Three-dimensional primitives in the context of the CityGML QIE

Hugo Ledoux
h.ledoux@tudelft.nl

October 21, 2014

1 What is an ISO 19107 solid?

ISO 19107 defines different geometric primitives[1]: a 0D primitive is a GM_Point, a 1D a GM_Curve, a 2D a GM_Surface, and a 3D a GM_Solid. A primitive is built with lower-dimensional primitives, eg in Figure 1 the GM_Surface is composed of 2 (closed) GM_Curves, which are composed on several GM_Points. Observe that primitives do not need to be linear or planar, ie curves defined by mathematical functions are allowed.

In our context, the following three definitions from ISO (2003) are relevant:

Definition 1. *A GM_Solid is the basis for 3-dimensional geometry. The extent of a solid is defined by the boundary surfaces. The boundaries of GM_Solids shall be represented as GM_SolidBoundary. [...] The GM_OrientablesSurfaces that bound a solid shall be oriented outward.*

Definition 2. *A GM_Shell is used to represent a single connected component of a GM_SolidBoundary. It consists of a number of references to GM_OrientableSurfaces connected in a topological cycle (an object whose boundary is empty). [...] Like GM_Rings, GM_Shells are simple.*

Definition 3. *A GM_Object is* simple *if it has no interior point of self-intersection or self-tangency. In mathematical formalisms, this means that every point in the interior of the object must have a metric neighbourhood whose intersection with the object is isomorphic to an n-sphere, where n is the dimension of this GM_Object.*

[1] All the geometric primitive have the prefix 'GM_'

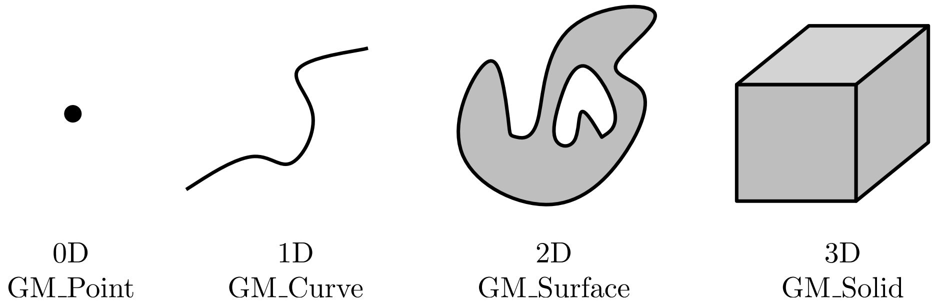


Figure 1: ISO 19017 primitives.

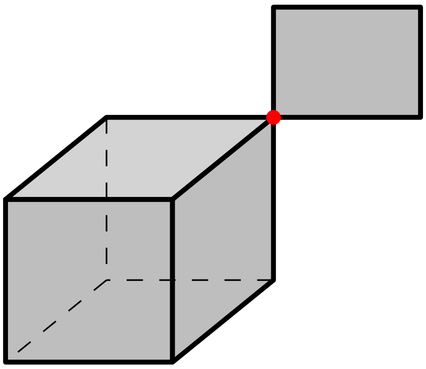


Figure 2: The red vertex is a non-manifold vertex since the neighbourhood aroud it is not topologically equivalent to a plane.

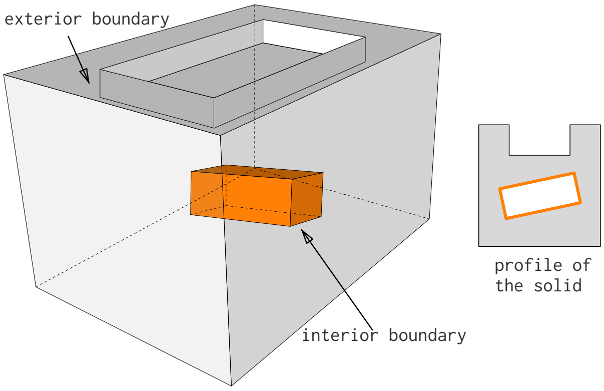


Figure 3: One solid which respects the international definition. It has one exterior shell and one interior shell (forming a cavity).

The bounding surfaces of a shell thus form a *closed* and *orientable two-dimensional manifold* (or 2-manifold for short). A 2-manifold is a space that is topologically equivalent to $\mathbb{R}^2$, the 2D Euclidean space. An obvious example is the surface of the Earth, for which near to every point the surrounding area is topologically equivalent to a plane. If we have a shell stored in a data structure, it implies that each edge is guaranteed to have a maximum of two incident faces, and that around each vertex the incident faces from one 'umbrella', as Figure 2 shows. To be valid shell, the 2-manifold should be closed, ie there should not be 'holes' in the surface (in other words, it should be watertight).

Figure 3 shows a solid that respects that definition. First observe that the solid is composed of two shells (both forming its boundaries), one being the exterior and one being the interior shell. The exterior shell has eleven surfaces, and the interior one six. An interior shell creates a cavity in the solid—cavities are also referred to as "voids" or holes in a solid. A solid can have no inner shells, or several. Observe that a cavity is not the same as a hole in a torus (a donut) such as that in Figure 4: it can be represented with one exterior shell having a genus of 1 and no interior shell.

2 Primitives in CityGML

CityGML uses the ISO 19107 geometric primitives for representing the geometry of its objects. However, as shown in Figure 5 only a subset is used, with the following two restric-

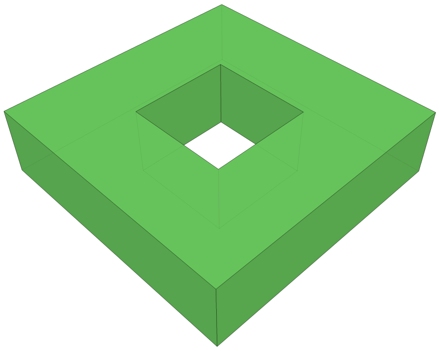
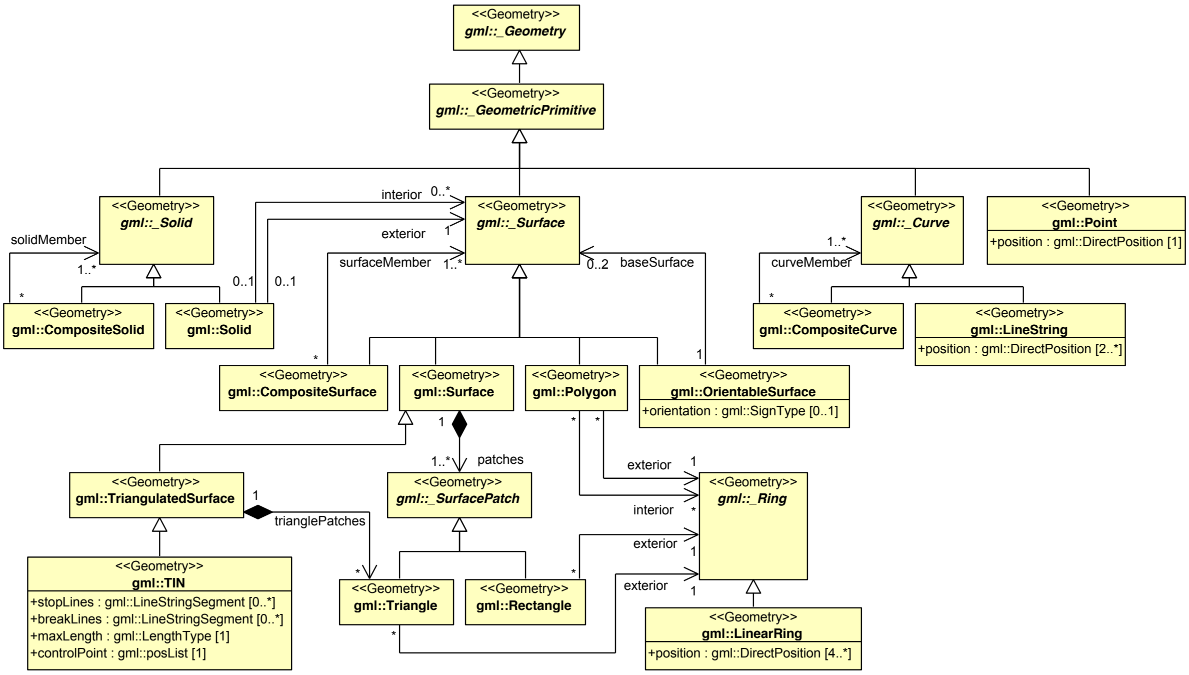


Figure 4: A ‘squared torus’ is modelled with one exterior boundary formed of ten surfaces. Notice that there are no interior boundary.



<<Geometry>> gml::_Geometry
<<Geometry>> gml::_GeometricPrimitive
<<Geometry>> gml::_Solid
<<Geometry>> gml::_Surface
<<Geometry>> gml::_Curve
<<Geometry>> gml::Point
+position : gml::DirectPosition [1]
solidMember
1..*
*
<<Geometry>> gml::CompositeSolid
<<Geometry>> gml::Solid
0..1
0..1
interior 0..*
exterior 1
surfaceMember 1..*
0..2 baseSurface
curveMember
1..*
*
<<Geometry>> gml::CompositeCurve
<<Geometry>> gml::LineString
+position : gml::DirectPosition [2..*]
*
<<Geometry>> gml::CompositeSurface
<<Geometry>> gml::Surface
<<Geometry>> gml::Polygon
1
<<Geometry>> gml::OrientableSurface
+orientation : gml::SignType [0..1]
1
1..* patches
*
*
exterior 1
interior *
exterior 1
exterior 1
<<Geometry>> gml::TriangulatedSurface
1
trianglePatches
<<Geometry>> gml::_SurfacePatch
<<Geometry>> gml::_Ring
<<Geometry>> gml::TIN
+stopLines : gml::LineStringSegment [0..*]
+breakLines : gml::LineStringSegment [0..*]
+maxLength : gml::LengthType [1]
+controlPoint : gml::posList [1]
*
<<Geometry>> gml::Triangle
*
<<Geometry>> gml::Rectangle
*
<<Geometry>> gml::LinearRing
+position : gml::DirectPosition [4..*]

Figure 5: UML diagram of the CityGML geometry model.

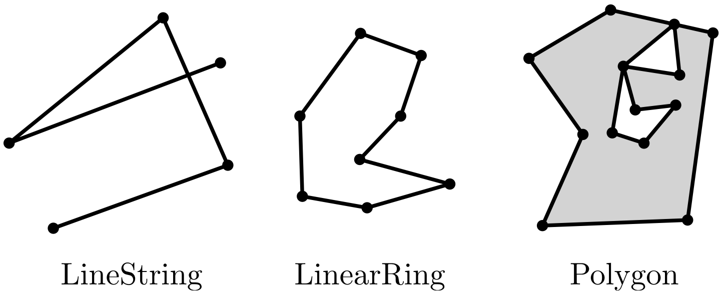


Figure 6: 2D CityGML primitives.

tions: (1) GM_Curves can only be *linear* (thus only LineStrings and LinearRings are used); (2) GM_Surfaces can only be *planar* (thus Polygons are used). The primitives for constructing Shells and Solids are shown in Figure 6.

Definition 4. *A LineString is a Curve with linear interpolation between each Point; each two consecutive Points defines a line segment. A LinearRing is a LineString that is both closed and simple.*

Definition 5. *A Polygon is a surface patch that is defined by a set of boundary curves and an underlying surface to which these curves adhere. The default is that the curves are coplanar and the polygon uses planar interpolation in its interior.*

Each shell of a solid is thus composed of Polygons, and these can have inner rings (which are often referred to as holes). Observe that the top polygon of the solid in Figure 3 has one inner ring, but that other polygons "fill" that hole so that the exterior shell is "watertight" (ie it has no holes and is thus closed).

3 QIE = no cavaties

It should be noticed that during the QIE, only buildings in LOD1, LOD2 and LOD3 will be considered, and, as a consequence, cavities in solids are ignored.

This imply that a solid has exactly *one* shell representing its exterior boundary. However, this does *not* mean that inner rings in the boundary surfaces are excluded, because simple LOD1 buildings having for instance an inner yard require inner rings, as in Figure 4 shows; an alternative to representing inner rings is to decompose the face into several polygons, eg to triangulate it.

4 Requirements for validity of the 3D primitives

Each primitive used to construct a higher-dimensional primitive should be valid. This means that in order to validate a solid, we need to also ensure that each ring and polygon used be valid. For rings and polygons, observe that these will be embedded in 3D (ie the points used to construct rings will have (x, y, z) coordinates).

1. Polygons are topologically closed;
2. The boundary of a Polygon consists of a set of LinearRings that make up its exterior and interior boundaries;
3. No two Rings in the boundary cross and the Rings in the boundary of a Polygon may intersect at a Point but only as a tangent, eg

$$\forall P \in \mathit{Polygon}, \forall c1, c2 \in P.\mathit{Boundary}(), c1 \neq c2,$$

$$\forall p, q \in \mathit{Point}, p, q \in c1, p \neq q, [p \in c2 \Rightarrow q \notin c2];$$

4. A Polygon may not have cut lines, spikes or punctures eg:

$$\forall P \in \mathit{Polygon}, P = P.\mathit{Interior}.\mathit{Closure};$$

5. The interior of every Polygon is a connected point set;
6. The exterior of a Polygon with 1 or more holes is not connected. Each hole defines a connected component of the exterior.

Figure 7: The six assertions for the validity of a 2D polygon embedded in $\mathbb{R}^2$ (OGC, 2006, pages 26–27).

4.1 Rings & Polygons

According to the ISO 19107 abstract specifications, the different boundaries of a polygon are allowed to interact with each other, but only under certain circumstances. The implementation specifications defined by the OGC (OGC, 2006) give clear requirements, see the assertions in Figure 7.

Some concrete examples of invalid polygons are shown in Figure 8, and here are a few examplanations:

1. Each ring should be closed (p_{11}): its first and its last points should be the same.
2. Each ring defining the exterior and interior boundaries should be *simple*, ie non-self-intersecting (p_1 and p_{10}). Notice that this prevents the existence of rings with zero-area (p_6), and of rings having two consecutive points at the same location. It should be observed that the polygon p_1 is not allowed (in a valid representation of the polygon, the triangle should be represented as an interior boundary touching the exterior boundary).
3. The rings of a polygon should not cross (p_3, p_7, p_8 and p_{12}) but may intersect at one tangent point (the interior ring of p_2 is a valid case, although p_2 as a whole is not since the other interior ring is located *outside* the interior one). More than one tangent point is allowed, as long as the interior of the polygon stays connected (see below).
4. A polygon may not have cut lines, spikes or punctures (p_5 or p_6); removing these is known as the *regularisation* of a polygon (a standard point-set topology operation).
5. The interior of every polygon is a connected point set (p_4).
6. Each interior ring creates a new area that is disconnected from the exterior. Thus, an interior ring cannot be located outside the exterior ring (p_2) or inside other interior rings (p_9).

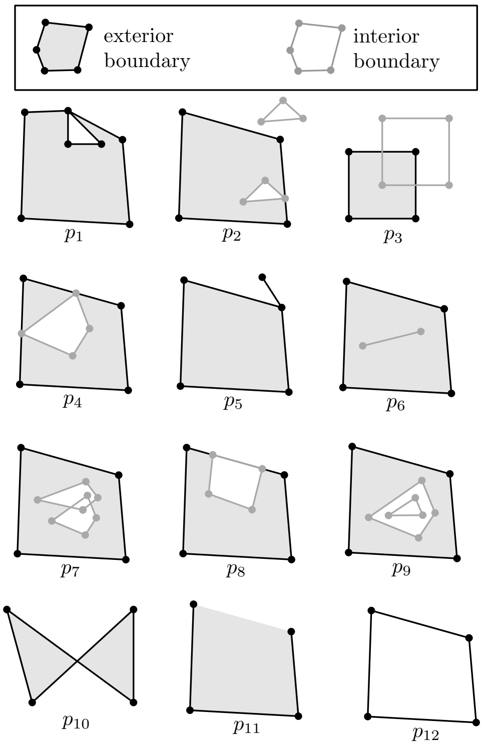


Figure 8: Some examples of invalid polygons. Polygon p_{12} has its exterior and interior rings defined by the same geometry.

4.2 Planarity requirement

A polygon must be planar, ie all its points (used for both the exterior and interior rings) must lie on a plane. Interestingly, the concept of *tolerance* is not mentioned in the standards ISO and OGC.

For the QIE, two requirements are proposed:

1. the distance between every point forming a polygon and a plane is less than ϵ_1, a given tolerance (eg 1mm). This plane should be a plane fitted with least-square adjustment.
2. the distance between every point forming a polygon and all the planes defined by all possible combinaisons of 3 non-colinear points is less than ϵ_1.

The second requirement is to ensure that cases such as that in Figure 9 are detected. From algorithmic point of view, the definition is not very efficient, but in practice it can be implemented with a triangulation of the polygon (any triangulation): the orientation of the normal of each triangle must not deviate more than than a certain usef-defined tolerance ϵ_2 (eg 1 degree).

4.3 Snapping tolerances for vertices

Geometries modelled in CityGML, and in GML in general, store amazingly very little topological relationships. For instance, all six surfaces of a cube are stored independently. This means that the coordinates (x, y, z) of a single point (where 3 polygons "meet") is stored 3 times. It is possible that these 3 vertices are not exactly at the same location, eg (0.01, 0.5, 1.0), (0.011, 0.49999, 1.00004) and (0.01002, 0.5002, 1.0007), and that would create problems when validating since there would for example be tiny cracks/overlaps in the cube. The snap tolerance

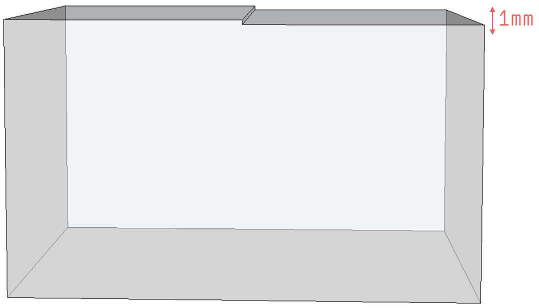


Figure 9: All the points of the top polygon are within 1mm but the polygon cannot be considered planar.

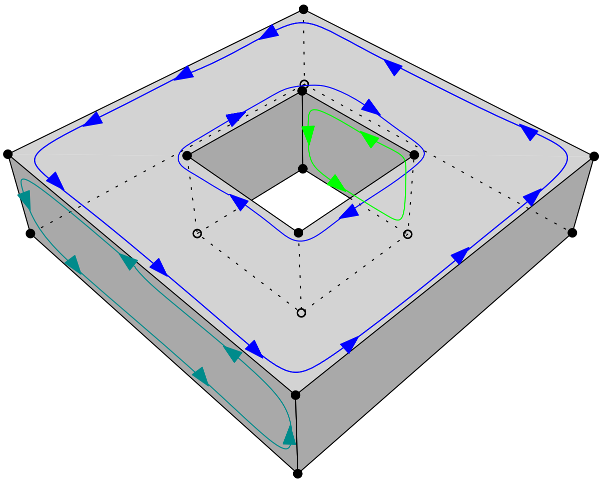


Figure 10: One solid and the orientation of 3 of its polygons (different colours).

basically gives a threshold that says: "if 2 points are closer then ϵ_3, then we assume that they are the same". This value should be defined by the user

do we enforce 1mm for instance? if yes, how? with metadata?

what about point-point and point-line tolerances for 2D validation?

4.4 Orientation requirement

For a polygon embedded in the 2D plane, the orientation of its exterior ring must be the opposite of that of its interior rings (eg clockwise versus counterclockwise).

If one polygon is used to construct a shell, the exterior ring must be oriented in such as way that when viewed from outside the shell the points are ordered counterclockwise. Figure 10 shows an example. In other words, the normal of the surface must point outwards if a right-hand system is used, ie when the ordering of points follows the direction of rotation of the curled fingers of the right hand, then the thumb points towards the outside. If the polygon has interior rings, then these have to be ordered clockwise.

If the polygon is part of a MultiSurface then there is no prescribed orientation, besides the fact that the outer ring must have a different orientation than the inner ring(s).

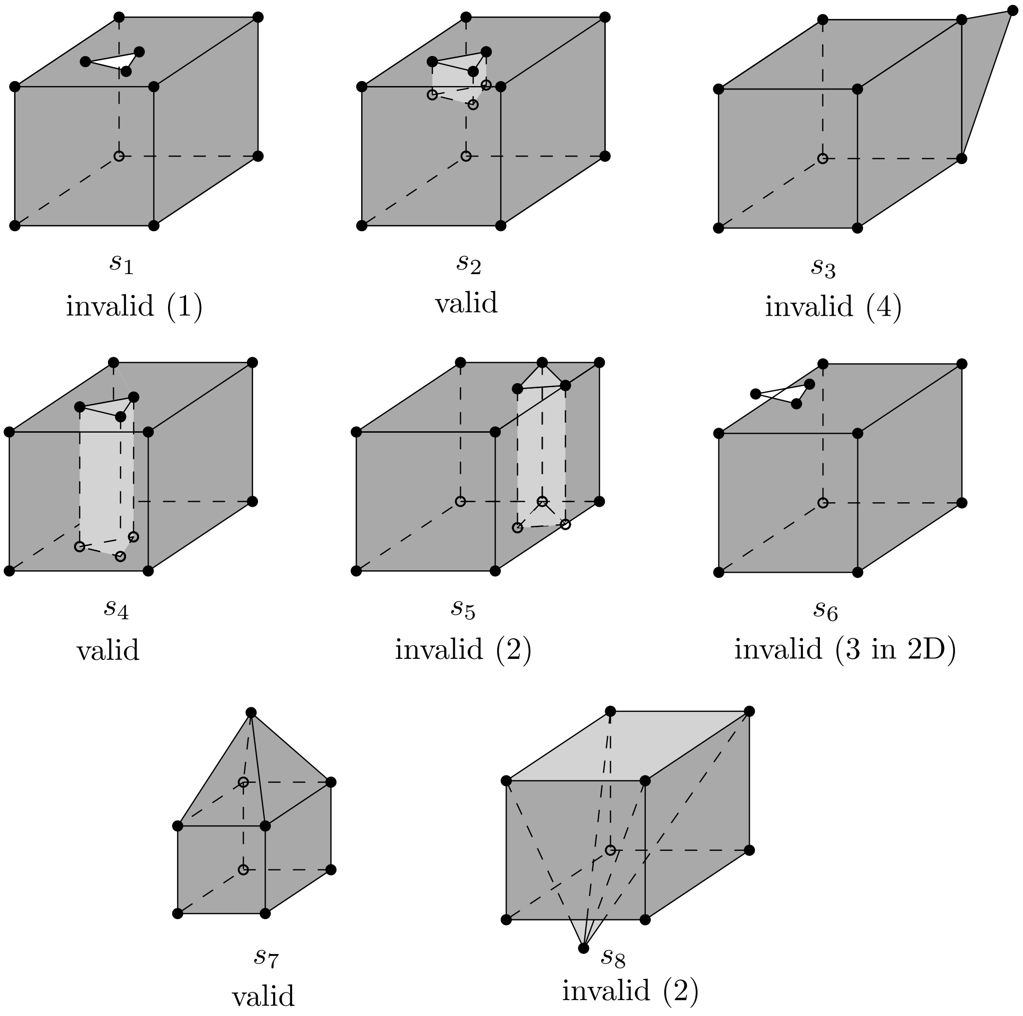


Figure 11: Nine solids, the number between brackets indicates which assertion(s) is/are violated (Figure 7).

4.5 Requirements for shells and solids

To understand the requirements for shells and solids, we can simply generalise the assertions in Figure 7: polygons become solids, rings become shells, and holes become cavities.

Figure 11 shows 9 solids, some of them valid some not.

The first assertion means that a solid must be closed, or 'watertight'. The solid s_1 is thus not valid but s_2 is since the hole in the top surface is 'filled' with other faces.

The second assertion implies that each shell must be simple, ie that it is a 2-manifold. s_5 and s_8 are thus invalid.

The fourth assertion states that a shell is a 2-manifold and that no dangling pieces can exist (such as that of s_3); it is equivalent to the *regularisation* of a point-set in $\mathbb{R}^3$.

The other assertions refer to solids having interior shells, which are out of scope for the QIE. These are thus ignored.

References

ISO (2003). ISO 19107:2003: Geographic information—Spatial schema. International Organization for Standardization.

OGC (2006). OpenGIS implementation specification for geographic information—simple feature access. Open Geospatial Consortium inc. Document 06-103r3.