters with defaults – red indicates a required input

	Name	Options	Prompt	Description
1	**INPUT**	**Full or relative paths. Wildcards (*) are also supported**	Source CityGML File(s)	The workspace supports multiple input CityGML files including data from within zip files and across multiple folders. For information on how to add multiple files, add file filters and read data form within zip files please see the following section from the FME Desktop Documentation. All features will be read in and processed together. This allows for the overlap check as well as a check for duplicates to be performed against multiple files. The file path will be included for each file and primitive in the output json report **Note** if using the *runner.fmw* the syntax is different (see section 4 below)
2	**ADE_XSD_DOC**	**Full or relative paths. Wildcards (*) are also supported**	Additional ADE Schema File(s)	If the dataset includes any ADE extensions which are not part of the Core CityGML schema then the xsd files can be specified here. If the xsd path is defined in the xml header and the process has access to the schema files, then this field can be skipped.
3	**SRS_AXIS**	**1,2,3/2,1,3**	GML SRS Axis Order	This relates to the order in which the coordinates are ordered in the CityGML file. Typically, data are stored as Long, Lat, z (x, y, z) however, in some cases data are stored as Lat, Long, z (y, x, z).
4	**COORD**	**String – recommended**	Pre-Validation Coordinate	This workspace supports input data which are both in Geographic (Lat Long) and Geometric (x,

		in the form: EPSG:XXXX	System (projected)	y), however, in order to perform correct validation, data which are in Geographic systems must first be reprojected into an appropriate geometric coordinate system. The process will fail is a coordinate system is selected which has units of degrees.
5	**OVER_CHECK**	**LoD1|LoD2|ALL|NO**	Perform (2D) Building overlap check?	This controls how the process will perform overlap checks on buildings. Elements which make up each building (Building Parts, Building Installations and boundedBy surfaces are aggregated together and then forced into 2D. This 2D representation is then dissolved into a single polygon feature per building). This process is fairly CPU and memory intensive and depending on the number of buildings in the datasets(s) this will have a negative effect on performance.
6	**OVERLAP**	**Float (0-100)**	Amount of allowed overlap (%)	This can be used to adjust how strict the overlap check is. The check is percentage based.
7	**TOLERANCE**	**Float >=0**	Snapping tolerance (distance units)	This parameter controls when a point is determined to be in the same location. A larger value will typically result in more reported errors. A value of 0 will only treat exact points as being duplicates. Note, this parameter is only used when determining duplicated points or Self-Intersecting (or self-touching) rings.
8	**BOUNDEDBY**	**Yes/No**	Check presence of mandatory boundedBy surfaces in Buildings and Building Parts (610,611,612)	It is expected that Lod2 Buildings and Building parts also contain at least one Wall Surface, Ground Surface, and Roof Surface. If this is set to “Yes” then this test will be performed. This test is optional because it may be the case that some Building Parts are intentionally modelled without either a roof or ground surface.
9	**PREC**	**Int (0-10)**	Coordinate Precision for solid boundary checks	This parameter will round the coordinates prior to checks on solid boundaries. The number specified here will determine the coordinate precision, effectively aligning coordinates to a grid. Any duplicate points on a ring created because of this rounding will be removed prior to the solid check. Note, this parameter only effects tests on Solid Boundaries (ERROR Codes 3XX).
10	**PLAN_ANG**	**Float (0-180)**	Planar surface tolerance based on normal deviation (degrees)	The surface normal deviation tolerance is specified in degrees and describes the maximum deviation from the average surface normal that a component can have before it is considered non-planar. A planar polygon has a surface normal deviation of 0. A non-planar polygon will have its average surface normal computed using

				Newell’s method, and the surface normal deviation of each part from the average surface normal will be computed with the cosine law.
11	**PLAN_DIST**	**Float >=0**	Planar surface tolerance based on distance ('thickness' of plane)	The thickness tolerance is specified in ground units and describes the maximum “thickness” a plane can have before it is considered non-planar. A planar polygon has a thickness of 0. A non-planar polygon will have its average surface normal computed using Newell’s method, and its thickness will be determined in the direction of the normalized surface normal.
12	**OUTPUT**	**Full or relative File path including the filename and .json extension**	Destination json report	The output location of the json report. **Note** if using the *runner.fmw* the syntax is different (see section 4 below)

Example Command-line to run this workspace:

```
"C:\Program Files\FME_20238\fme.exe" C:\Validation\CityGMLValidation.fmw
               --INPUT """C:\Validation\Sample\53394507_bldg_6697.gml"
               "C:\Validation\Sample\53394517_bldg_6697.gml"""
              --ADE_XSD_DOC ""
              --SRS_AXIS "2<comma>1<comma>3"
              --COORD "EPSG:6677"
              --OVER_CHECK "LoD2"
              --OVERLAP "5"
              --TOLERANCE "0.001"
              --BOUNDEDBY "No"
              --PREC "3"
              --PLAN_ANG "20"
              --PLAN_DIST "0.03"
              --OUTPUT "C:\Validation\report.json"
```

4. Running in Batch mode:

FME has several methods to run jobs in a batch mode, this can be a good option to reduce the overall memory consumed by the workspace. Read about the basic options here.

These basic methods, however, would not be suitable where the input is a zip file. Additional preprocessing would be required to create a file listing needed to create a batch file.

Within FME Desktop there is a tool called a WorkspaceRunner which is able control the running of sub processes. Details on how to use the WorkspasceRunner works can be found here.

Included in this package is a workspace file called *runner.fmw* which includes a WorksapceRunner transformer. This workspace can be used to validate the CityGML (.xml or .gml) on a per file bases and allows for up to 7 sub processed to run in parallel.

The *runner.fmw* will produce one output file for each input CityGML. In addition, overlap checking is therefore limited to the data included in that file only.

Calling the *runner.fmw* via the command line requires adjustment on the input parameters and includes additional paraments:

	Name	Options	Prompt	Description
1	**INPUT**	**Full or relative path to a zip file**	Source zip file	The path to a single .zip file. This zip will be extracted and files with either a .xml or .gml extension will be used as input for the *CityGMLValidation.fmw* tool. Please be aware that while the input zip can have sub folders, the output will not preserve the folder structure
12	**OUTPUT**	**Output folder (full or relative)**	Destination json report folder	The output folder for the validation reports. The output reports will be named by their filename inside the zip file. The folder structure inside the zip file is not preserved and any files which share the name filename will have their validation reports overwritten.
13	**MAX_PROC**	**Int (1-7)**	Maximum Number of Concurrent Processes (1-7):	This controls the maximum number of sub processed which can be run in parallel. The number is capped at 7 due to licensing.
14	**WORKSPACE_FILE**	**Full or Relative file path to *CityGMLValidation***	CityGML Validation path:	The path to the actual validation workspace. This workspace will be run once for each CityGML file.
15	**LOG**	**Output folder (Existing)**	Choose output logfile folder:	Output location for FME log files of the subprocess called by *runner.fmw*

Example Command-line to run the runner:

```
"C:\Program Files\FME_20624\fme.exe" C:\Validation\runner.fmw
              --INPUT "Sample\Sample.zip"
              --ADE_XSD_DOC ""
              --SRS_AXIS "2<comma>1<comma>3"
              --COORD "EPSG:6677"
              --OVER_CHECK "LoD2"
              --OVERLAP "5"
              --TOLERANCE "0.001"
              --BOUNDEDBY "No"
              --PREC "3"
              --PLAN_ANG "20"
              --PLAN_DIST "0.03"
              --OUTPUT "OutputFolder"
              --MAX_PROC "7"
              --WORKSPACE_FILE "CityGMLValidation.fmw"
              --LOG "logs"
```

Please be note that if using relative paths these should be relative to the active directory (not the *runner.fmw* workspace. It is recommended to use full paths if at all possible.

5. Understanding the results:

The output json report is roughly modeled on the structure produced by val3dity. A summary of detected ERRORS is reported followed by a summary of input parameters.

Results are reported by feature grouped by top level city object. Geometry specific errors are reported at the primitive level nested inside the feature. For example, a Building feature may be made up of several Solid primitives - a lod1Solid Building, a lod2Solid Building and a lod2Solid Building Installation. These solid primitives will be reported separately nested inside the feature. Only features and primitives with ERROS are reported in the result.

6. The ERRORS:

ERROR codes relating to geometry (1XX-4XX), wherever possible, are taken form val3dity. This provides some consistency when comparing results, however, in several cases the errors are not equivalent. Differences are outlined in the following table.

ERROR Code	ERROR Description	Comment
Geometry Errors – Linear Ring Level		
100	Contains NaNs, Infinities, Null Geometry Parts or -0	The FME data model supports exotic coordinate values such as NaN and -0 which may be present in datasets. This will check for these.
102	Duplicate Consecutive Points	Reported when points are detected to be in the same place. Influenced by parameter 7.
104	Self-Intersections in 2D	Influenced by parameter 7 – This check also includes Self-Touching rings.
105	Degenerate or Corrupt Geometries	This will be returned when a rings geometry collapses into a line or point for a given Snapping Tolerance (parameter 7).
Geometry Errors – Polygon Level		
201	Donut: Overlapping or Touching Rings	
202	Donut: Duplicate Rings	
203	Non Planar Polygon: Thickness	
204	Non Planar Polygon: Normal Deviation	
205	Donut: Disjoint Interior	
206	Donut: Hole Outside Shell	
207	Donut: Nested Hole	
208	Incorrect Orientation	
Geometry Errors – Shell Level		
300	Not a Valid 2-Manifold	
301	Not Enough Faces	
302	Surface Not Closed	

305	Multiple Connected Components	
306	Surface Self Intersects	
307	Face Wrong Orientation	Faces can have an invalid orientation if they have a donut area where the normal of any inner boundary has the same direction as the normal of the outer boundary.
308	Surface Wrong Orientation	Composite surfaces can have an invalid orientation if they have two parts that share an edge and those parts are not consistently oriented with respect to each other.
309	Vertices Not Used	
Geometry Errors – Solid Level		
401	Shells Intersect	
402	Duplicate Shells	
403	Inner Shell Outside Outer	
404	Interior of Shell Not Connected	
405	Incorrect Solid Orientation	
Top Level Feature Errors		
609	Building with no Solid primitive	This error is returned when a Building contains no Solid Primitives.
610	LoD2 Building No GroundSurface	This is returned when a Building (or Building Part) has no Ground Surface. Test is optional
611	LoD2 Building No WallSurface	This is returned when a Building (or Building Part) has no Wall Surface. Test is optional.
612	LoD2 Building No RoofSurface	This is returned when a Building (or Building Part) has no Roof Surface. Test is optional.
614	Feature Overlap in 2D	This is retuned is the components making up a Building overlap another Building. Test is optional.
617	LoD2 Solid contains unlinked boundedBy Surfaces	This indicates that a surfaceMember which makes up a lod2Solid is not xlinked. This error is reported at the primitive level.
619	LoD2 Solid contains Invalid xlinked surface	For solids which ***do*** have xlinked surfaces. This checks to make sure that these surfaces are ‘reasonable’ surfaces. Reasonable means that the feature role of the surface is a boundedBy surface. This error is reported at the primitive level.
Dataset Errors		
905	Duplicate gml:id	A check is performed against the entire dataset for duplicate gml:id’s. Note that this does not check the gml:id’s of geometry elements but rather at the feature level.

7. Additional Resources

FME has a lot of available free online resources including prerecorded training sessions, live online training sessions, topic specific webinars and tutorials. There is also a very active user community. For professional FME Assistance you can reach out to a verified partner.

For the fastest understanding of FME it is recommended to Complete the Introduction to FME Desktop training followed by the Advanced FME Desktop training. These will introduce a new user to the concepts and possibilities of FME. If completed back to back over a few days this should kick start any FME based project.

For more 3D specific content please review the following:

- Getting Started with 3D (Tutorial)
- 3D Transformations (Tutorial)
- Working with 3D Webinar
- BIM Tutorial (Tutorial)

If English is a barrier, content and support in Japanese can be found here at https://fmesupport.com/ which is hosted and provided by Pacific Spatial Solutions.

8. A Note on FME Server

The Bulk or Batch processing options are discussed above only applicable to FME Desktop. FME Server has its own set of tools for batch processing. It should be noted that FME Server (depending on the license type) does not have a limitation to the number of parallel jobs which can be run on a single server and can therefore be an attractive option if performance is important.