# A ROADMAP FOR GENERATING SEMANTICALLY ENRICHED BUILDING MODELS ACCORDING TO CITYGML MODEL VIA TWO DIFFERENT METHODOLOGIES

G. Floros*, D. Solou, I. Pispidikis and E. Dimopoulou

School of Rural and Surveying Engineering, National Technical University of Athens, 9 Iroon Polytechneiou str, 15780 Zografou, Greece (flwrosg@gmail.com; di.solou@gmail.com; pispidikisj@yahoo.gr; efi@survey.ntua.gr)

**KEY WORDS**: 3D Modeling, Trimble SketchUp, CityEngine, FME, CityGML, 3DCitiesProject, 3DCIM, Semantics

**ABSTRACT:**

The methodologies of 3D modeling techniques have increasingly increased due to the rapid advances of new technologies. Nowadays, the focus of 3D modeling software is focused, not only to the finest visualization of the models, but also in their semantic features during the modeling procedure. As a result, the models thus generated are both realistic and semantically enriched. Additionally, various extensions of modeling software allow for the immediate conversion of the model's format, via semi-automatic procedures with respect to the user's scope. The aim of this paper is to investigate the generation of a semantically enriched Citygml building model via two different methodologies. The first methodology includes the modeling in Trimble SketchUp and the transformation in FME Desktop Manager, while the second methodology includes the model's generation in CityEngine and its transformation in the CityGML format via the 3DCitiesProject extension for ArcGIS. Finally, the two aforesaid methodologies are being compared and specific characteristics are evaluated, in order to infer the methodology that is best applied depending on the different projects' purposes.

## 1. INTRODUCTION

Virtual 3D city models have been used in the past, mainly for the visualization or graphical exploration of cityscapes. Nowadays, an increasing number of applications like environmental and training simulations, urban planning and facility management, disaster management and homeland security, and personal navigation require additional information about the city objects given in a standardized representation (Kolbe, 2009). Additionally, the increasing urbanization, the rapid growth of urban areas, and the development of mega cities, are among the important changes occurring in the world. With this increasing urbanization, more than half of the world population will stay in urban areas, counting about 5 billion houses (United Nations, 2008). Therefore, developing new techniques in order to manage the complex urban environment seems to be necessary, along with the development of semantic 3D models, revealing the interrelations of the spatial components of the cities (Sadidi, 2015). Semantics have gradually attracted international scientific interest due to their ability of storing data that describe relations between different object parts and their environment (Diakité et al., 2014). This results in a semantically enriched visualization of the reality, or even in a semantic 3D city model. Semantic 3D city models comprise, besides the spatial and graphical aspects, the ontological structure including thematic classes, attributes, and their interrelations. The result is that the semantic modeling of cities requires the appropriate qualification of 3D data (Gröger & Plümer, 2012). Current research and applications focus on the semantic enrichment of distinctive city objects or 3D geometries which can be decomposed into their structural elements including attributes and their correlations. The semantic modeling approach along with the application of 3D geometry and topology of real-world objects is realized by the CityGML open data model (Kolbe, 2009). However, questions arise about the most effective way to complement the geometries' semantics or how to efficiently extract semantics from pure geometric models (Zhu et al., 2011), issues which can be addressed by the concept of interoperability. The rapid growth of 3D modeling methodologies and techniques, as well as the development of various 3D formats, have certainly assisted in this direction. Various tools of conversion have been developed. However, none of the converters is currently capable of automatically creating valid geometries nor fully correct semantics, especially when high levels of details are involved (Donkers, 2013). Isikdag and Zlatanova (2009) define a framework automating generation of buildings from BIM to CityGML, claiming that it is possible to define rules for geometrical transformation and facilitation of semantic matching from IFC to CityGML models, De Laat and van Berlo (2011) describe the development of GeoBIM extension on CityGML for IFC data, while El-Mekawy et al., (2011) proposed a unified building model (UBM) for integration of IFC and CityGML, allowing bilateral transformation between the two models. Dimopoulou et al., (2014) investigate integration and interoperability options between procedural modeling techniques and BIM-ready software within the CityGML framework from a semantic viewpoint using ESRI CityEngine environment and Trimble SketchUp Pro software to create 3D building models and evaluate modeling techniques, and Floros et al., (2015) investigate the semantic functionality of conversion methodologies with Trimble SketchUp and FME up to Level of Detail 4. This paper explores two different methodologies for generating a CityGML Model in LoD3, while presenting guidelines for the two semi-automatic procedures pointing out the necessary steps of the conversion. The structure of the paper is as follows: in Section 2, the basic characteristics of each methodology are presented, as well as the comparison between the generated CityGML models. In section 3, a case study that employs the two methodologies in the built environment is

* Corresponding author

presented and evaluated in terms of semantics coherence, model's geometry, topology and appearance (Kolbe, 2009). Section 4 presents the comparison between the two methodologies by the creation of specific attributes that are used to characterize quantitative various parts of the two procedures. Finally, the last section highlights the results of the comparison and discusses further research.

## 2. METHODOLOGY

### 2.1 Methods and Software

This paper focuses on the generation of LoD 3 Buildings according to the CityGML Standard with the use of commercially available software. More specifically, two different modeling approaches and conversion processes are investigated and presented. The area of interest is a building that is located in the municipality of Chalandri in Athens. The software used for modeling are Trimble SketchUp and ESRI CityEngine, while for the conversion in CityGML format, the FME Workbench and the 3DCIM CityGML toolbox were used. Secondly, as soon as the models are generated, an evaluation of the two above-mentioned procedures is explicitly presented. The evaluation delves into fields such as the abilities of the software in modeling and converting, their cost and their potential for expansion among others. The purpose of this research is not only to analyse two optimal ways for creating a CityGML Building, but also to present the benefits and the disadvantages of each procedure applied in different scientific fields.

### 2.2 Procedures

The first methodology describes the basic concepts on generating a LoD3 CityGML model through Trimble SketchUp and FME's Workbench. A semi-automatic procedure is presented and specific instructions that focus on the maximum simplicity and the maximum effectiveness of the procedure in terms of compatibility with the CityGML Standard are highlighted.
In regard to the second methodology, a 3D building model is initially implemented in ESRI's software CityEngine and then is automatically transformed into a CityGML model, by using the tools of 3DCIM CityGML toolbox. The methodological steps followed are elaborately presented in order to address interoperability options and guarantee a lossless transformation.

The workflows of the two procedures are presented in Fig.1 and Fig. 2 respectively:

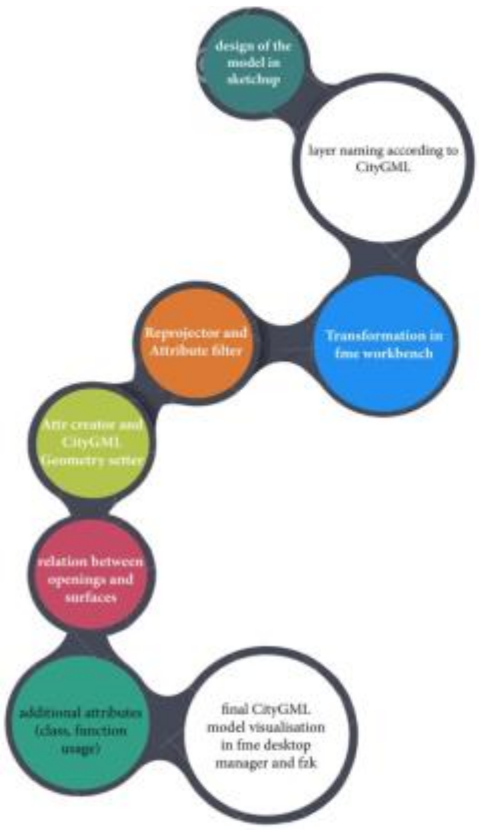

design of the model in sketchup
layer naming according to CityGML
Transformation in fme workbench
Reprojector and Attribute filter
Attr creator and CityGML Geometry setter
relation between openings and surfaces
additional attributes (class, function usage)
final CityGML model visualisation in fme desktop manager and fzk

Figure 1: Workflow of the 1st methodology

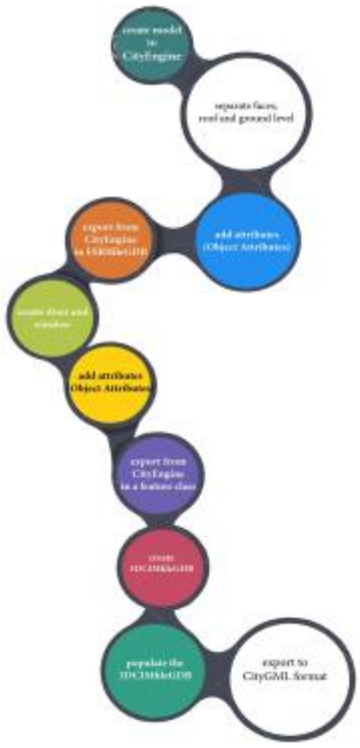

create model in CityEngine
separate faces, roof and ground level
add attributes (Object Attributes)
export from CityEngine in ESRIfileGDB
create door and window
add attributes Object Attributes
export from CityEngine in a feature class
create 3DCIMfileGDB
populate the 3DCIMfileGDB
export to CityGML format

Figure 2: Workflow of the 2nd methodology

Based on the results of the two above mentioned approaches, a comparison between the procedures and the generated CityGML models is performed and resented. The comparison focuses on aspects such as: technical background knowledge, time-efficiency, comprehensiveness of the model in terms of semantics and geometry, possibilities of further exploitation of the model, availability of the software and data shown, limits and constraints of the procedures etc. The comparison is based on a ranking order of the involved attributes, and when completed, the optimal procedure is presented.

## 3. CASE STUDY

### 3.1 Model generation with Trimble SketchUp and FME

The generated CityGML model consists of two concrete, although relevant between each other procedures: the design of the 3D building model in Trimble SketchUp 2016 and the transformation of the designed building model into a CityGML building model with the use of FME Software. For the specific model, LoD 3 was achieved as maximum Level of Detail. For the CityGML Model, the following methodological steps have been elaborated.

**3.1.1 Modelling in Trimble SketchUp:** Firstly, during the design of the model, specific characteristics had to be considered to ensure its compatibility for the transformation to a CityGML model. Specifically, after the design of the model completes, the user creates the layers that correspond to each object in the model. In order to achieve the highest possible time-efficiency, labels that fully describe the objects that are also included in the CityGML standard should be assigned. The format of these labels include the Level of Detail, the specific Building and the feature types, e.g. LoD3_Bldg1_WallSurface1_Window1, LoD3_Bldg1_RoofSurface, etc. The labels could include less information, although the aim of this methodology is to present an optimal way for creating a CityGML model that may be consisted of more than one features. As a result, the user is able to identify easily and quickly every object in the model.
The second major step in modeling with Trimble SketchUp is the connection of every boundary surface with its Openings (Door, Windows). The latter is achieved, by distinguishing each and every WallSurface that includes an opening and creating two new layers that include the Surface and the Opening(s) (Window, Door) separately, as shown in Figure 3. Moreover, a layer named LoD3_Bldg1, is created in order to store all the geometry that is generated by the SketchUp software that is specifically used for design purposes and is not related with the model's correct CityGML transformation in the next stage.

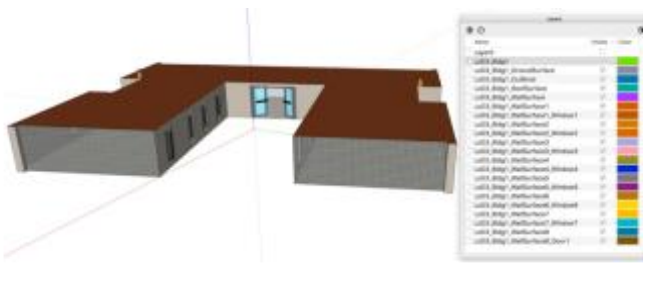


Figure 3: Assignment of Layers in Trimble SketchUp

The following table, presents the matching, between the SketchUp layers naming and the features in CityGML Standard.

| *SketchUp Layer Naming* | *CityGML Feature Type* |
|---|---|
| LoD3_Bldg1_GroundSurface | GroundSurface |
| LoD3_Bldg1_RoofSurface | RoofSurface |
| LoD3_Bldg1_WallSurface, LoD3_Bldg1_WallSurface8 | WallSurface |
| LoD3_Bldg1_WallSurface8_Door1 | Door |
| LoD3_Bldg1_WallSurface1_Window1-WallSurface7_Window7 | Window |

Table 1: Co-relation between layers and CityGML Features

Also, in order to achieve the greatest realism possible, textures from the appearance of the building in reality were assigned to the designed model, as shown in figure 4.
Finally, as soon as the model is completed, Trimble SketchUp, provides the ability of Geolocation. Although, it is achievable only in the Coordination System of Google Earth, this is an important information that the user must consider during the transformation to a CityGML Model through the FME on later stages.

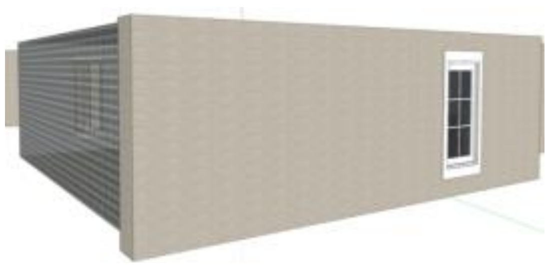


Figure 4: Exterior WallSurface of the designed building

**3.1.2 Transformation via FME Software:** The transformation of the designed Building model into a CityGML Building Model with the use of FME Software is described as follows. The workbench presented, is able to transform every SketchUp model, up to LoD 3, into a CityGML Building Model. Most of the procedure is executed without any outer intervention from the user, although there are certain modifications the user shall apply, depending on the imported SketchUp model.
Firstly, the user is able to change the Coordination system of Google Earth, to the system that fits the purposes of the model, through the *Reprojector* transformer.
Secondly, in order to confirm the correctness of the generated CityGML Building model, the workbench executes tests, that remove all the unnecessary geometries that might have been created during the design in Trimble SketchUp and the CityGML Standard does not require them.
As soon as the coordination system and the model's geometry is accurate, the transformer AttributeFilter (figure 5) is being used and the user adds the attributes labelled exactly as the layers in SketchUp. By the time the layers are added, the user can manually connect every attribute in the AttributeFilter Transformer, with the corresponding CityGML Feature Type. In case the model includes Openings, it is crucial that the user connects the Surface, not only with the specific CityGML Feature Type, but also with the FeatureMerger (Supplier) Transformer. This is necessary, in order to identify the semantic hierarchy between the surfaces and the openings.

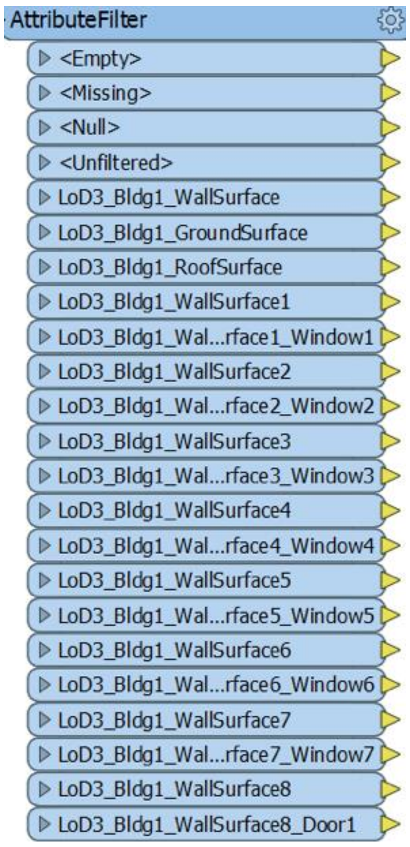


Figure 5: The AttributeFilter Transformer

Then, by utilizing the *AttributeCreator* and the *CityGMLGeometrySetter* transformer, the user is able to define the attributes as well as the geometries and their Levels of Detail, depending on the scope of the model.
If the model includes openings, the workbench executes a routine that assures that each opening belongs to the appropriate surface, based on the CityGML Standard, as shown in figure 6. The FeatureMerger Transformer is consisted of the Requestor and the Supplier. In order to merge the assigned WallSurface with its Opening, the user creates two tables in both AttributeCreator Transformers. The tables have the Title: Join and the same Value numbered e.g as1, 2, etc. Every FeatureMerger hase the same Title, but never the same Value.

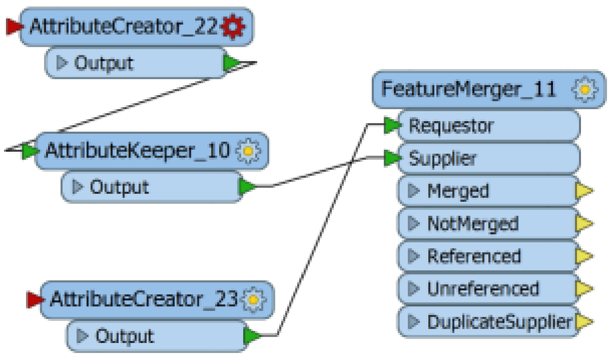
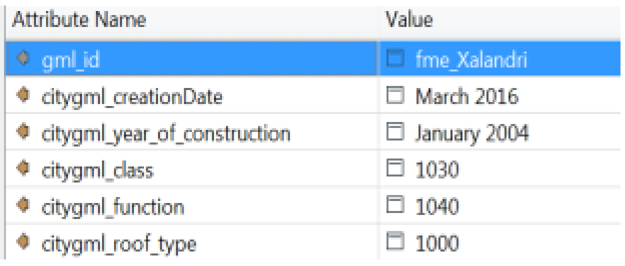


Figure 6: Connection between the Openings and the Surfaces

The CityGML feature types are already inside the Workbench, so the user will be able to choose the necessary ones for the model. Lastly, it is possible to add attributes such as class, function, usage, year of Construction, etc., through the distinct *AttributeCreator* transformer that is connected with the Building feature, as shown in figure 8.



| Attribute Name | Value |
|---|---|
| gml_id | fme_Xalandri |
| citygml_creationDate | March 2016 |
| citygml_year_of_construction | January 2004 |
| citygml_class | 1030 |
| citygml_function | 1040 |
| citygml_roof_type | 1000 |

Figure 7: Addition of attributes in the model

As soon as the workbench is finalized and the user runs the routine, by right clicking a random feature type, it is possible to view the Building in the FME Data Inspector, as shown in figure 8.

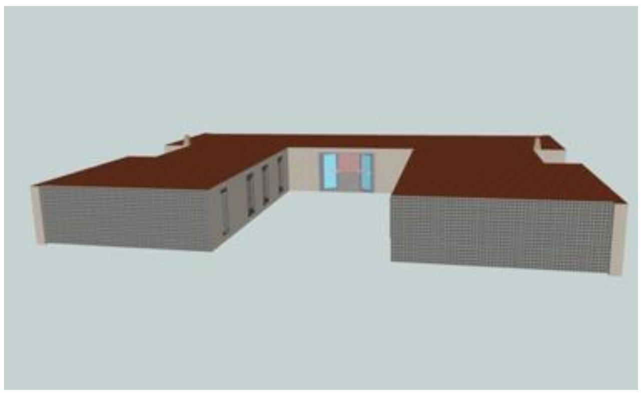


Figure 8: Visualisation in FME Data Inspector

### 3.2 Model's generation with ESRI's CityEngine and managing with 3DCities Project for ArcGIS

The following methodology consists of two clearly separated but fully compatible steps. Firstly, the creation of the model in ESRI's CityEngine and then the semantic enrichment and the transformation to a CityGML model by using the tools of 3DCities Project for ArcGIS. A detailed guideline of these procedure is further explained.

**3.2.1 Modeling in CityEngine:** the first step to create a model in the CityEngine environment is to import the building footprint (figure 9). All the necessary information was stored to the Object Attributes fields, according to the attributes that the *Building* feature class of the 3DCIM determines, as shown in figure 10.

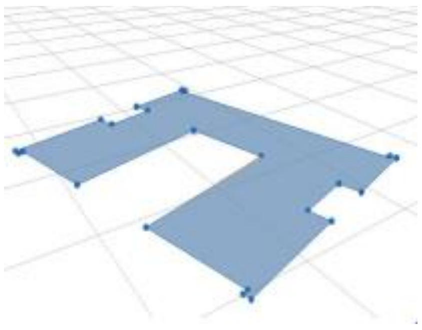
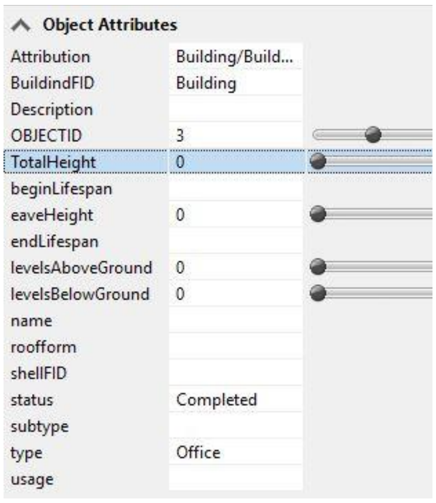


Figure 9: The 2D building footprint



**Object Attributes**

| | |
|---|---|
| Attribution | Building/Build... |
| BuildindFID | Building |
| Description | |
| OBJECTID | 3 |
| TotalHeight | 0 |
| beginLifespan | |
| eaveHeight | 0 |
| endLifespan | |
| levelsAboveGround | 0 |
| levelsBelowGround | 0 |
| name | |
| roofform | |
| shellFID | |
| status | Completed |
| subtype | |
| type | Office |
| usage | |

Figure 10: The attributes of the building footprint

Since CityEngine incorporates the Procedural modeling approach for creating 3D models, a rule is necessary. The rule repetitively applied, will create the facade of the model. At first the rule is used to extrude the building model to the appropriate height value and then the faces of the model are separated, as shown in figure 11. To each face of the building the appropriate attributes are stored in the Object Attributes field according to the fields of the *BuildingShellParts* feature class of the 3DCIM, as presented in figure 12. At this point, also, the roof and the ground surface of the building are diverged from the wall surfaces.

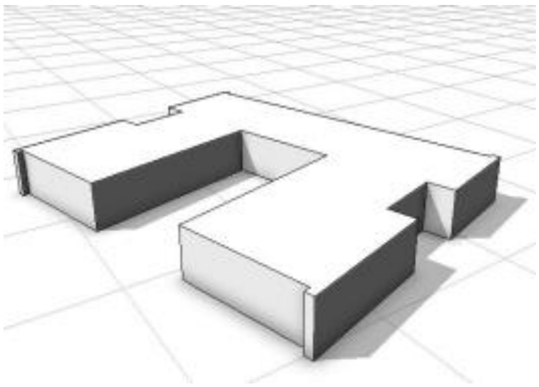
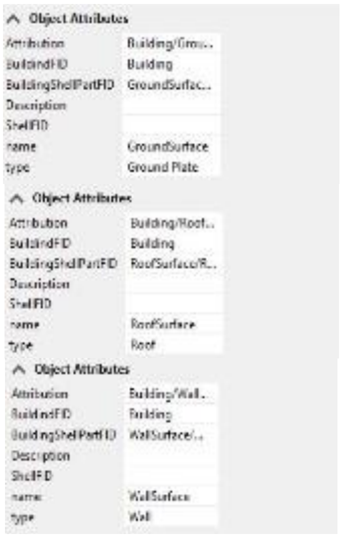


Figure 11: The extruded model of the building with separated surfaces



**Object Attributes**

| | |
|---|---|
| Attribution | Building/Grou... |
| BuildindFID | Building |
| BuildingShellPartFID | GroundSurfac... |
| Description | |
| ShellFID | |
| name | GroundSurface |
| type | Ground Plate |

**Object Attributes**

| | |
|---|---|
| Attribution | Building/Roof... |
| BuildindFID | Building |
| BuildingShellPartFID | RoofSurface/R... |
| Description | |
| ShellFID | |
| name | RoofSurface |
| type | Roof |

**Object Attributes**

| | |
|---|---|
| Attribution | Building/Wall... |
| BuildindFID | Building |
| BuildingShellPartFID | WallSurface/... |
| Description | |
| ShellFID | |
| name | WallSurface |
| type | Wall |

Figure 12: The attributes rendered to the ground, the roof and the wall surfaces

After the extrusion of the model and the assignment of the required thematic and semantic information, the surfaces (the wall surfaces, the roof and the ground level) are exported in shapefile format.
In order to reach the LoD3 of the building model the designing of the openings (windows and door) is required. The formation of the rule in the CityEngine leads to the creation of the model. The final model is presented in figure 13.

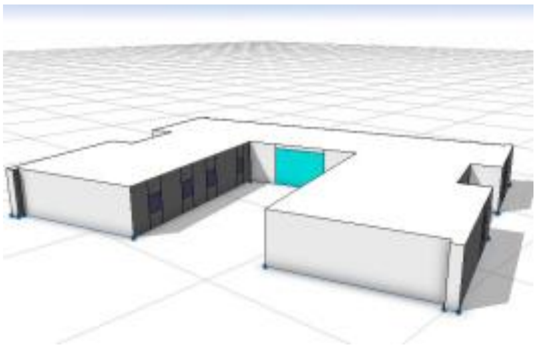
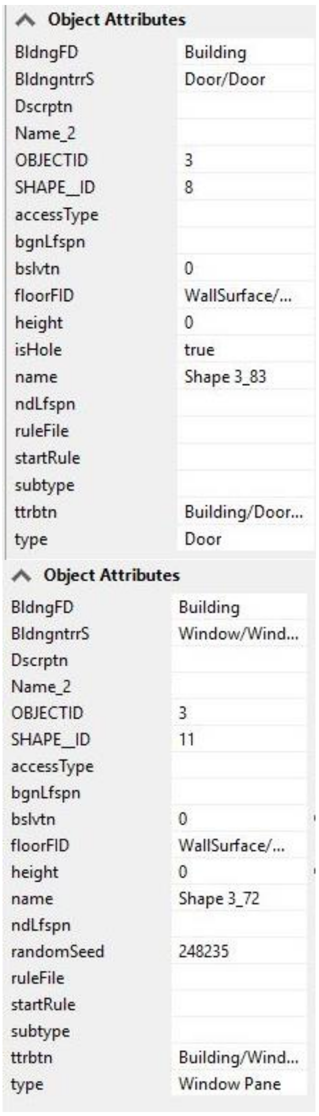


Figure 13: The final building model

At this point the Object attributes of both the window and the door, are enriched, as previously done for the wall surfaces, the roof and the ground floor, as shown in figure 14.



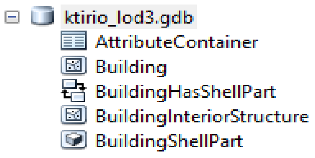
| Object Attributes | |
|---|---|
| BldngFD | Building |
| BldngntrrS | Door/Door |
| Dscrptn | |
| Name_2 | |
| OBJECTID | 3 |
| SHAPE_ID | 8 |
| accessType | |
| bgnLfspn | |
| bslvtn | 0 |
| floorFID | WallSurface/... |
| height | 0 |
| isHole | true |
| name | Shape 3_83 |
| ndLfspn | |
| ruleFile | |
| startRule | |
| subtype | |
| ttrbtn | Building/Door... |
| type | Door |

| Object Attributes | |
|---|---|
| BldngFD | Building |
| BldngntrrS | Window/Wind... |
| Dscrptn | |
| Name_2 | |
| OBJECTID | 3 |
| SHAPE_ID | 11 |
| accessType | |
| bgnLfspn | |
| bslvtn | 0 |
| floorFID | WallSurface/... |
| height | 0 |
| name | Shape 3_72 |
| ndLfspn | |
| randomSeed | 248235 |
| ruleFile | |
| startRule | |
| subtype | |
| ttrbtn | Building/Wind... |
| type | Window Pane |

Figure 14: Left the attributes of the door. Right the attributes of the windows

Next step is the export of the Openings as shapefile format.
It is important to mention that during the export process neither the wall surfaces and the model nor the openings, missed information.

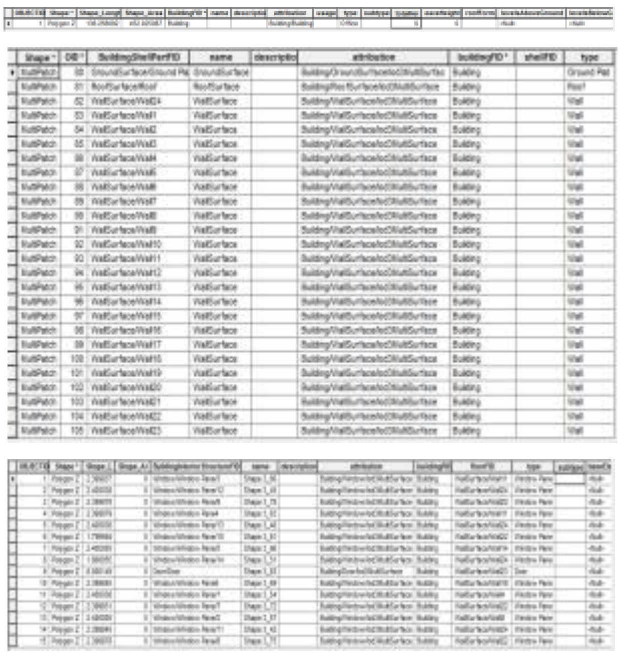
**3.2.1 Semantic Enrichment using 3DCities Project:** After the creation of the model and the export of the data from CityEngine, the 3DCities Project tools were used. Firstly, the geodatabase of the 3DCIM was created in order to semantically enrich the model. In this case the feature classes Building, BuildingShellParts, BuildingInteriorStructures are included in the geodatabase. The geodatabase created for this case study is shown in figure 15.



- ktirio_lod3.gdb
  - AttributeContainer
  - Building
  - BuildingHasShellPart
  - BuildingInteriorStructure
  - BuildingShellPart

Figure 15: The geodatabase of the 3DCIM in this case study

In the *Building* feature class the 2D footprint of the building is in the *BuildingShellPart* each wall surface, the roof and the ground level are separately loaded and the windows and the door are in the *BuildingInteriorStructure*. The index of each feature class is presented below (figure 16).



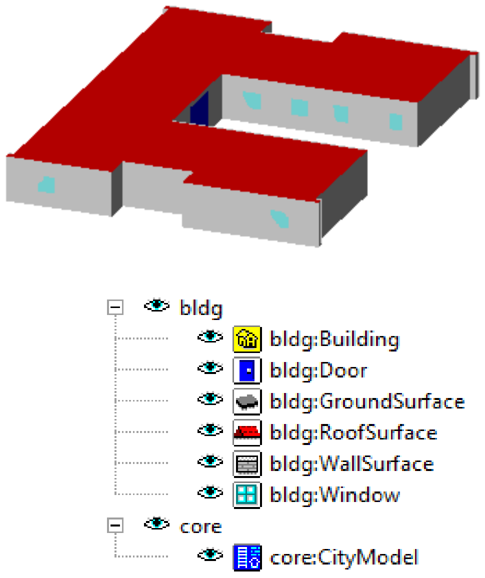
| OBJECTID | Shape* | Shape_Length | Shape_Area | BuildingFID* | name | description | attribution | usage | type | subtype | [illegible] | eaveHeight | roofForm | [illegible] | [illegible] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Polygon Z | [illegible] | [illegible] | Building | | | Building/Building | | Office | | 0 | 0 | | [illegible] | [illegible] |

| Shape* | OID* | BuildingShellPartFID | name | description | attribution | buildingFID* | shellFID | type |
|---|---|---|---|---|---|---|---|---|
| MultiPatch | 80 | GroundSurface/Ground Pl | GroundSurface | | Building/GroundSurface/lod3MultiSurfac | Building | | Ground Pl |
| MultiPatch | 81 | RoofSurface/Roof | RoofSurface | | Building/RoofSurface/lod3MultiSurface | Building | | Roof |
| MultiPatch | 82 | WallSurface/Wall24 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 83 | WallSurface/Wall1 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 84 | WallSurface/Wall2 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 85 | WallSurface/Wall3 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 86 | WallSurface/Wall4 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 87 | WallSurface/Wall5 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 88 | WallSurface/Wall6 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 89 | WallSurface/Wall7 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 90 | WallSurface/Wall8 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 91 | WallSurface/Wall9 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 92 | WallSurface/Wall10 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 93 | WallSurface/Wall11 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 94 | WallSurface/Wall12 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 95 | WallSurface/Wall13 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 96 | WallSurface/Wall14 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 97 | WallSurface/Wall15 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 98 | WallSurface/Wall16 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 99 | WallSurface/Wall17 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 100 | WallSurface/Wall18 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 101 | WallSurface/Wall19 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 102 | WallSurface/Wall20 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 103 | WallSurface/Wall21 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 104 | WallSurface/Wall22 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |
| MultiPatch | 105 | WallSurface/Wall23 | WallSurface | | Building/WallSurface/lod3MultiSurface | Building | | Wall |

| OBJECTID | Shape* | Shape_L | Shape_Ar | BuildingInteriorStructureID | name | description | attribution | buildingFID | floorFID | type | subtype | [illegible] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 2 | Polygon Z | [illegible] | 0 | Window/Window Pane12 | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 3 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 4 | Polygon Z | [illegible] | 0 | Window/Window Pane4 | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 5 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 6 | Polygon Z | [illegible] | 0 | Window/Window Pane10 | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 7 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 8 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 9 | Polygon Z | [illegible] | 0 | Door/Door | Shape 1_[illegible] | | Building/Door/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Door | | <Null> |
| 10 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 11 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 12 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 13 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 14 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |
| 15 | Polygon Z | [illegible] | 0 | Window/Window Pane[illegible] | Shape 1_[illegible] | | Building/Window/lod3MultiSurface | Building | WallSurface/Wall[illegible] | Window Pane | | <Null> |

Figure 16: The index of the Building, the BuildindShellPart and the BuildingInteriorStructure

Final step of the methodology is to export the 3DCIM File Geodatabase to CityGML format. This is an automatic procedure since the ETL tools of the 3DCitites Project incorporate the FME Workbench (Feature Manipulation Engine, Safe Software) and its functions. The correct match of the data is very significant at this step, in order to ensure a lossless transformation. The final CityGML model is being inspected in the FZKViewer environment. The final model is presented below (figure 17).



- bldg
  - bldg:Building
  - bldg:Door
  - bldg:GroundSurface
  - bldg:RoofSurface
  - bldg:WallSurface
  - bldg:Window
- core
  - core:CityModel

Figure 17: Visualization of the model in the FZK Viewer

As it is concluded after evaluating the final model, the only deviation from the CityGML standard is that the wall surfaces are represented as Multisurfaces in LoD2 instead of LoD3. In figure 18 the structure of the 3DCIM CityGML model is shown.

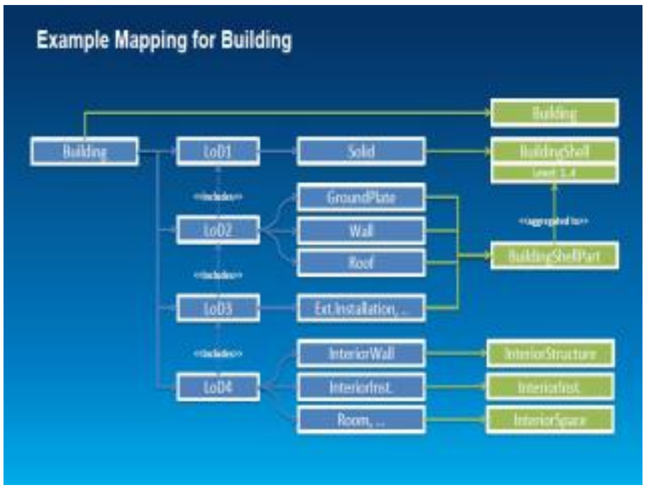


Figure 18: The structure of the 3DCIM (Arcgis.com, 2014)

## 4. EVALUATION OF THE METHODOLOGIES

### 4.1 Georeference

1st Methodology: In Trimble SketchUp there are limitations regarding the modeling in regard to data coordinates with specific reference system. However, the georeference of a model is partially solved utilizing the Google Earth Geolocation plugin. Additionally, the FME provides the ability to re-project the reference system of the SketchUp file to the reference system that fits the purposes of the user.

2nd Methodology: The import of a georeferenced building footprint in the CityEngine successfully addresses the georeference issue. Moreover, CityEngine supports the reference system used in this case study and as a result the model has the correct georeference. In the 3DCities Project, the specific reference system is also supported, and consequently the georeference is preserved.

### 4.2 User friendliness

1st Methodology: Trimble SketchUp is a user-friendly modeling tool. However, if the aim of the project is to create a CityGML Model, the user has to be familiar with the basic structure and principles of the CityGML Standard, especially for a LoD 4 model, since some features such as the boundary surfaces become more complicated. The FME Desktop Manager requires several trials since it provides numerous conversion options are. Moreover, the user must be deeply aware of the CityGML structure, in order to generate a semantically, geometrically and topologically concrete CityGML Model.

2nd Methodology: The modeling process in CityEngine is standard since it simulates a simple form of programming language. The user can intervene in the ruling in order to create a model. However, it is important to be familiar with the principles of CityGML, in case this format is the desirable outcome of the transformation. The 3DCIM toolbox is incorporated in the ArcGIS environment, which is a popular GIS software. Nevertheless, the comprehension of the information model's structure is advisable.

### 4.3 Ability of Expansion

1st Methodology: The transition from a LoD1 to a LoD3 Model has a low level of difficulty. In order though to create a LoD 4 Model, a start-over is required if the interior is added complicating the new model, both geometrically and topologically. However, Trimble SketchUp seems to be a solid tool for various modeling projects in terms of realism and in presenting different aspects of the city. As soon as the model is enriched in the Trimble SketchUp software, the FME is able to convert each and every additional information to the gml format. However, the user must edit the workbench based on the CityGML Model that wants to generate, a procedure that requires a certain level of knowledge of the FME conversion tools.

2nd Methodology: Creating a detailed building model in CityEngine has a low level of difficulty. However, an even more detailed model in Lod4, with information about the interior structures, demands appropriate data and greater effort when creating the procedural rule. The 3DCities Project offers a standardized information model in order to serve a wide range of applications. Although FME tools are incorporated, an adequate knowledge of FME and its functions is required in order to customize or further expand its information model.

### 4.4 Cost

1st Methodology: Trimble SketchUp is a software commercially available to the public. The FME Desktop Manager is commercially available to the public as well, however it offers multiple trial periods and special licenses. In terms of cost of time, the modeling process requires some experience in order to become familiar with the tools of Trimble SketchUp. A complete Building in LoD 3 can be modelled within a day. However, the FME Workbench offers unlimited capabilities of converting the model. As a result, in case there is not a previous workbench to work upon a generous amount of time is required in order to figure out the optimal way for a CityGML conversion.

2nd Methodology: The CityEngine and the ArcGIS are ESRI's software and are both commercially available, also offering trial periods and academic licenses. The time invested to complete the particular project is considered rather short. Modeling in the CityEngine is fast since it is characterized by repetitiveness, and so it does not demand much time. Once you grasp how the rules are formed the modeling process is quick. In the ArcGIS environment, the data are loaded to the relevant feature classes in the 3DCitiesProject quickly and since the transformation to the CityGML format is fully automatic, the final model is generated in a short time.

### 4.5 Automatism of the Procedure

1st Methodology: The modeling may be characterized as a manual process, although there are various tools that enhance the automatism of specific modeling steps, such as the floors or the installations. Inside the FME, the conversion procedure can be described as a semi-automatic one. As soon as the workbench is created, the user has to interfere in specific parts of the procedure to customize the transformers based on the purposes of the project.

2nd Methodology: In CityEngine, the syntax of the rule and the assignment of the necessary attributes to create the model can be characterized as manual. Nevertheless, there are functions that are automatic and fulfill the modeling process. As soon as, the data is loaded in the 3DCitiesProject, the transformation to CityGML format is characterized as an automatic process.

### 4.6 Visualization

1st Methodology: In terms of visualization, Trimble SketchUp provides great modeling tools that enhance the realism of the model, such as shadows, fog etc. The online warehouse is a great source of various components and commands such as the ‘’section plane’’ that allows for the observation of the model’s interior areas. The FME provides the ability to visualize the model at every stage of the conversion procedure. This is a particularly useful feature, since it provides a great help for the detection of errors and malfunctions. Moreover, the FME Desktop Manager is able to visualize the generated CityGML model. The textures and the semantic information of the model are also visible through FME Desktop Manager.

2nd Methodology: CityEngine offers the ability to repetitively produce building models with high visual quality. It is a software designed for 3D modeling and the assignment of the facade’s details results to the creation of a truly realistic model. The information model of the 3D Cities template supports appearance in order to produce a realistic model. As a result all the information, attributes, geometry and appearance of the procedurally generated model is preserved and may be successfully loaded in the 3DCIM file geodatabase.

### 4.7 Accuracy of the Model

1st Methodology: Trimble SketchUp is a software mainly designed for architectural use, providing a great level of accuracy during the modeling. The length and width of the objects are calculated with an accuracy of at least two decimals, while the three axis that exist in the model support the successful orientation of the model’s surfaces. However, the lack of a highly accurate geolocation can be an issue for specific study field areas. The FME software provides accuracy in the semantic and topological aspects of the model. The user is able to add specific attributes to every CityGML object, as well as determine their geometry, feature role and level of detail.

2nd Methodology: Modeling in CityEngine is accurate, since 3D modeling is the main purpose for its development. The rules for the creation of the model in CityEngine are based on precise data, and so the final model is geometrically correct. The 3DCities Project is used for the semantic enrichment of the procedural model and the tools that it incorporates are fully capable of preserving the semantic and topological aspects of the information model and even enrich them.

### 4.8 Data Structure

1st Methodology: The generation of a model, can be achieved through various modeling methods depending on the project. When the model is generated, in order to achieve a successful conversion to a gml file based on the CityGml Standard, the data must be structured in a specific form. The more parts the model is consisted of, the higher the level of difficulty to structure the data, since it is a completely manual process. The creation of the workbench in FME, requires a certain structure of the transformers and the parameters being used. As soon as the workbench is completed, the data in terms of identification match the data in the Trimble SketchUp, a procedure that is manually executed by the user.

2nd Methodology: CityEngine provided different options in developing a 3D model. The higher the complexity and the information of the model, the more difficult is the modeling and stronger the need for better structure. Moreover, the data structure needs to be compatible with the structure of the Information Model of the 3DCities Project. In order to achieve interoperability, the compliance to the structure of the 3DCIM and consequently to the CityGML Standard is vital.

## 5. CONCLUSIONS AND FUTURE RESEARCH

Based on the above evaluation of the two methodologies, the following conclusions can be obtained: the model generated via the first methodology is visually concrete and realistic. The limitless available components render Trimble SketchUp as a solid tool for complicated visual projects that include more information than the surfaces of the building. Moreover, the simple modeling procedure produces precise results in a short time. The expansion of the model, by using various different CityGML modules is also available. The FME, is a transformation tool with a wide range of attributes and semantic information that can be implemented to the model. Hence, despite Trimble SketchUp is limited in terms of processing semantical information, this disadvantage is adequately covered by the FME Workbench capabilities. In case that the model is consisted of significantly more buildings and other CityGML modules, the transformation in FME requires respectively more effort and time. Finally, the limited abilities for geolocation do not recommend the use of the generated model for land use and cadastral purposes. To sum up, the first methodology presents a procedure that generates a CityGML model that supports every function of the CityGML Standard, in terms of the Building Module. The FME allows the user, to edit the parameters and the information that are being transformed and is fully compatible with the semantic aspects of the CityGML Standard. The generated model can be successfully used in the wider field of the 3D Smart Cities’ simulation, such as architecture, energy, environment and urban planning.
The CityEngine mainly focusses on the 3D modelling of the city fabric, and lacks semantic information. The export options offered by this software are not able to preserve the semantic information incorporated to the model. The template solution for ArcGIS, the 3DCities Project, may be exploited for the semantic enrichment of the model. Potential interoperability issues may be overcome if the initial data are structured based on the specifications defined by the 3DCIM and consequently the CityGML. On the other hand, the creation of the model in CityEngine environment was neither time-consuming nor cumbersome, since no particular programming skills are needed and the final model is efficient in terms of appearance and realistic visualization. As far as the 3D Cities Project is concerned, it is a collection of geoprocessing tools, base layers, sample projects and information model, very useful in a wide range of applications. The information model, and particularly the geodatabase, serve many different purposes. The 3DCIM file geodatabase contains defined feature classes with specific attributes and relations between them. Consequently, it is important the structure of the geodatabase to be thoroughly investigated in order to be populated with the appropriate data, according to the specifications defined. Additionally, incorporating the FME software, which has strong capabilities to translate and transform data, it is possible to automatically transform the data stored in the 3DCIM file geodatabase to a CityGML compatible format. The semantic enrichment of the initial building model and the transformation to CityGML were successfully completed, harmonizing with the restrictions and the

structure of the 3DCIM and the CityGML, thus achieving interoperability and no losses of data. The only problem pointed out from this procedure relates to the rendering of the surfaces in LoD2 instead of LoD3.
With regard to future research, the Building module of CityGML, has been thoroughly investigated and successfully generated in various LoDs. An interesting field for further research is exploring the other CityGML Modules, such as the Transportation Complex, the Bridge Module, the Land Use and the City Furniture in an effort of generating CityGML model with multiple modules that uniquely interact with each other. Within this context, a comparison with the BIM Modeling methodology and the adequacy of the two methodologies in terms of semantics would also be worth investigating. Moreover, another challenge is to convert the semi-automatic procedure of the FME Transformation into a fully-automatic procedure where the user will not have to manually intervene in the workbench. Finally, since the assignment of the layer in Trimble SketchUp can be described as a challenging task, the creation of an external plug-in that semi-automatically assigns the layers to the corresponding objects might be proven as a solid tool to enhance the modeling procedure. Concerning the 3DCities Project, addressing the malfunction of the geometries in the appropriate level of detail needs to be also investigated. Another advantage of using the 3DCities Project is its potential application to land use and property related issues, by storing relevant information to the geodatabase incorporated, leading to the development of an organized way of maintaining records.

## REFERENCES

Benner, J., Geiger, A., Gröger, G., Häfele, K. H., & Löwner, M. O. 2013. Enhanced LoD concepts for virtual 3D city models. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2, W1.

Breunig, M., & Zlatanova, S., 2011. 3D geo-database research: Retrospective and future directions. *Computers & Geosciences*, 37(7), pp.791-803.

Brown, G., Nagel, C., Zlatanova, S., & Kolbe, T. H., 2013. Modelling 3D topographic space against indoor navigation requirements. In *Progress and New Trends in 3D Geoinformation Sciences* pp. 1-22. Springer Berlin Heidelberg.

deLaat, R., & van Berlo, L., 2011. Integration of BIM and GIS: The development of the CityGML GeoBIM extension. In *Advances in 3D Geo-Information Sciences* pp. 211-225. Springer Berlin Heidelberg

Diakité, A.A, Damiand, G. and Gesquière, G., 2014. Automatic Semantic Labelling of 3D Buildings Based on Geometric and Topological Information. In *Proceedings of 9th International 3D GeoInfo Conference* (3D GeoInfo), Nov. 2014, Dubai. United Arab Emirates. Karlsruhe Institute of Technology, 3DGeoInfo conference proceedings series.

Dimopoulou, E., & Elia, E., 2012. Legal aspects of 3D property rights, restrictions and responsibilities in Greece and Cyprus. In *Proceedings of the 3rd International Workshop on 3D Cadastres*, Developments and Practices pp. 25-26.

Dimopoulou, E., Tsiliakou, E., Kosti, V., Floros, G., and Labropoulos, T., 2014. Investigating Integration Possibilities between 3D Modeling Techniques. In *Proceedings of 9th International 3D GeoInfo Conference*, Nov. 2014, Dubai. United Arab Emirates.

Döllner, J., Baumann, K., Buchholz, H., 2006. Virtual 3D City Models as Foundation of Com- plex Urban Information Spaces. In: Schrenk, M. (Ed.), Proc. *1st international conference on Urban Planning and Spatial Development in the Information Society* (REAL CORP). CORP - Competence Center of Urban and Regional, Planning, pp. 107-112.

Donkers, S., 2013. Automatic Generation of CityGML LoD3 Building Models from IFC Models. MSc thesis, Delft University of Technology, Department of GIS Technology, OTB Research Institute for the Built Environment

El-Mekawy, M., 2010. Integrating BIM and GIS for 3D City Modelling: The Case of IFC and CityGML. Licentiate Thesis, KTH, Stockholm, Sweden

El-Mekawy, M., Östman, A., & Shahzad, K., 2011. Towards interoperating cityGML and IFC building models: a unified model based approach. In *Advances in 3D Geo-Information Sciences* pp. 73-93. Springer Berlin Heidelberg.

Floros, G., Tsiliakou, E., Kitsakis, D., Pispidikis, I., Dimopoulou, E., 2015. Investigating Semantic Functionality of 3d Geometry for Land Administration. In *Proceedings of 10th Joint International 3D GeoInfo Conference*, Oct. 2015, Kuala Lumpur, Malaysia.

Gröger, G., Kolbe, T., & Czerwinski, A., 2007. Candidate OpenGIS® CityGML Implementation Specification (City Geography Markup Language). Open Geospatial Consortium Inc, OGC.

Gröger, G., Plümer, L., 2012. CityGML – Interoperable semantic 3D city models. *ISPRS Journal of Photogrammetry and Remote Sensing* 71 (2012) 12–33

Isikdag, U., Zlatanova. S., 2009. Towards Defining a Framework for Automatic Generation of Buildings in CityGML using Building Information Models. In: Lee J. and Zlatanova, S. (Eds.) *3D Geoinformation Sciences*, pp.79-96, Springer Berlin Heidelberg.

Kolbe, T.H., 2009. Representing and Exchanging 3D City Models with CityGML. Proceedings of the 3rd International Workshop on 3D Geo-Information, Seoul, Korea. *Lectures in Geoinformation and Cartography*, Springer Verlag.

Kolbe, T.H., 2008. Representing and Exchanging 3D City Models with CityGML. In: Lee, J., Zlatanova, S. (Eds.), *3D Geo-information Sciences. Springer*, Berlin, pp.15–31.

Muller, P., Wonka, P., Haegler, S., Ulmer, A., & Gool, L.V., 2006. Procedural Modeling of Buildings. *ACM Transactions on Graphics*, 25(3), pp. 614-23.

Sadidi, J., Talebzadeh, M., Rezaian, H., Zeaiean, P., 2015. Designing 3D Semantic Model in LOD4 to Simulate Building Utility Network, *Indian Journal of Science and Technology*, Vol. 8(16), July 2015

Stadler, A., Kolbe, T. H. 2007. Spatio-semantic coherence in the integration of 3D city models. In *Proceedings of the 5th International Symposium on Spatial Data Quality*, Enschede.

Zlatanova, S., Stoter, J., Isikdag, U., 2012. Standards for exchange and storage of 3D information: Challenges and opportunities for emergency response. In *Proceedings of the Fourth International Conference on Cartography and GIS*, Albena, Bulgaria, pp. 17-28.

Müller, P., Wonka, P., Haegler, S., Ulmer, A., and Van Gool, L., 2006. Procedural modeling of buildings. ACM Trans. Graph., vol. 25, no. 3, p. 614.

Zhu, Q., Zhao, J., Du, Z., Zhang, Y., Xu, W., Xie, X., & Wang, T., 2011. Towards semantic 3D city modeling and visual explorations. In *Advances in 3D Geo-Information Sciences* (pp. 275-294). Springer Berlin Heidelberg.

ESRI, (2014). Getting started with 3D Cities. https://github.com/Esri/3d-cities-template

ESRI, (2014). 3D Cities Information Model Overview. https://github.com/Esri/3d-cities-template