



L1.3 3D Data Standards

Lecture Notes

Author(s)/Organisation(s):

Ariana Kubart, Ocellus Information Systems AB, Sweden

License



<https://creativecommons.org/licenses/by/4.0/>

Version

Version 2.0

Date: April 2025

Summary

The last lecture of this block describes the 3D city models on data level. It introduces the student into the CityGML conceptual model and describes its modules and how they can be used in diverse aspects of city modelling. It goes more deeply into certain representation, e.g. geometric, topologic or time, because knowledge of these is important for understanding the process of BIM-GIS integration. Further, the lecture provides information on different CityGML encodings as well as other 3D formats.

Learning outcomes

At the end of this lecture, the learner is expected to be able to:

Name several ways how the 3D data can be stored, with focus CityGML encodings

Summarize the main parts of CityGML Conceptual model and how they can be used

Understand the aspects of CityGML that are important for conversion to and from BIM



Expected competences when entering the lecture

Intermediate GIS knowledge

L1.1 Concepts of 3D Modelling

L1.2 Semantic City Models

Expected workload

26 slides with learning content, approximately 5 hours

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Contents

Semantic 3D City Models.....	4
CityGLM Standard.....	5
CityGLM Conceptual Model	6
Class Modules in the Conceptual Model.....	7
Specific-Aspect modules in CM	8
Semantic Information in the 3D models	9
Aggregation schema.....	10
Coherent Semantic-Geometric Modelling	11
Levels of Detail, LODs, in CityGML.....	12
Levels of Detail, LoD II	13
Previous versions of CityGML, 1.0 and 2.0	14
Geometry representation.....	15
Topology Representation in CityGML 3.0.....	16
Physical and logical spaces	17
Exteriors, Interiors and Hierarchies.....	18
Coordinates and elevation	19
4D – time dimension	20
Application Domain Extensions, ADEs in CityGML.....	21
Data Quality.....	22
Encodings of CityGML.....	23
XML encoding.....	24
CityJSON Encoding I.....	25
CityJSON Encoding II.....	26
3D City Database Encoding.....	27
Other 3D formats	28
LandInfra	29
Inspire 3D buildings	30
References.....	31

3D Data Standards

Semantic 3D City Models

- Provide geographic, topographic and semantic information about objects
- Interactions among objects
- Hierarchical decomposition into smaller parts (e.g. building-wall-window)
- Can be complex and cover large areas



A lot of semantic information can be added into a 3D model
3D model of Nancy, France, Google Earth

4

Semantic 3D City Models

The semantic city models provide geographic, topographic and semantic information about objects in urban areas (see Lecture 1.2). The semantic objects can be spatially represented by multiple geometries and in different LODs. Further, the objects can be decomposed into smaller parts in several levels and this decomposition is hierarchical (e.g. building-wall-window).

The ready-to-use 3D models have to be stored somewhere and accessible to the final users. At the same time, the semantic models can be very complex and cover large areas.

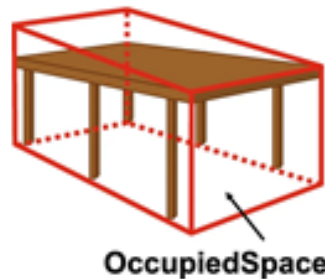
It means that a 3D geodatabase for a 3D model must be able to cope both with semantic and geometric hierarchy, to manage the large dataset and allow access, updates and queries of the data.

3D Data Standards

CityGML Standard

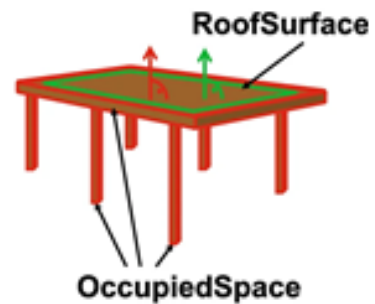
- CityGML is open data model by the Open Geospatial Consortium (OGC)
- Aimed to represent semantic 3D models
- Current version is 3.0, approved in 2021

Carport in LOD1



OccupiedSpace

Carport in LOD2/3



OccupiedSpace

Representation of a carport as OccupiedSpace in different LODs. From: [CityGML 3.0: New Functions Open Up New Applications](#)

5

CityGLM Standard

How is all that coded in the computer?

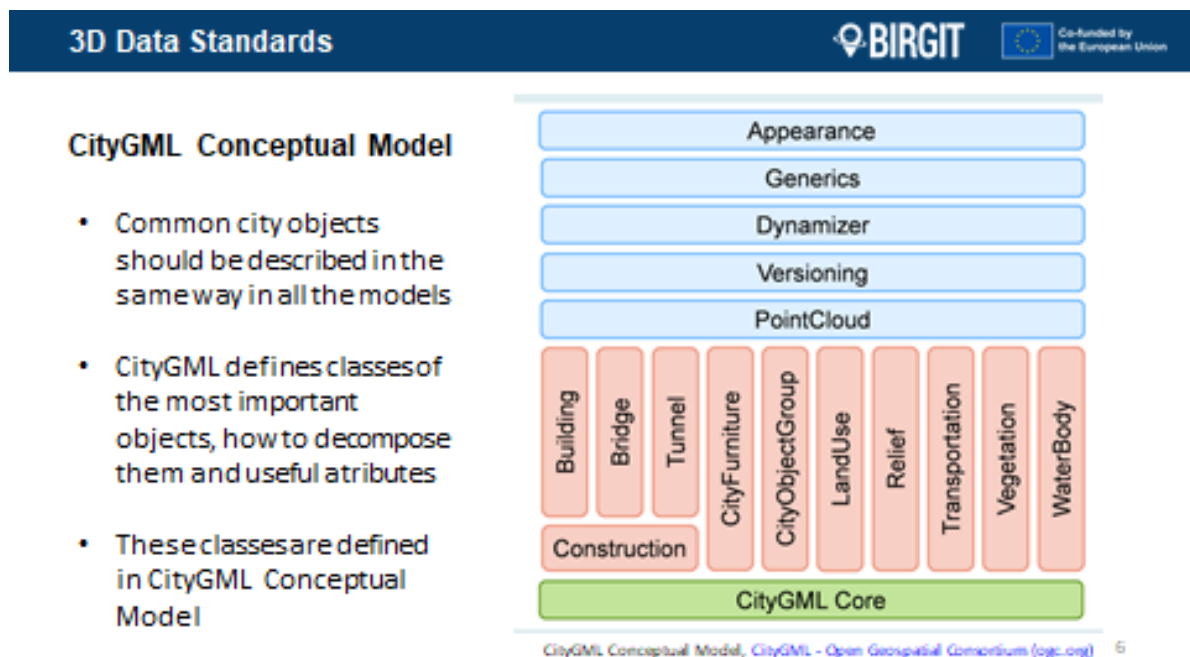
There is an open data model standardised by the Open Geospatial Consortium (OGC), aimed to represent the semantic 3D models. The standard is called CityGML and the current version is 3.0, approved in 2021.

OGC own definition sounds: “CityGML is a common semantic information model for the representation of 3D urban objects that can be shared over different applications”.

How does it work? We are going to look at it closer in this lecture.

[CityGML - Open Geospatial Consortium \(ogc.org\)](https://ogc.org/)

[CityGML 3.0: New Functions Open Up New Applications | PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science \(springer.com\)](#)



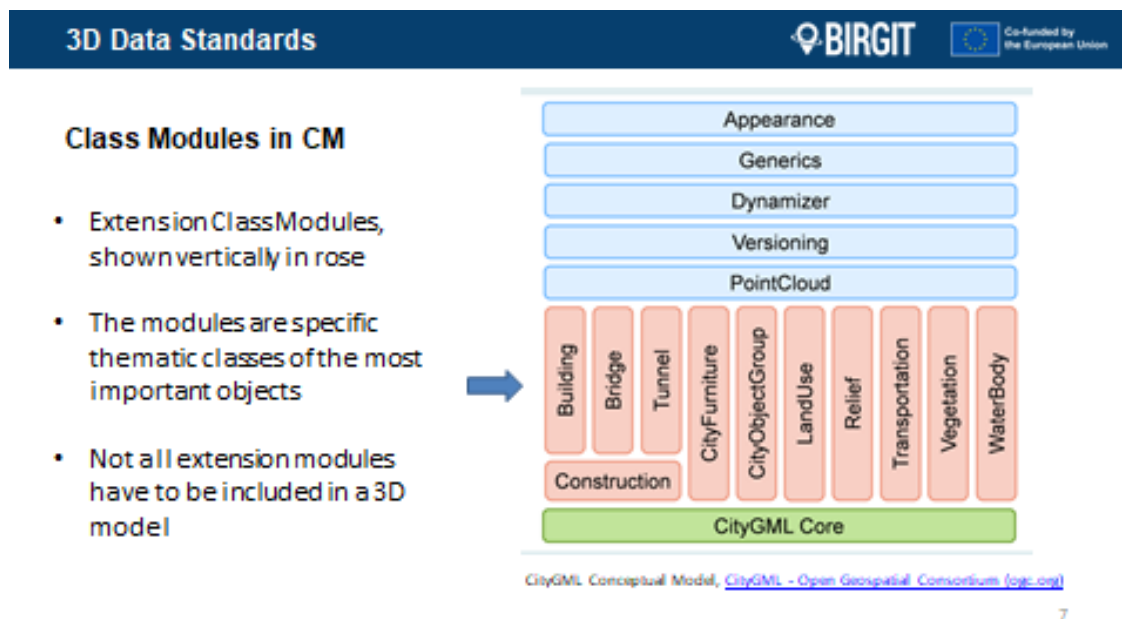
CityGLM Conceptual Model

Many objects, like buildings, streets or bridges, are common in all cities. Therefore, they should be described in the same way in all of the models. Otherwise “buildings” might for example be called “houses” in another city and there would be no interoperability.

Therefore, CityGML defines its own classes of the most important objects, how to decompose them and often some useful attributes to be included. These classes are defined in CityGML Conceptual Model (CM).

The Conceptual Model is visualised by the figure on the slide.

The basic part of the Conceptual Model is Core module, shown down in green. The core module comprises the basic concepts and components of a virtual city. Thus, it must always be implemented in any 3D city model.



Class Modules in the Conceptual Model

Besides the Core module, there are the extension class modules, shown vertically in rose colour.

Each extension module covers a specific thematic class for the most important objects within virtual 3D city. These include constructions, relief, e.g. in form of DTM (digital terrain model), water bodies and so on, as shown on the figure.

The three modules *Building*, *Bridge*, and *Tunnel* model civil structures and share common concepts. That is why they are grouped within the *Construction* module.

The Transportation module defines classes for the representation of the traffic infrastructure. It allows using CityGML for traffic and driving simulations, driving assistance systems, autonomous driving, as well as with road and railway facility management systems.

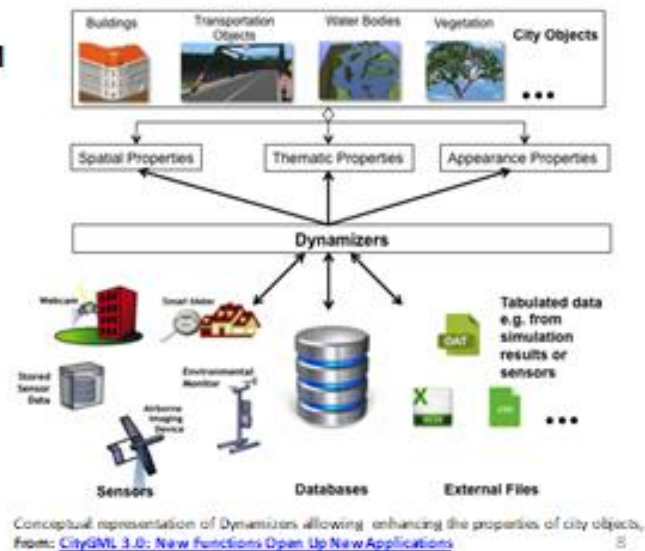
A city model does not need to support all the classes (=extension modules). It may employ only a subset according to a model's specific needs.

For example, if an application works exclusively with building data, it is enough with the *Core*, *Construction*, and *Building* modules only.

3D Data Standards

Specific Aspect Modules in CM

- The five blue-coloured vertical modules in the CM
- Add specific modelling aspects, e.g. textures, colours, time, model versions...
- Can be used together with all extension modules



Specific-Aspect modules in CM

The five blue coloured modules add specific modelling aspects. These that can be used together with all extension modules.

The *Appearance* module represents e.g. textures and colours of city objects.

The *PointCloud* module provides concepts to represented object geometry by 3D point clouds.

The *Generics* module defines generic objects, attributes, and relationships. The generic objects represent 3D objects, which are not covered by the explicitly modelled classes. Similarly, generic attributes and relationships are those additional ones.

Versioning adds concepts for the representation of concurrent versions of the objects.

The *Dynamizer* module allows to represent object properties by time series data and to link them with sensors or external files.

Semantic Information in the 3D models

- All objects belong to a class or can be defined as “generic” objects
- The object can be represented by semantics (= non-spatial properties), 3D geometry, 3D topology, appearances and changes over time
- Unique and mandatory *featureID* for each object



TU Delft example of five types of roofs: flat, gabled, hipped, pyramidal, and shed.

[Github - tudelft3d/3dbook: Book for the course GEO1004: 3D modelling of the built environment](#)

9

Semantic Information in the 3D models

In CityGML, all city-objects are modelled by the classes, as we have seen above. These objects thus “know” what they are and where they are.

Combining the diverse CityGML-modules, each object can be represented by its semantics (= non-spatial properties), 3D geometry, 3D topology, appearances (e.g. surface characteristics), and its changes over time.

Each object has unique and mandatory *featureID*. It can also have an optional *identifier*. The *identifier* is identical for all versions of the same object, like a window or a table. That facilitates thematic queries, analysis tasks, or spatial data mining, far beyond visualization.

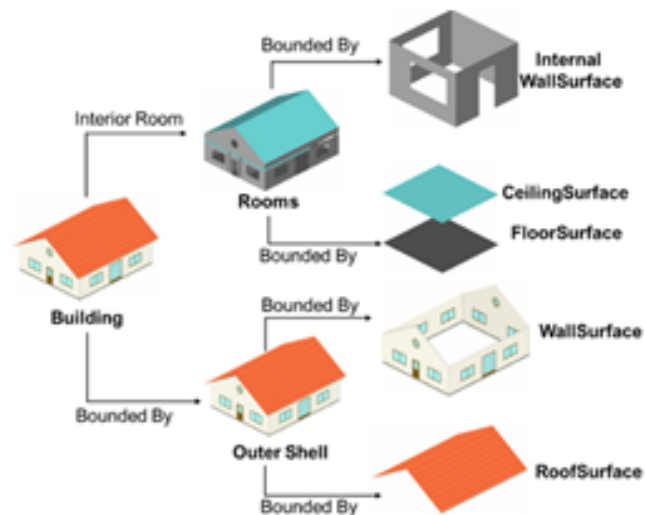
The attributes for classifying objects are often restricted to a set of discrete values. For example, there is just limited number of roof types. They are specified as external code-lists. However, the user can redefine these external code-lists.

Further, objects which are not explicitly covered by the conceptual-model’s classes can be represented as the generic objects and generic attributes.

3D Data Standards

Aggregation Schema

- There can be hierarchical interrelations among city objects
- Aggregation hierarchy – whole building and its decomposition to parts (see figure)
- Useful for queries, simulations and analyses



Overview of a building represented in CityGML format. From Malhotra et al (2021): Open-Source Tool for Transforming CityGML Levels of Detail¹⁰

Aggregation schema

City models need to reflect the complexity of city objects and their interrelations.

The objects can be related to each other by different types of relations in CityGML, and these relations are usually hierarchical.

Complex objects like buildings or transportation objects typically consist of parts. These parts are individual features of their own, and can even be further decomposed. We have already mentioned in L1.2 example of hierarchy building – building part – roof surface, wall surface, and building installation – door, window as subcategory of wall surface. This is referred as aggregation hierarchies of components, which, in other words, define part-whole relations between features.

It is not only an object that can be decomposed. Even whole urban-area can be semantically decomposed into e.g. land use, plant cover, buildings.

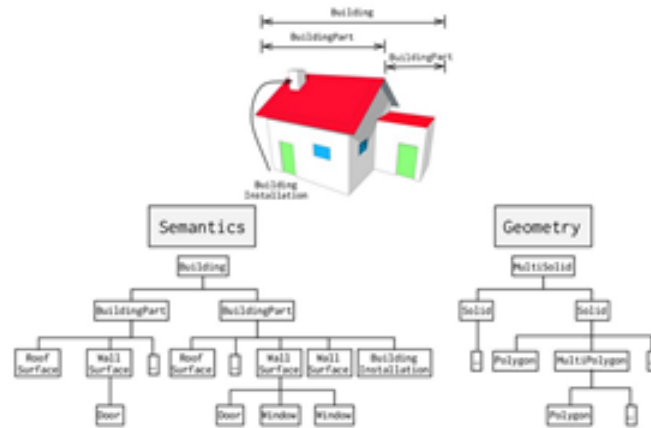
This hierarchical structure is necessary for queries, simulations and analyses.

3D Data Standards

Coherent Semantic-Geometric Modelling

- Building and its parts = semantic aggregation hierarchy
- There is also geometric hierarchy – location, shape, extent...
- Crucial that semantic and geometry of the corresponding objects are linked together

Spatio-semantic coherence



From Ledoux (2018): CityGML and its two encodings, CityGML and CityJSON

11

Coherent Semantic-Geometric Modelling

Hence, the city-objects are represented by features, such as buildings or windows. They can also include attributes, relations and hierarchies between the features. All that is the semantic information.

However, the objects are also assigned their location, shape, and extent, i.e. by their geometry.

So, the model consists of two hierarchies: the semantic one and the geometrical one, as shown on the figure. The corresponding objects (and their parts) are linked by relationships. It is crucial that these links are matched and that they fit together.

It is possible to navigate in both hierarchies and between them arbitrarily, for answering thematic and geometrical queries or to perform analyses.



Levels of Detail, LODs, in CityGML

Each object can have different spatial representations at the same time. It is provided by four predefined Levels of Detail (LOD 0-3).

Briefly, level of detail is the quantity of information in the model that portrays the real world.

The LODs are:

LOD0 – Highly generalized model;

LOD1 – Block model / extrusion objects;

LOD2 – Realistic, but still generalized model;

LOD3 – Highly detailed model.

For instance, a building can also be abstracted by a footprint or roof print (LOD0), by a 3D solid with flat roof (LOD1) up to detailed visualisation in LOD3.

3D Data Standards

Level of Detail, LoD II

- LoDs are applicable to both interior and exterior
- Individual building or whole neighbourhoods
- possible to combine different LODs in same model

Visual representation of several housing-blocks in Vienna (a) underlying image from Google Maps, (b) representation of open LoD2 CityGML model, (c) LoD1 model transformed from LoD2, (d) LoD0 CityGML models. from Melchior et al (2021)



(a)



(b)



(c)



(d)

13

Levels of Detail, LOD II

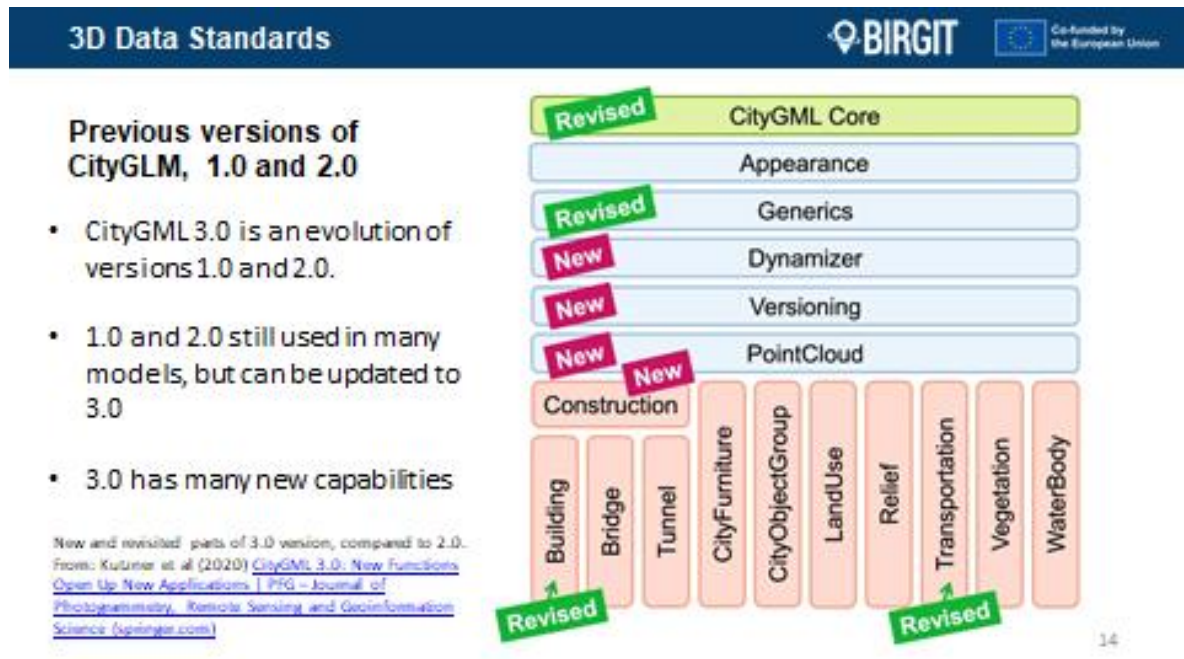
LOD1 models are easy to reconstruct: the footprint of a building can be extruded to its height. This height can be e.g. the average of all the LIDAR points inside the footprint.

LOD2 includes the generalised roof shape. As such, LoD2 models are useful for estimations of photovoltaic potential. They are usually obtained with photogrammetric techniques, and derived automatically.

LOD3 is a detailed model containing openings (windows and doors), chimneys, and other façade details. Models at LOD3 are usually obtained with a conversion from BIM models or from terrestrial laser scanning. The presence of windows and other details makes them useful in applications such as energy simulations.

These LODs are applicable to both interior and exterior and they can be combined. Building floor plans are LOD0 representations of building interiors.

It is possible to combine different LODs in same 3D model. For example, outside shell of a building can be represented in LOD2 and the indoor elements like rooms or stairs in LOD1. It is even possible to model the outside shell of a building in LOD1, while representing the interior structure in LOD2 or LoD3.



Previous versions of CityGML, 1.0 and 2.0

CityGML 3.0 is an evolution of the previous versions 1.0 and 2.0.

It has many new capabilities. These include much better integration with BIM, the ability to represent indoor spaces in different Levels of Detail (LOD), support for dynamic sensor data, and the capability to extend the information model into Application Domain Extensions, ADEs (as we will discuss later).

As there are many models created using CityGML 1.0 and 2.0, these specifications will not be deprecated. However, these existing CityGML data can be upgraded to CityGML 3.0. in order to reach the new possibilities.

3D Data Standards

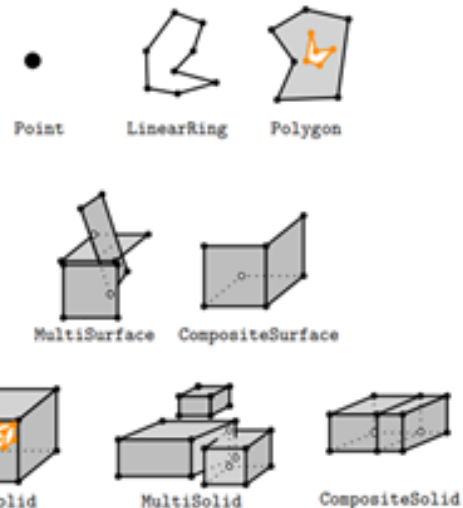
Geometry Representation

Spatial properties of all CityGML objects are represented by geometry classes defined in ISO 19107

These include:

- primitive geometries- points, curves, surfaces and solids
- different kinds of aggregated geometries

All geometries saved in Core module



Some of the CityGML primitives, including aggregates and composites.
from Othori et al (2020-2022) [Release: tudelft3d/3dbook \[github.com\]](https://github.com/tudelft3d/3dbook) 15

Geometry representation

Spatial properties of all CityGML objects are represented using the geometry classes defined in ISO 19107. These include primitive geometries, that is to say single points, curves, surfaces, and solids.

Besides, there are different kinds of aggregated geometries: spatial aggregates (*MultiPoint*, *MultiCurve*, *MultiSurface*, *MultiSolid*) and composites (*CompositeCurve*, *CompositeSurface*, *CompositeSolid*).

Objects are defined by attributes that define their sub-elements. These sub-elements are then combined to form the complete object. It sounds complicated, so let's illustrate it by a figure:



[GitHub - tudelft3d/3dbook: Book for the course GEO1004: 3D modelling of the built environment](https://github.com/tudelft3d/3dbook)

Geometry of this building is composed of the house geometry (*CompositeSolid*) and the garage/annex on the right (*Solid*). The house's geometry is further decomposed into the roof geometry (*Solid*) and the geometry of the house body (*Solid*).

In CityGML 3.0 all geometric representations are defined in the Core module only.

Topology Representation



CityGML schema showing relationship from Window towards the Room to the Building.
From: Salheb (2019) Automatic Conversion of CityGML to IFC

- Topology follows full decomposition, similar to the geometry
- Relationships between element well defined in CityGLM 3.0
- Spaces – real world objects
- Space boundaries – delimit and connect the Spaces (e.g. wall surface, road surface....)

16

Topology Representation in CityGML 3.0

The topology model of CityGML3.0 follows the full decomposition of n-dimensional topologies down to the level of nodes, similarly to the geometric decomposition.

The schema of the relationship between elements is well defined. For instance, the relation between a window and a room is defined in the schema shown in the figure.

CityGML 3.0 also maps all city objects onto two semantic concepts: these two are ‘space’ and ‘space boundary’.

A *Space* is a real-world object with volumetric extent, such as buildings, water bodies, trees, rooms, and traffic spaces.

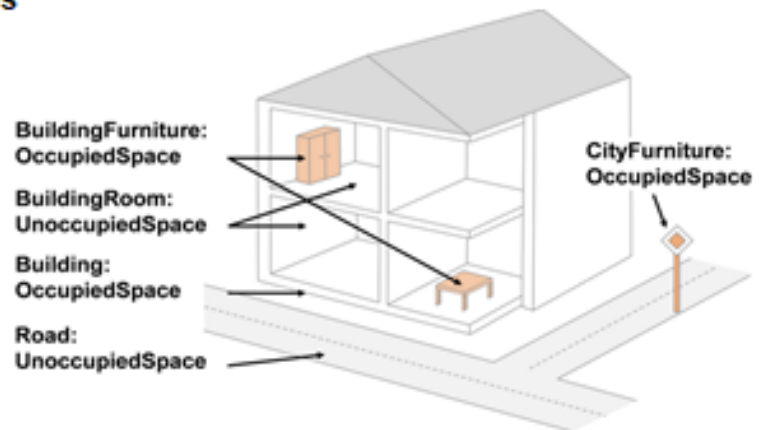
Space Boundaries delimit and connect *Spaces*. Examples are the wall surfaces and roof surfaces that bound a building, the water surface as boundary between the water body and air, the road surface or the digital terrain model.

This is important when converting CityGML to IFC (the open BIM format).

3D Data Standards

Physical and Logical Spaces

- Physical spaces are bounded by physical objects, e.g. buildings, trees...
- Occupied or unoccupied (see figure)
- Logical spaces – thematic, e.g. city district, building unit



Occupied and unoccupied spaces. From: Kulmer et al (2020) CityGML 3.0: New Functions Open Up New Applications

17

Physical and logical spaces

The spaces are further subdivided into physical spaces and logical spaces.

Physical spaces are fully or partially bounded by physical objects. Buildings and rooms are physical spaces as they are bounded by walls and slabs. Traffic spaces of roads are physical spaces as they are bounded by road surfaces against the ground.

Logical spaces, in contrast, are defined according to thematic considerations, such as building unit, city district and security zones in airports.

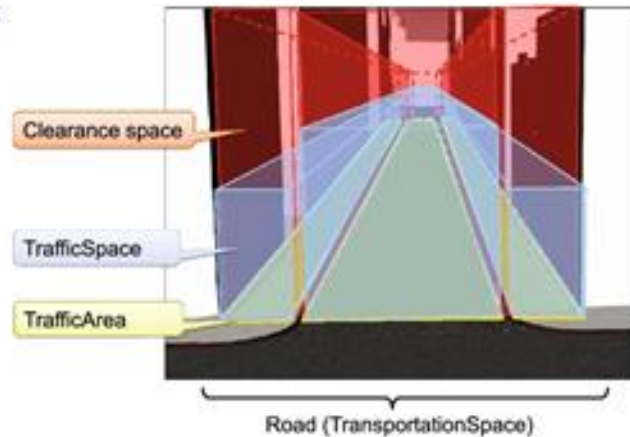
Physical spaces are further classified into occupied spaces and unoccupied spaces.

Occupied spaces represent physical volumetric objects, for example buildings, bridges, trees, city furniture, and water bodies. Unoccupied spaces are building rooms and traffic spaces. Let's look at it on Figure xx.

3D Data Standards

Exteriors, Interiors and Hierarchies

- Each building can have exterior, interior, underground
- Automatic classification and decomposition possible only for outer (visible) parts, not the inner ones
- Applications require more detailed information
- Semi-automatic classification for smaller units possible



Decomposition of transportation space; it is hard to get it in fully automatised way. From: Kuttner et al (2020) CityGML 3.0: New Functions Open Up New Applications

18

Exteriors, Interiors and Hierarchies

Both the exterior and the interior of a building can be described by CityGML.

Further, the interior of a building can have different rooms (BuildingRoom), different storeys or units (BuildingStorey and BuildingUnit), but also installations (e.g. chimneys, antennas, balconies, etc.).

One can model even underground, such as hollow spaces and geological rock layers using the concept of spaces and space boundaries.

One of the major issues for application is how to add the object hierarchies, i.e. how to decompose them into the basic parts and how to deal with the interiors. As point clouds register observable parts, the object decompositions are possible only for the visible surfaces. For example, buildings can be decomposed into walls, roof, and ground surfaces more or less automatically. It is also relatively easy to classify the other city-objects such as vegetation or water-bodies. These can be converted to GML classes.

However, diverse application, such as digital twins used for simulations and analyses, require more detailed semantic knowledge. In best case, it should be full semantic 3D city models, containing hierarchically structured semantic and spatial information. Unfortunately, it is still a fundamental challenge to obtain these by automatic image processing.

A solution for smaller areas or individual buildings is to define the hierarchies (semi-) manually.



3D Data Standards



Coordinates and Elevation

All geometries in CityGML must:

- use 3D coordinate values
- be absolutely georeferenced

They can include terrain models

2.5D – only one Z coordinate for all buildings



Screenshot from ArcGIS Pro using Esri's training data

19

Coordinates and elevation

All geometries in CityGML must use 3D coordinate values. Each 3D point is absolutely georeferenced. That means that all coordinates belong to a coordinate reference system (CRS) and local transformations are not allowed.

This is in contrast to BIM, where use of local coordinates is common.

CityGML can include representation of digital terrain models (DTMs), for instance as point clouds over raster data or TINs. The LOD concept even allows combining several terrain variants in different resolutions. That can be for example a fine-resolution TIN embedded into a gridded DTM of a large area.

Sometimes one meets term 2.5D data. In that case, there is same Z for each (X,Y) position. Such 2.5D models keep costs down in comparison with 3D models, being still more useful for space analyses compared to 2D data.

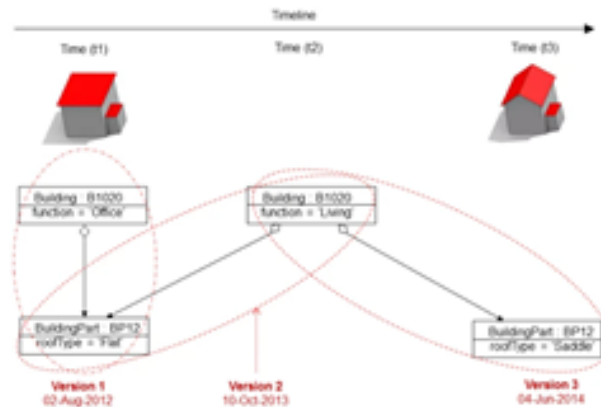
3D Data Standards



Co-funded by
the European Union

4D – Time Dimension

- Important in Smart Cities, Digital Twins
- Versioning module – slow changes
- Dynamizer module – fast changes, e.g. sensor data

[illegible]

Example of versions representing modifications of a building (up)
Representation of different versions of city objects within one CityGML dataset
encoded in GML (left)
From: Kutner et al (2020) CityGML 3.0: New Functions Open Up New
Applications

20

4D – time dimension

Time dimension is more and more important in applications like smart cities and digital twins.

The time-dependent properties are managed by Versioning module and Dynamizer module, two of the blue specific-aspect modules in the conceptual model (Fig XX).

The Versioning module manages qualitative changes that are slower in nature.

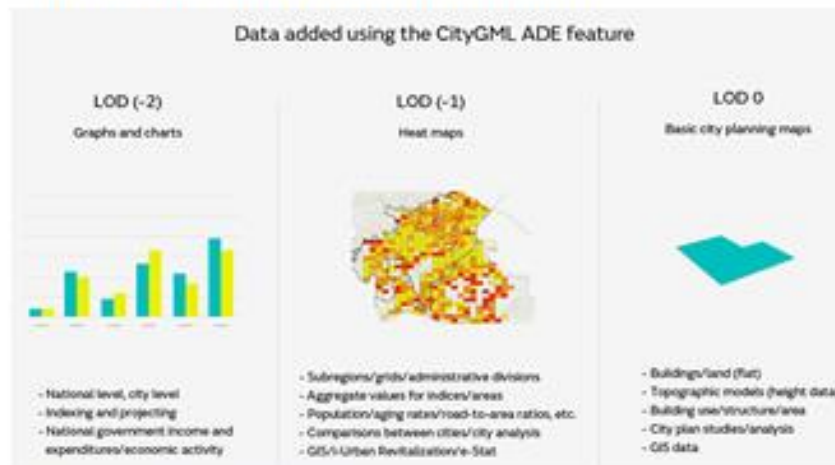
All features can have the attributes `creation-Date` and `termination-Date`, as well as `validFrom` and `validTo`. These represent a specific version of an object at a given time. Hence, the CityGML dataset can be queried for how the city model look at a specific point in time.

The Dynamizer module manages quantitative changes with frequent or dynamic variations of object properties. These can be real-time sensor observations, data from IoT devices or from simulations. Examples are traffic density, air pollution, electricity consumption, solar irradiation or position of moving objects.

In this case, only some of the properties of otherwise static objects need to have time-varying values.

3D Data Standards

Application Domain Extensions, ADEs



[3d-urban-models-04-en.png \(1200x650\) \(munata.com\)](#)

21

- ADEs facilitate to add new classes, attributes or relationships
- e.g. Energy ADE
- Utility Network ADE

Application Domain Extensions, ADEs in CityGML

CityGML 3.0 core data model can be extended by ADEs, Application Domain Extensions.

ADEs facilitate the systematic extension of the CityGML conceptual model by new classes, attributes, and relations for specific application domains.

That is to say that ADEs are used when practitioners want to model additional features and when GenericCityObjects and generic attributes are not systematic enough for the purpose.

Many ADEs have been developed for diverse applications; for example, the Energy ADE to support energetic analyses of buildings or the Utility Network ADE for representation and analysis of multiple networks.

With CityGML 3.0, ADEs become platform-independent models at a conceptual level. As such, they can be freely combined with the original Conceptual-model classes.

A comprehensive discussion of existing CityGML ADEs is provided by Biljecki et al. (2018) CityGML Application Domain Extension (ADE): overview of developments, Open Geospatial Data, Software and Standards, DOI: [10.1186/s40965-018-0055-6](https://doi.org/10.1186/s40965-018-0055-6).



3D Data Standards

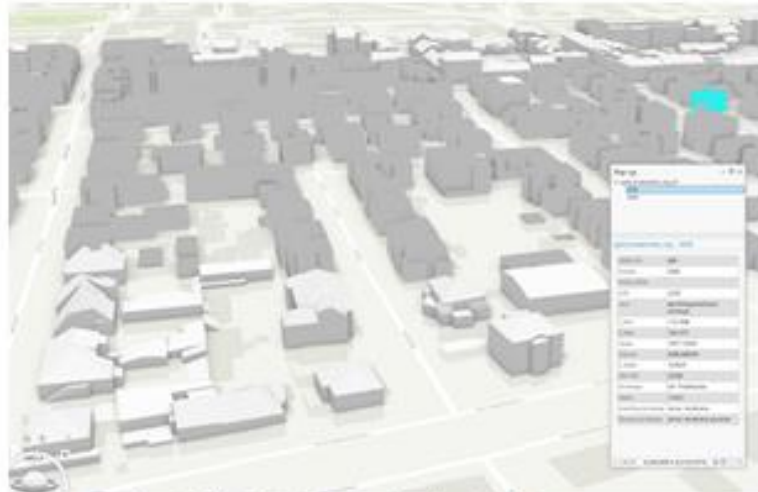


Data Quality

City models are complex
large datasets

Data quality is crucial issue

- Accuracy
- Completeness
- Usability
- Consistency
- Uniqueness



Semantic 3D model of Zagreb, Croatia. Screenshot from ArcGIS Pro.

22

Data Quality

The city-models are highly complex with huge datasets and lot of information originating from multiple sources. Thus, data-quality is a crucial issue to be considered. From bad data come bad decisions.

There are multiple dimensions of the data quality. These dimensions can have equal or varying weights, but all of them should be covered.

In case of spatial data, main quality dimensions are Accuracy and Completeness.

Positional Accuracy: Do the geographic coordinates of a feature correspond with the coordinates of the object in the real world? Within the same reference system, of course.

Thematic Accuracy: Do the data correctly represent classifications associated with locations or objects? E.g. is a river in model really a river in reality?

Temporal Accuracy: Are the data up-to-date and do they represent reality for a required period of time?

Topological Accuracy: Do the spatial relationship between features properly reflect their position in real world?

Completeness: Are all available data points in the database; are there many missing values for any required data?

We can add even other dimensions as:

Usability: Do the data align with the needs of the end-user? Is the level of detail appropriate for the purpose?

Consistency: Are the data synchronised in all data sources? Are there any non-possible values?

Uniqueness: Are there any duplicates in the data? Does everybody use same dataset?



3D Data Standards

Encodings of CityGML

GML – Geography Markup Language

CityGML is name both for:

- XML-based GML encoding
- conceptual data-model

Issued by Open Geospatial
Consortium

CityGML 3 allows data to be encoded
in XML, in JSON or database schemas

Three encodings:

- XML-based →



- JSON-based →



- SQL-based →



23

Encodings of CityGML

First, we should note that CityGML is the name both for GML encoding and for the data-model.

GML stands for the [Geography Markup Language version 3.1.1 \(GML3\)](#). It is the extensible international standard for spatial data exchange issued by the [Open Geospatial Consortium \(OGC\)](#) and the [ISO TC211](#), enabling easy and free access to all the international community.

CityGML 3 allows data to be encoded in GML/XML, but also in JSON or database schemas.



3D Data Standards



XML Encoding

Original encoding of CityGLM

- verbose
- hierarchical
- complex
- not adapted for the web

Not much used any more

```
<?xml version="1.0" encoding="UTF-8"?>
<CityModel xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:bdg="http://www.opengis.net/citygml/building/2.0"
  xsi:schemaLocation="http://www.opengis.net/citygml/2.0"
  <cityObjectMember>
    <bdg:Building gml:id="9ad6451673c7">
      <bdg:function>070</bdg:function>
      <bdg:lodSolid>
        <gml:Solid>
          <gml:exterior>
            <gml:CompositeSurface>
              <gml:surfaceMember>
                <gml:Polygon>
                  <gml:exterior>
                    <gml:LinearRing>
                      <gml:pos>0.0 0.0 0.0</gml:pos>
                      <gml:pos>0.0 1.0 0.0</gml:pos>
                      <gml:pos>1.0 1.0 0.0</gml:pos>
                      <gml:pos>1.0 0.0 0.0</gml:pos>
                      <gml:pos>0.0 0.0 0.0</gml:pos>
                    </gml:LinearRing>
                  </gml:exterior>
                </gml:Polygon>
              </gml:surfaceMember>
            </gml:CompositeSurface>
          </gml:exterior>
        </gml:Solid>
      </bdg:lodSolid>
    </bdg:Building>
    <bdg:Building gml:id="jdh70sa">
      ...
    </bdg:Building>
  </cityObjectMember>
</CityModel>
```

24

XML encoding

Originally, CityGML were XML files. An example is given in the figure. As we can see, the XML encoding is verbose, hierarchical, complex, and not adapted for the web. Therefore, it is not much used any more (in CityGML 3).



3D Data Standards



CityJSON Encoding I

- the most used alternative to XML encoding
JSON - JavaScript Object Notation
- even CityJSON is an OGC standard
- coordinates are stored on one place only, in a
separate array, i.e. the "vertices"

CityJSON allows full compression and
simplifies file structure, compared to
XML encoding (right figure)
Example of coordinates in CityJSON (left
figure)

```
1  "vertices": [
2    [23234, 111009, 1392],
3    [29456, 115134, 1007],
4    [54508, 229995, 1961],
5    ...
6    [23134, 625134, 203]
7  ]
```

```
1  "CityObjects": {
2    "id-1": {
3      "type": "Building",
4      "attributes": {...},
5      "children": ["id-2", "id-3"],
6      "geometry": [{...}]
7    },
8    "id-2": {
9      "type": "BuildingPart",
10     "parents": ["id-1"],
11     "geometry": [{...}]
12     ...
13   },
14   "id-3": {
15     "type": "BuildingPart",
16     "parents": ["id-1"],
17     "geometry": [{...}]
18     ...
19   }
20 }
```

25

CityJSON Encoding I

Currently, the most used alternative to XML encoding of CityGML is CityJSON (JavaScript Object Notation).

Even CityJSON is a standard of the Open Geospatial Consortium (OGC). The current version of CityJSON is 1.1.3 and it provides a number of advantages over CityGML-XML.

[CityJSON Specifications 1.1.3](#)

First, the coordinates are stored on one place only, in a separate array, i.e. the "vertices" property of the CityJSON object. Geometric primitives then refer to the position of a vertex in the array.

3D Data Standards

CityJSON Encoding II

CityJSON:

- is well suitable for web application
- reduces data-size; JSON file takes about 6x less space than XML
- can be stored both in relational as well as in no-SQL database

Two kinds of city objects – 1st and 2nd level (parents and children)



[CityJSON Specifications 1.1.3](#)

26

CityJSON Encoding II

Then, JSON dominates the web, which makes data exchange between applications to be an easy task.

Third, JSON reduces data-size; JSON file takes about 6x less space than XML.

Last, JSON files can be stored both in relational as well as in no-SQL databases.

In JSON, the schema of CityGML has been flattened out compared to XLM encoding, as demonstrated by the figure.

There are two kinds of city objects in CityJSON: 1st level, or parents, that can exist by themselves, like building, bridge or water body.

And 2nd level, or children, that need to have a “parent” to exist. Examples can be Building part, furniture or room. More examples are shown by the figure. Both 1st- and 2nd-level city objects are stored in the dictionary "CityObjects".

A city object must have at a minimum a "geometry" property. If attributes are to be stored, they have to be in the "attributes" property. This simplifies the work of the developer because there is a single point of entry for all geometries and attributes (unlike XML-encoded CityGML).

To explore a simple CityJSON model, go to: <https://ninja.cityjson.org/> which is a free online JSON-viewer, and upload the LoD3_Railway.city.json, available with the lecture. Alternatively, you can download it at: [Datasets | CityJSON](#), together with other .json files.

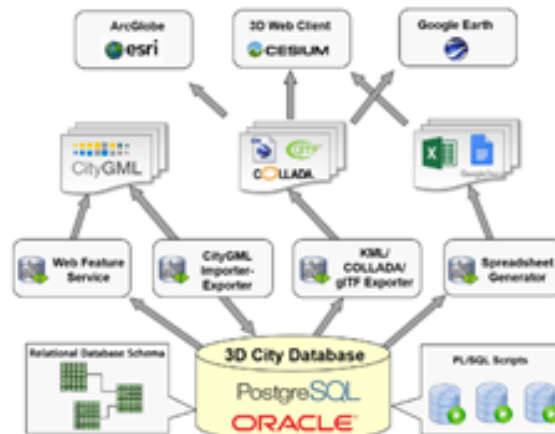


3D Data Standards

3D City Database Encoding

- database schema called 3DCityDB
- not official standard
- even open source software

[3DCityDB Database – Homepage](http://www.3dcitydb.net/3dcitydb/fileadmin/TUM_Workshop/Documents/Tutorial.pdf)



3DCityDB - a 3D geodatabase solution for the management, analysis, and visualisation of semantic 3D city models based on CityGML, Yao et al. (2018)

27

3D City Database Encoding

The third CityGML encoding is a database schema called 3DCityDB. It can be implemented both for PostgreSQL and Oracle Spatial. It is an Open-Source software and although it is not an official standard, it is used by several municipalities around the world.

A widely-used software package to deal with 3D models is called 3D City Database, shortened as 3DCityDB. It is a free, open-source database schema for spatially enhanced relational database management systems, such as ORACLE Spatial or PostgreSQL/PostGIS.

On top of the database system, 3DCityDB provides all the tools needed, including maintenance and web-visualisation of the 3D city models.

For Tutorials / course see

http://www.3dcitydb.net/3dcitydb/fileadmin/TUM_Workshop/Documents/Tutorial.pdf



3D Data Standards

Other 3D formats

GML can be combined with many other formats

- Web Feature Service (WFS)
- Web Processing Service (WPS)
- KML/COLLADA or X3D files
- Web 3D Service (W3DS)
- Web Terrain Service (WTS)
- Indoor GLM



Example of a 3D model

28

Other 3D formats

As OGC standard, GML3 can be combined with the full range of other OGC standards. The Web Feature Service (WFS), the Catalog Service (CS-W), the Web Coordinate Transformation Service (WCTS), and the Web Processing Service (WPS) are especially relevant to access, process, and identify CityGML resources.

For 3D visualization, CityGML is a base format from which 3D graphic formats can be easily derived. The rich semantic information of CityGML objects also help in automatic cartographic generalization and symbolization. It is also suitable for generating computer graphics represented e.g. as a KML/COLLADA or X3D files. Corresponding OGC portraying services are the Web 3D Service (W3DS) and the Web Terrain Service (WTS).

There is also Indoor GML for indoor-space data modelling and navigation. It has minimal overlap with CityGML, but it is possible to combine these two (CityGML and Indoor GLM) in one model, if needed.

3D Data Standards

LandInfra

- Another 3D data standard
- Land and civil engineering
- Some overlap with CityGML
- Includes features not available in CityGML



Comparison between IFC, CityGML and LandInfra. Source: LandInfra BIM GIS.pdf

29

LandInfra

Besides CityGML, there is another 3D data standard called LandInfra.

It is a conceptual model for land and civil engineering infrastructure. As such, it is published for predetermined use cases such as facilities, projects, road, railway, survey, land division, wastewater or water distribution systems.

LandInfra has some potential overlap with CityGML, but does not support ADEs and LODs.

On the other hand, it includes some features that are not available in CityGML. These include: enhanced subsurface data modelling, framework to model legal information and storing of survey related information.

It even partly overlaps with IFC, but is not used to bridge the gap between BIM and GIS yet.

To learn more, you can read article [LandInfra BIM GIS.pdf](#)

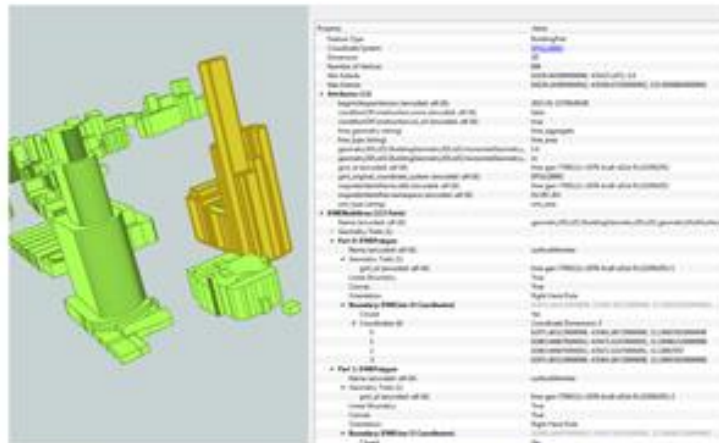
3D Data Standards

Inspire 3D Buildings

Influenced by CityGML,
but simplified

Aim to guarantee
interoperability of spatial
data and services from
different EU countries

Requirements of EU
Directives (e.g. Noise,
Energy Performance)



INSPIRE Buildings GML viewed with Data Inspector. From: [Converting CityGML to INSPIRE 3D Buildings \(Annex III\) \(uafu.com\)](#)

30

Inspire 3D buildings

CityGML has strongly influenced the model for INSPIRE 3D Buildings.

INSPIRE GML includes attributes to support INSPIRE's mandate, many of which are common to other INSPIRE themes.

INSPIRE Buildings is somewhat simplified, compared to CityGML. For instance, windows and doors are not required, parts cannot have subparts, and appearances are simplified.

INSPIRE data specification does not require collection of new data. Its aim is to guarantee interoperability of spatial data and services from different EU countries. This addresses requirements related to European reporting, such as the Noise Directive, the Air Quality Directive or the Energy Performance of Building Directive.



References

Malhotra et al. (2022) Information modelling for urban building energy simulation—A taxonomic review. Building and Environment 208, <https://doi.org/10.1016/j.buildenv.2021.108552>

Ledoux (2018) CityGML and its two encodings CityGML and CityJSON, presentation Delft University

Ken Arroyo Ohori, Hugo Ledoux, and Ravi Peters (2020–2022): 3D modelling of the built environment, available at: [Releases · tudelft3d/3dbook \(github.com\)](https://github.com/tudelft3d/3dbook)

Sahleb (2019) Automatic Conversion of CityGML to IFC, TU Delft, MSc. thesis