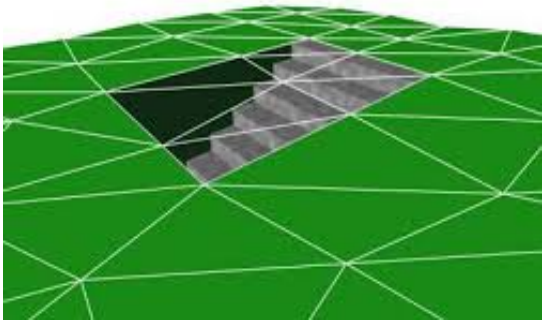
# Introduction

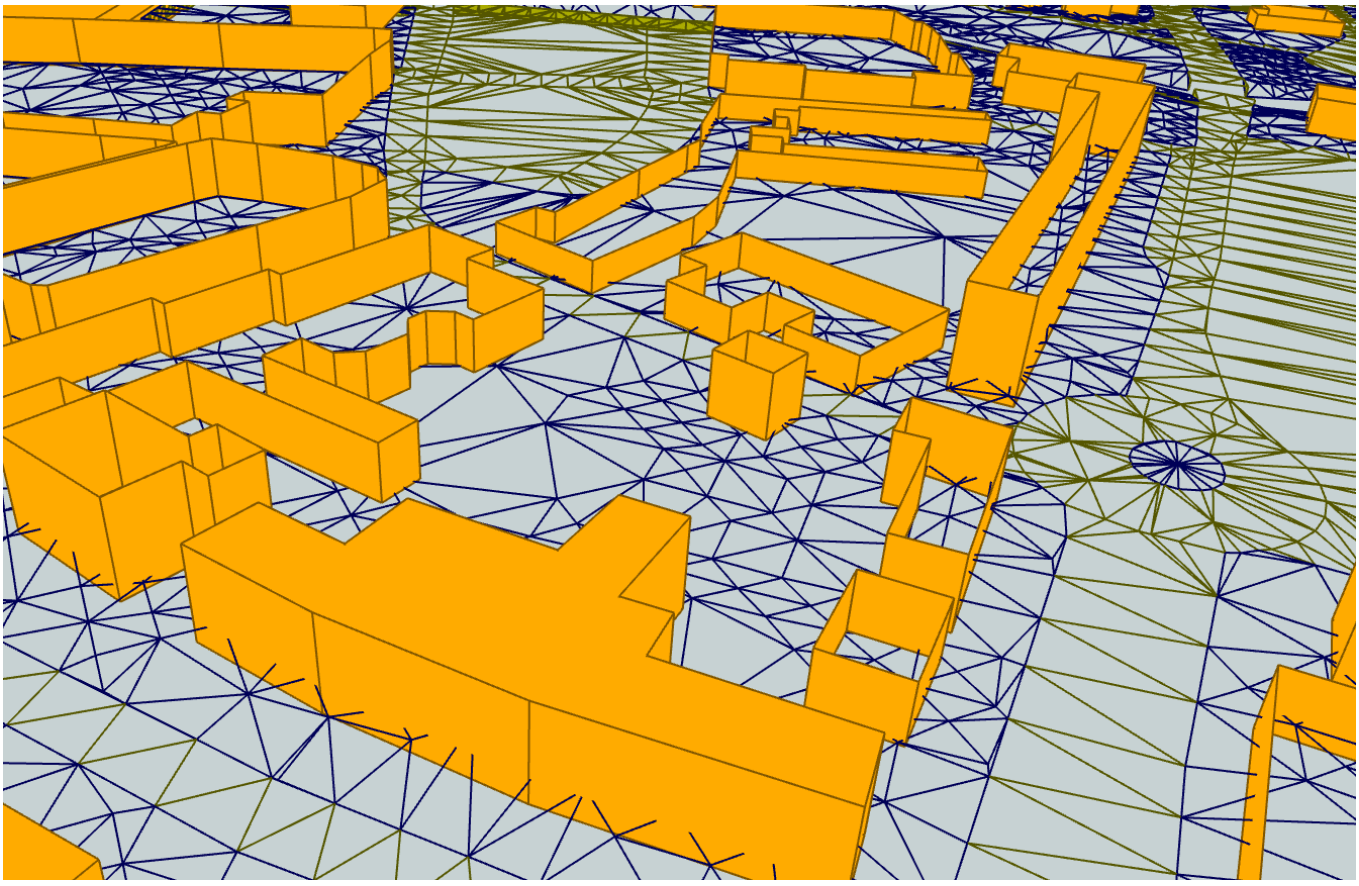
The following description is based on CityGML version 2.0. CityGML is an application schema for the Geography Markup Language version 3.1.1 (aka GML3) from which is draws its geometry classes (refer to Annex D).

CityGML V2.0 UML diagrams, page 5

# Underlying geometric primitives are few (GML based)

## Digital Terrain Model (Relief component)


(Rectified Grid Coverage example: source)



(Triangulated Surface example: source)

As illustrated by the following figure, the **Relief component** (Digital Terrain Model) **surfacic** geometry can be:

- `RectifiedGridCoverage` i.e. a regular grid height field
- `TriangulatedSurface` i.e. a collection of topologically disjoint triangles.

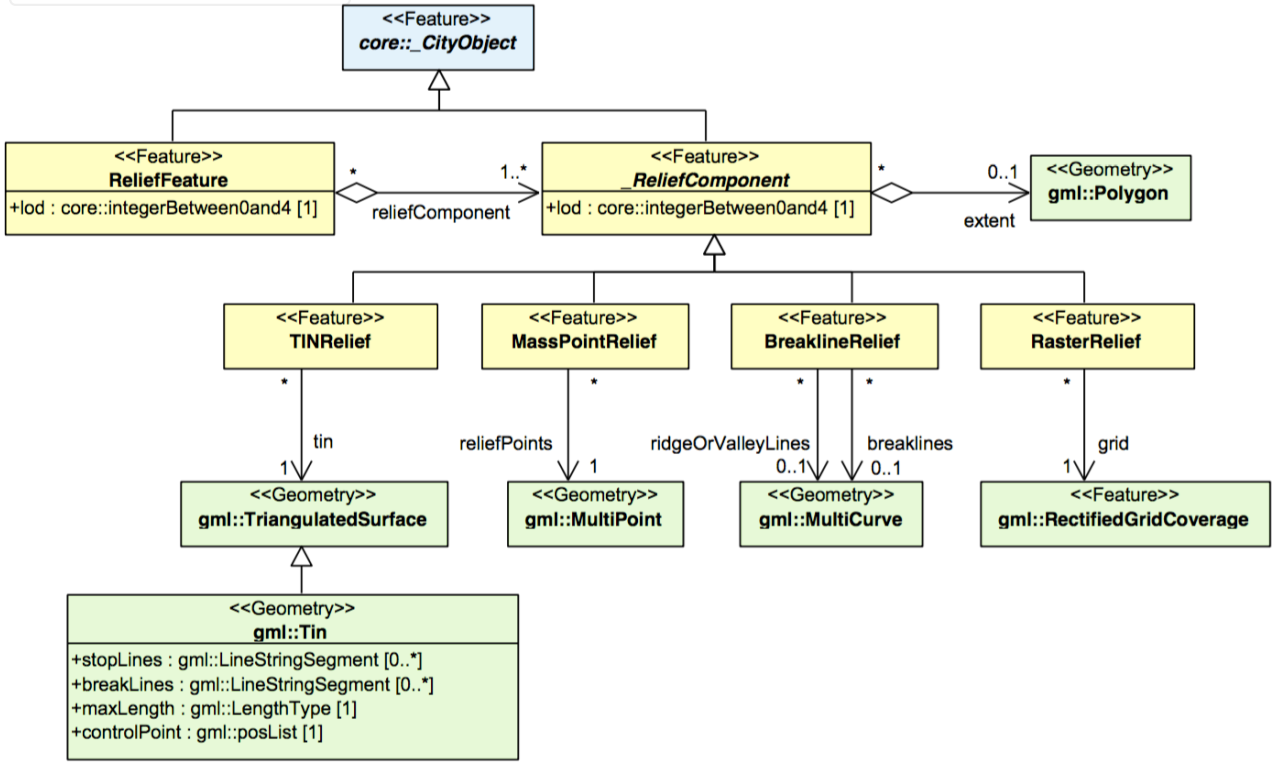
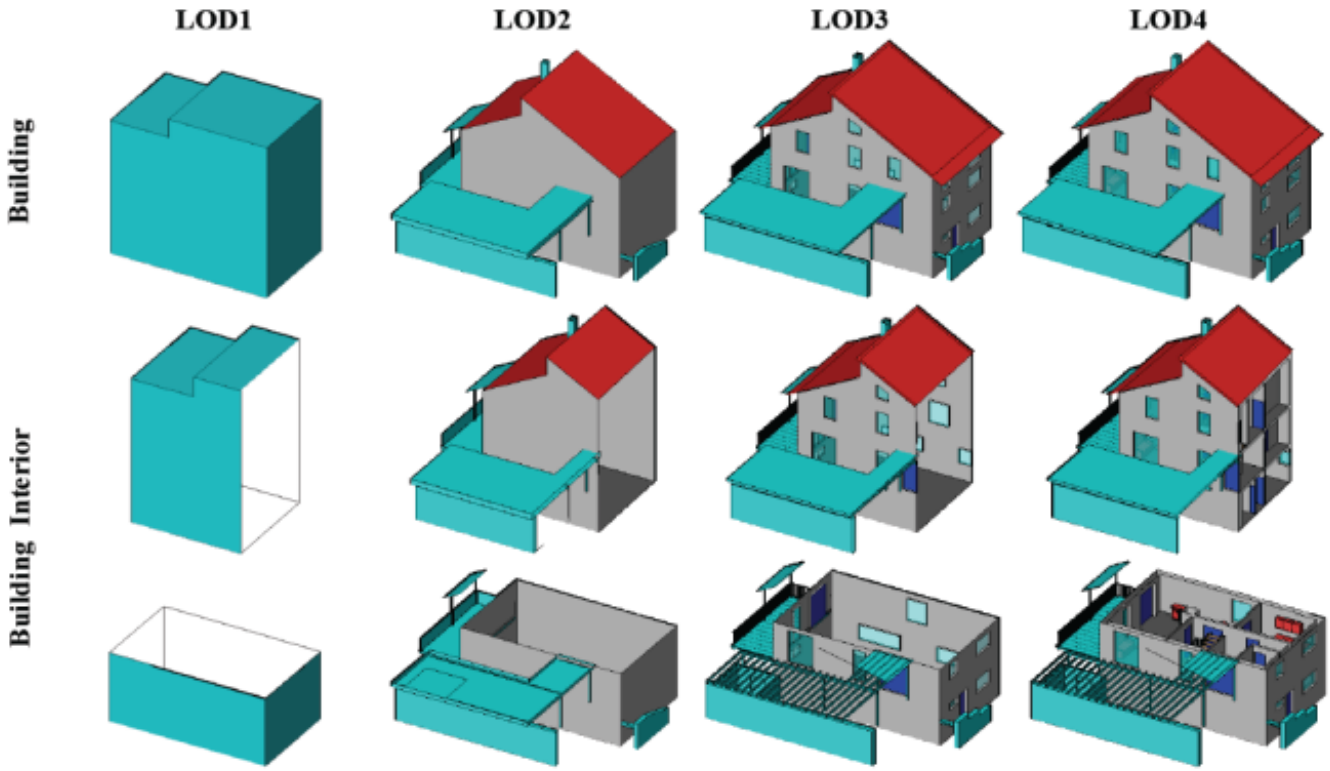


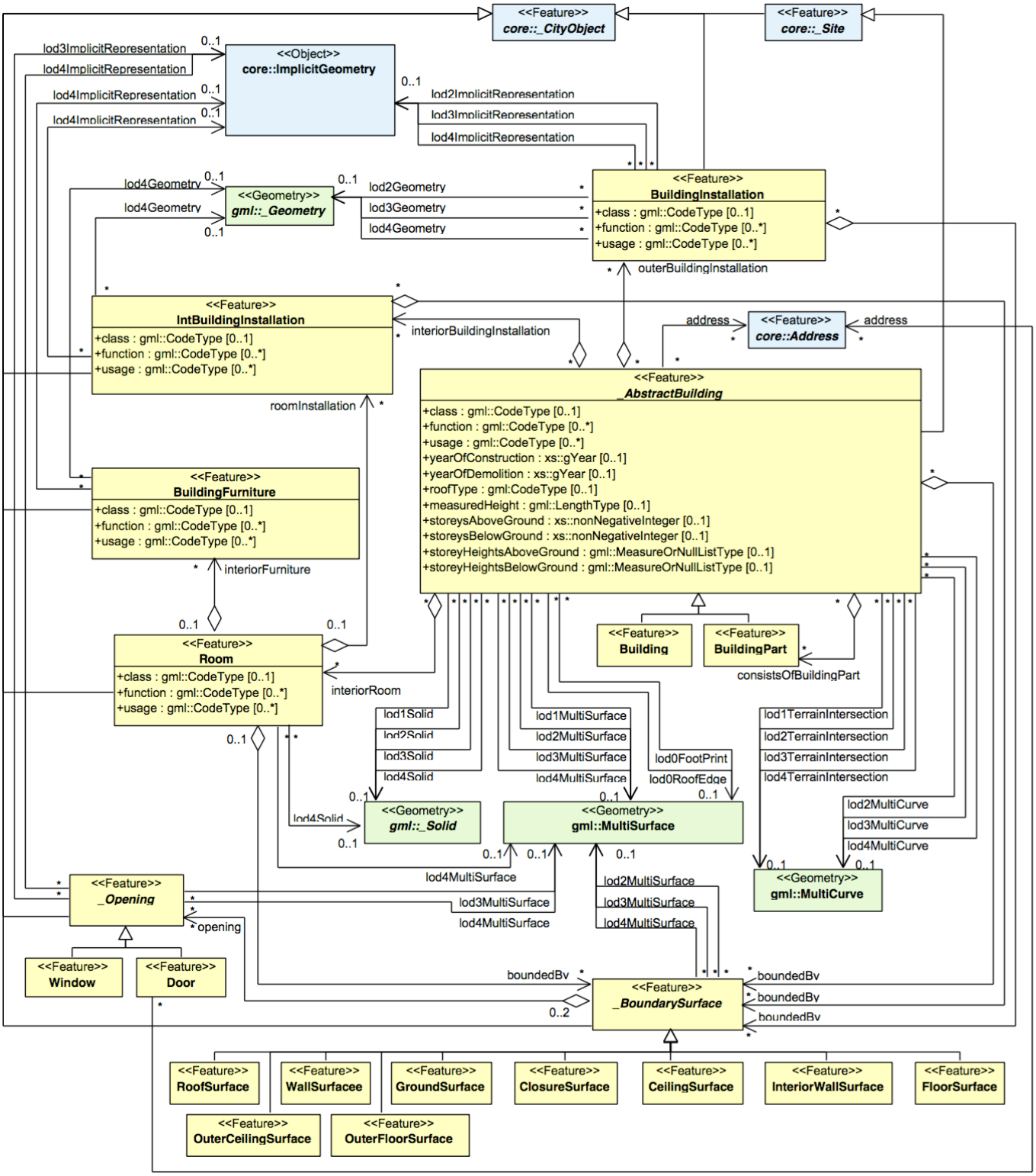

Fig. 24: UML diagram of the Digital Terrain Model in CityGML. Prefixes are used to indicate XML namespaces associated with model elements. Element names without a prefix are defined within the CityGML *Relief* module.

(source: figure 24, page 57 of CityGML version 2.0).

# Buildings

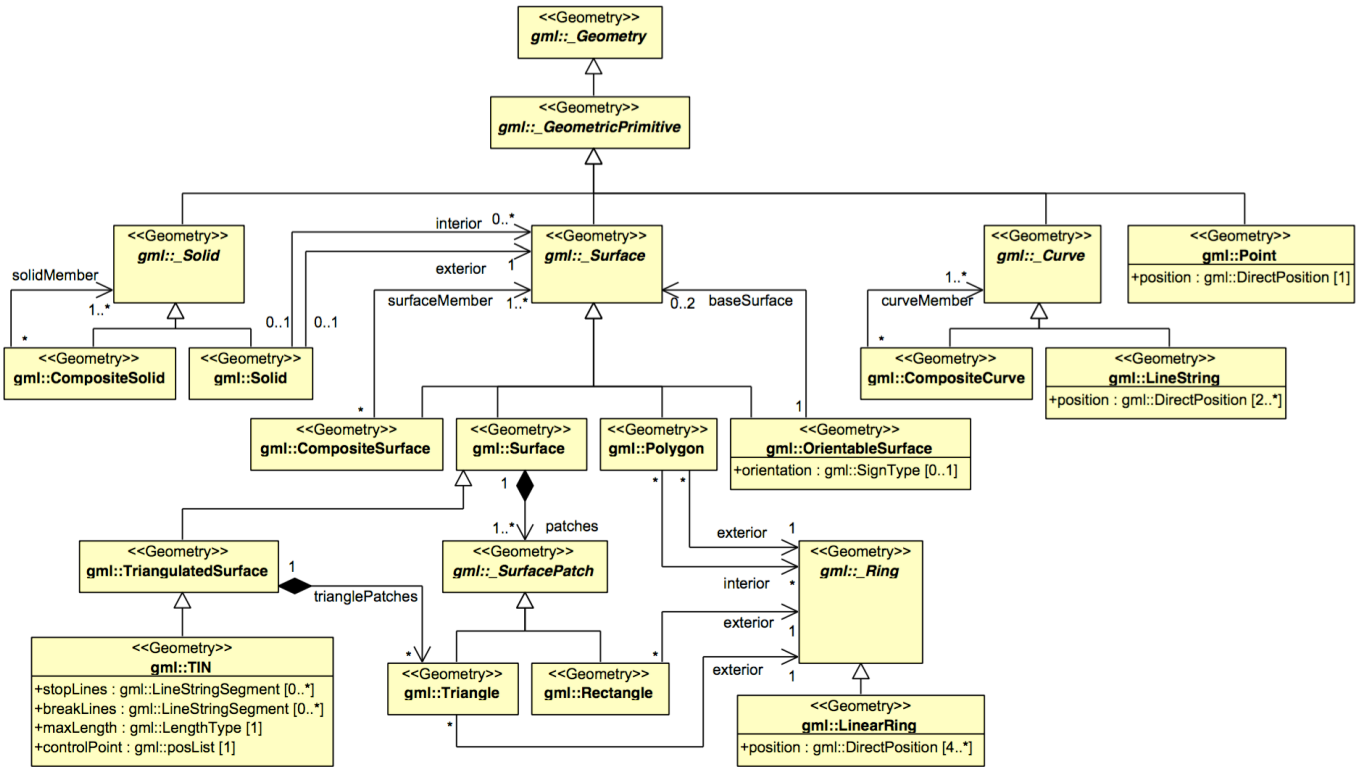



(source)




(source diagram page 11 of CityGML V2.0 UML diagrams)

# GML: everything boils down to `gml:_Ring`




(source diagram page 5 of CityGML V2.0 UML diagrams)

A Triangle patch xml example (Source):

```
<gml:trianglePatches>
  <gml:Triangle>
    <gml:exterior>
      <gml:LinearRing>
        <gml:posList>
          -6.0 5.0 0.0
           0.0 5.0 0.0
          -6.0 9.0 0.0
          -6.0 5.0 0.0
        </gml:posList>
      </gml:LinearRing>
    </gml:exterior>
  </gml:Triangle>
  <gml:Triangle>
      ...
  </gml:Triangle>
</gml:trianglePatches>
```

Refer to gml's definitions: TriangulatedSurface, Patches and Triangle.

# No hope for “fixing the topology”

As specified in GML 3.1.1 spec, page 23: “A GML object is an XML element of a type derived (⋯) from AbstractGMLType. From this derivation, a GML object may have a gml:id attribute.”
We could hope to provide each `gml:pos` (spatial position) with an id, and define a triangle as three ids of positions. Alas the norm imposes to repeat position coordinates and thus **CityGML does not allow to topologically relate vertices of two triangles**. This has to be indirectly done through coordinates comparison⋯

# Semantic vs Geometry hierarchies (possibly coherent)

Semantic and geometry hierachies can (and thus will) lack coherence:

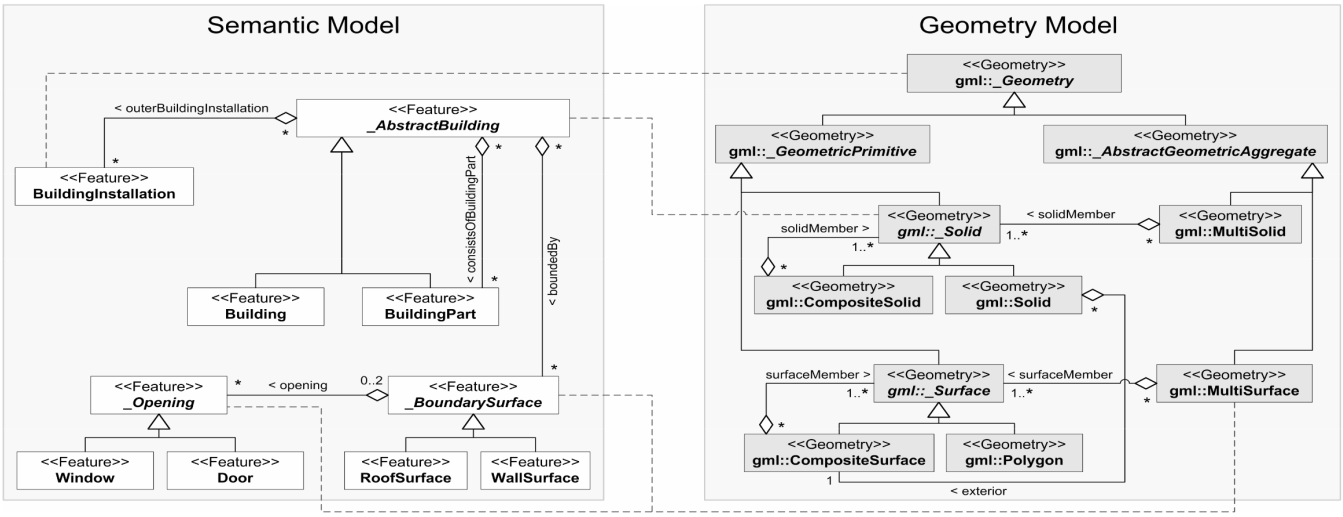



Figure 1. UML class diagram (Booch et al., 1997) of CityGML’s semantic and geometry model (left: excerpt from the building model, right: excerpt from ISO 19107 Spatial schema). Both structures allow for aggregations on several levels.

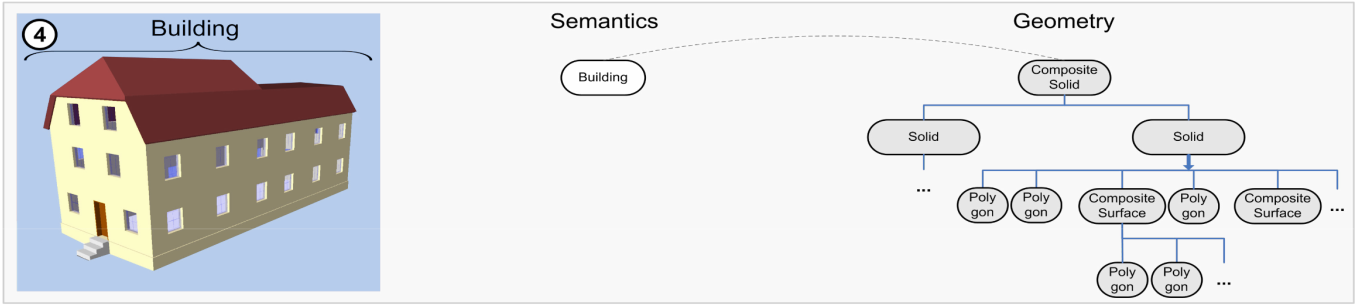



Figure 4. Simple object with complex structured geometry (Case 4).

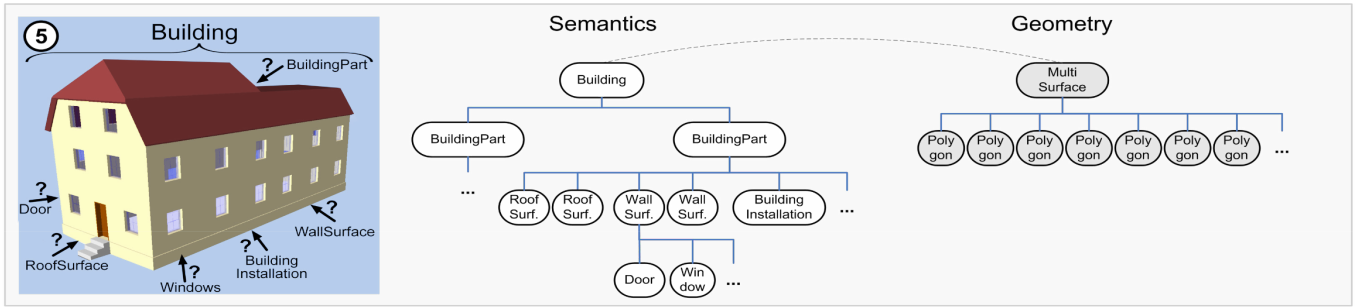



Figure 5. Object with complex semantic structure and detailed, but unstructured and uncorrelated geometry (Case 5).

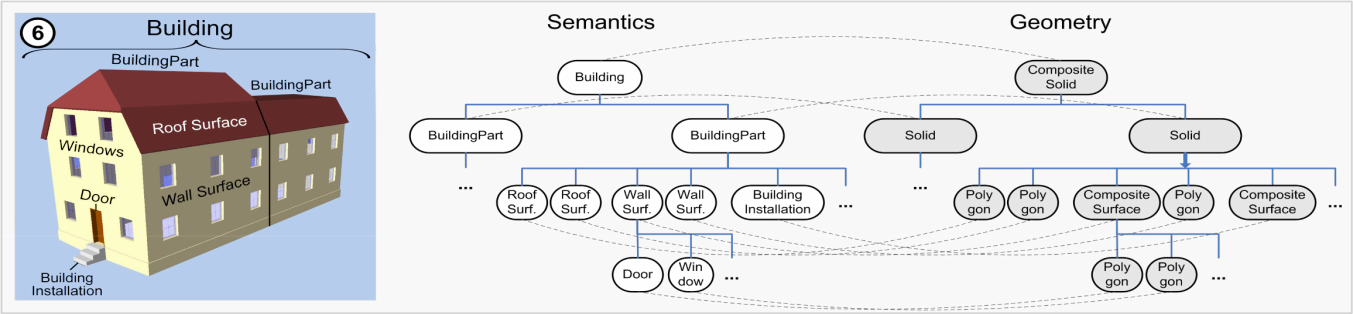



Figure 6. Complex object with fully coherent spatio-semantic structure (Case 6).

# Conclusion

```
CityGMLFeature (Buildings, Terrain)
 --> boundary surface
    --> gml:multisurface
       --> gml:_Surface
          --> gml:Triangle
             --> gml:_Ring
                --> list of 3 points coordinates
```

**CityGML surfacic geometries are polygon soups i.e. groups of unorganized triangles, with generally no relationship whatsoever. Geometrical "proximity" (based on coordinates comparison) blended with hierarchic semantic information can be used to reconsitute 2-manifold (flat) polygons / triangle based surfaces.**

# References


- CityGML version 2.0
- CityGML V2.0 UML diagrams
- Stadler, A., Kolbe, T. H. (2007): Spatio-Semantic Coherence in the Integration of 3D City Models