The Bridge Model of the SIG 3D

an Application Domain Extension (ADE) for CityGML and a Proposal for a new Module in CityGML (CityGML Change Request CR 68)

prepared by the *Special Interests Group 3D (SIG 3D)* of the initiative *Spatial Data Infrastructure Germany (GDI.DE)*

March 2010

This document describes a model for the representation of bridges, which was developed by the SIG 3D (www.sig3d.de), particularly by Ulrich Gruber, in 2009. The model is implemented as an ADE for CityGML, and is a proposal to be included as new module in CityGML, initiated by the *CityGML Change Request CR68 - Thematic module for bridges (25 Feb 2010)*. This document contains a documentation of the bridge model (page 1), and detailed proposals which part of the CityGML specifications have to be supplemented (page 12). An UML diagram of the bridge model is depicted on page 13, and the XML schema files can be obtained from http://www.citygmlwiki.org/index.php/CityGML_Bridge_ADE. Contact persons for the Bridge Model are Ulrich Gruber (Ulrich.Gruber@kreis-re.de) and Gerhard Gröger (groeger@igg.uni-bonn.de).

1. Bridge model

The bridge model allows for the representation of the thematic, spatial and visual aspects of bridges, bridge parts and construction elements in four levels of detail. The model is defined by the thematic extension module *Bridge*. The bridge model was developed in analogy to the building model (c.f. chapter 10.3 of the CityGML specification, version 1.0). The UML diagram of the bridge model is depicted on page 13[1].

The abstract base class for all types of bridges (movable or unmovable) is the class *_AbstractBridge.* The sub classes *Bridge* and *BridgePart* inherit its (semantical and geometrical) properties and associations. An aggregation named *ConsistOfBridgePart* allows for an aggregation hierarchy between the classes *Bridge* and *BridgePart*: A Bridge consist of multiple *BridgeParts,* where the *BridgeParts* again may be composed of *BridgeParts*. This aggregation is necessary to represent a twin bridge for example. One *BridgePart* has to belong to exactly one *Bridge*.

```
<xs:complexType name="AbstractBridgeType" abstract="true">
<xs:annotation>
<xs:documentation>Type describing the thematic and geometric attributes and the associations of bridges. It is an abstract
type, only its subclasses Bridge and BridgePart can be instantiated. An _AbstractBridge may consist of BridgeParts, which are
again _AbstractBridges by inheritance. Thus an aggregation hierarchy between _AbstractBridges of arbitrary depth may be
specified. In such an hierarchy, top elements are Bridges, while all other elements are BridgeParts. Each element of such a
hierarchy may have all attributes and geometries of _AbstractBridges. It must, however, be assured that no inconsistencies
occur.
</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="core:AbstractSiteType">
                <xs:sequence>
                    <xs:annotation>
                        <xs:documentation> The name will be represented by gml:name (inherited from _GML) The lodXMultiSurface
must be used, if the geometry of a bridge is just a collection of surfaces bounding a solid, but not a topologically clean solid
boundary necessary for GML3 solid boundaries. </xs:documentation>
                    </xs:annotation>
                    <xs:element name="class" type="gml:CodeType" minOccurs="0"/>
                    <xs:element name="function" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="type" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="is_movable" type="xs:boolean" default="false" minOccurs="0"/>
                    <xs:element name="lod1Solid" type="gml:SolidPropertyType" minOccurs="0"/>
                    <xs:element name="lod1MultiSurface" type="gml:MultiSurfacePropertyType" minOccurs="0"/>
                    <xs:element name="lod1TerrainIntersection" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="lod2Solid" type="gml:SolidPropertyType" minOccurs="0"/>
                    <xs:element name="lod2MultiSurface" type="gml:MultiSurfacePropertyType" minOccurs="0"/>
                    <xs:element name="lod2MultiCurve" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="lod2TerrainIntersection" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="outerBridgeConstructionElement" type="BridgeConstructionElementPropertyType"
minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="outerBridgeInstallation" type="BridgeInstallationPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element name="interiorBridgeInstallation" type="IntBridgeInstallationPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element name="boundedBy" type="BoundarySurfacePropertyType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element name="lod3Solid" type="gml:SolidPropertyType" minOccurs="0"/>
                    <xs:element name="lod3MultiSurface" type="gml:MultiSurfacePropertyType" minOccurs="0"/>
                    <xs:element name="lod3MultiCurve" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="lod3TerrainIntersection" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="lod4Solid" type="gml:SolidPropertyType" minOccurs="0"/>
                    <xs:element name="lod4MultiSurface" type="gml:MultiSurfacePropertyType" minOccurs="0"/>
                    <xs:element name="lod4MultiCurve" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="lod4TerrainIntersection" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="interiorBridgeRoom" type="InteriorBridgeRoomPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
```

[1] Note that there are some small inconsistencies between the documentation, the UML diagram and the XML schema files regarding some attributes. This inconsistencies will be fixed as soon as possible.

```
                <xs:element name="consistsOfBridgePart" type="BridgePartPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
                <xs:element ref="_GenericApplicationPropertyOfAbstractBridge" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
  <!-- ====================================================================================== -->
   <xs:element name="_AbstractBridge" type="AbstractBridgeType" abstract="true" substitutionGroup="core:_Site"/>
  <!-- ====================================================================================== -->
```

The gml attribute *gml:name*, which is inherited from the base class *_GML*, is used to denote the name of the bridge. The geometry of a bridge or a bridge part is represented in each LoD by a solid. Alternatively, a *MultiSurface* can be used.

The semantic attributes of a bridge in the abstract class *_AbstractBridge* are: *class, function, type, usage* and *is_movable*. The attribute *class* is used to make a rough classification of bridges (e.g., traffic, supply, historical) . The type of this attribute is *BridgeClassType*, the values of which are defined by a Code List. The attribute *function* allows to represent the utilization of the bridge independently of the construction. The type of the attribute *function* is *BridgeFunctionType*, which again is defined by a Code List. Possible values may be railway bridge, roadway bridge, pedestrian bridge, aqueduct, etc. The option to denote a usage which is divergent to one of the primary functions of the bridge (*function*) is given by the attribute *usage*, the type of which is *BridgeUsageType*.

The attribute *type* denotes the construction of the bridge: Arch bridge, suspension bridge, swing-bridge etc. This declaration is in analogy to the attribute *roofType* in the *Building*-module. The type of that *attribute* is *BridgeClassType*. Figure 2 depicts examples for construction types of bridges.

```
   <xs:complexType name="BridgeType">
      <xs:complexContent>
         <xs:extension base="AbstractBridgeType">
            <xs:sequence>
               <xs:element ref="_GenericApplicationPropertyOfBridge" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
   <!-- ====================================================================================== --
>
   <xs:element name="Bridge" type="BridgeType" substitutionGroup="_AbstractBridge"/>
   <!-- ====================================================================================== --
>
   <xs:element name="_GenericApplicationPropertyOfBridge" type="xs:anyType" abstract="true"/>
   <!-- ====================================================================================== --
>
   <xs:complexType name="BridgePartType">
      <xs:complexContent>
         <xs:extension base="AbstractBridgeType">
            <xs:sequence>
               <xs:element ref="_GenericApplicationPropertyOfBridgePart" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
   <!-- ====================================================================================== --
>
   <xs:element name="BridgePart" type="BridgePartType" substitutionGroup="_AbstractBridge"/>
   <!-- ====================================================================================== --
>
   <xs:element name="_GenericApplicationPropertyOfBridgePart" type="xs:anyType" abstract="true"/>
   <!-- ====================================================================================== --
>
   <xs:complexType name="BridgePartPropertyType">
      <xs:annotation>
         <xs:documentation>Denotes the relation of an _AbstractBridge to its bridge parts. The BridgePartPropertyType
```

```
element must either carry a reference  to a BridgePart object or contain a BridgePart object inline, but neither both nor
none.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:restriction base="gml:AssociationType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="BridgePart"/>
                </xs:sequence>
            </xs:restriction>
        </xs:complexContent>
    </xs:complexType>
```

The Boolean attribute *is_movable* is defined to specify whether a bridge is movable or not. The modeling of the geometrical aspects of the movement is delayed to later versions. Some types of movable bridges are given in Figure 1.

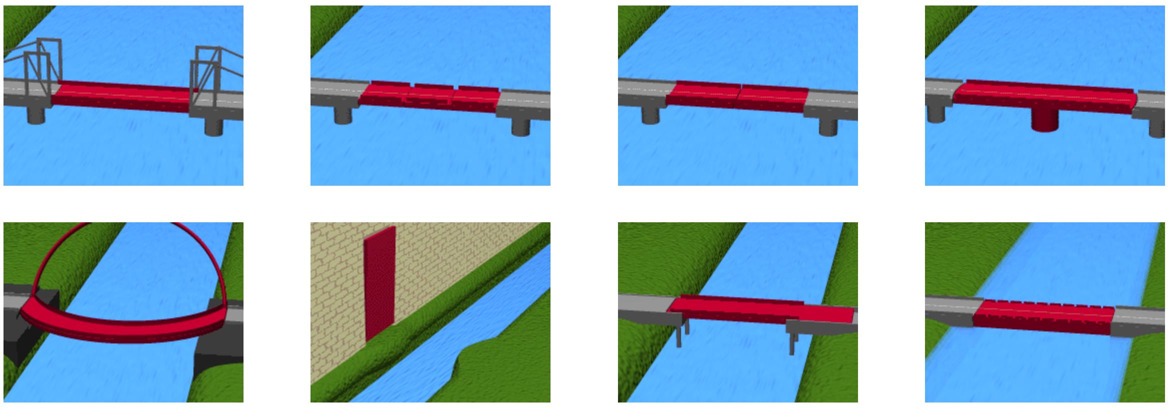
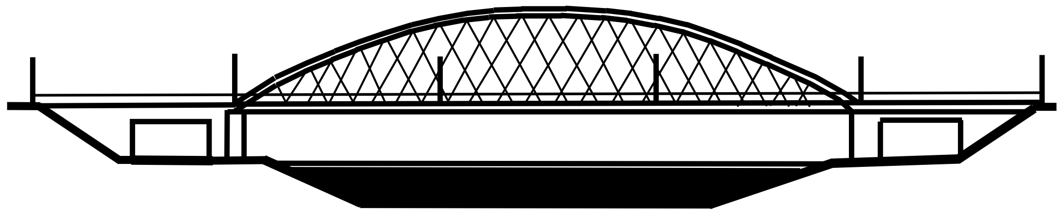
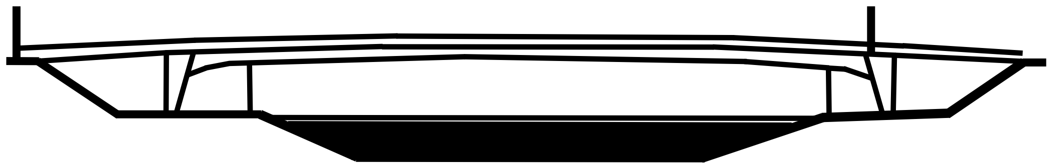
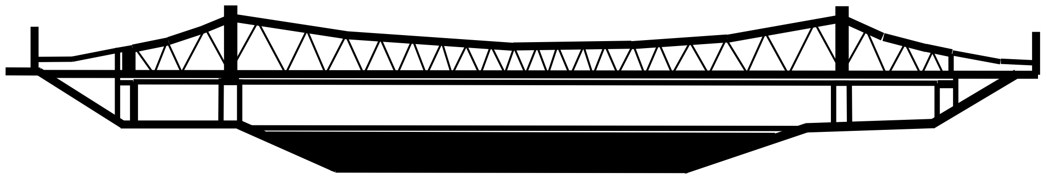


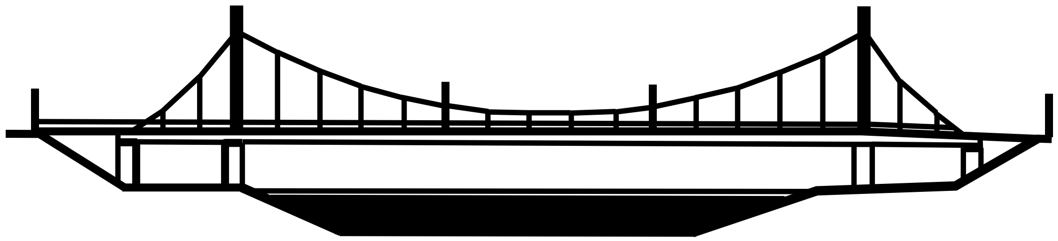
Figure 1: Types of movable bridges (Source: ISO 6707)

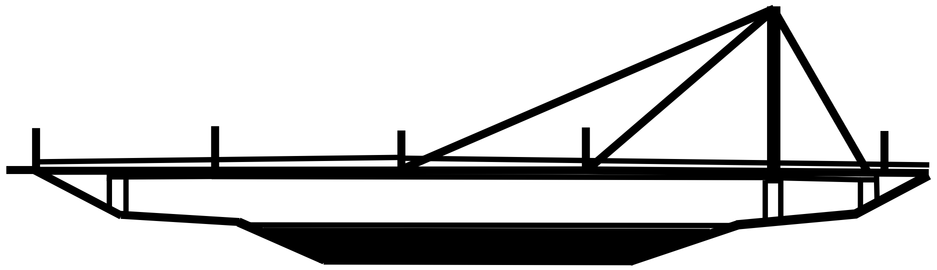
Figure 2: Examples of several types of bridges:

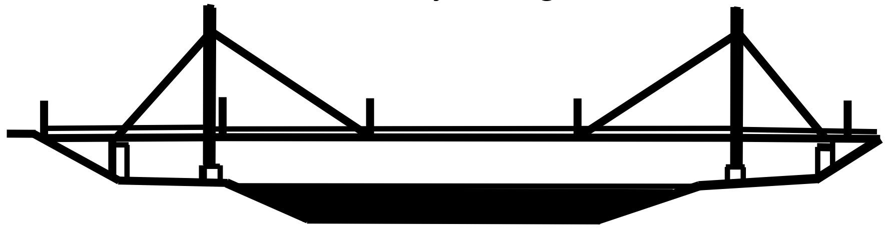

arced bridge

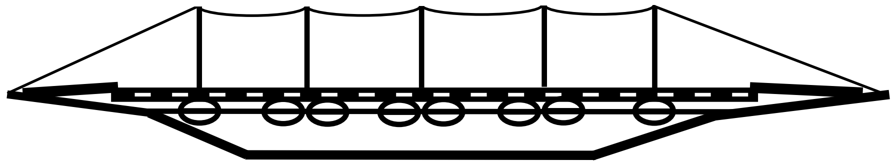

deck bridge


truss bridge


suspension bridge


cable-stayed bridge


cable-stayed overpass


pontoon bridge

BridgeConstructionElement

Bridge elements which are essential from a structural point of view are modeled as *BridgeConstructionElement*. Particularly structural elements like pylons, anchorages etc. are covered (see Figure 3 and Figure 4). The function of the construction element is represented by the attribute *function* with type *BridgeConstructionElementFunctionType*. The possible values are given as Code List, e.g., pylon, anchorage etc. The significant difference between *BridgeConstructionElement* and *BridgeInstallation* is that the first must have structural relevance. The geometry of a *BridgeConstructionElement* is *gml:_geometry;* it is very general to allow for linear characteristics.

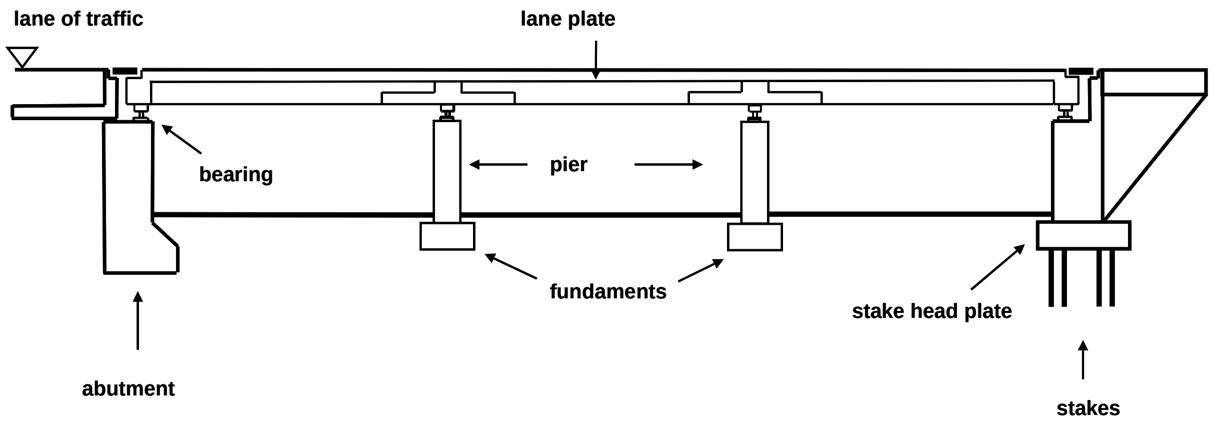


Figure 3: Construction elements of a deck bridge

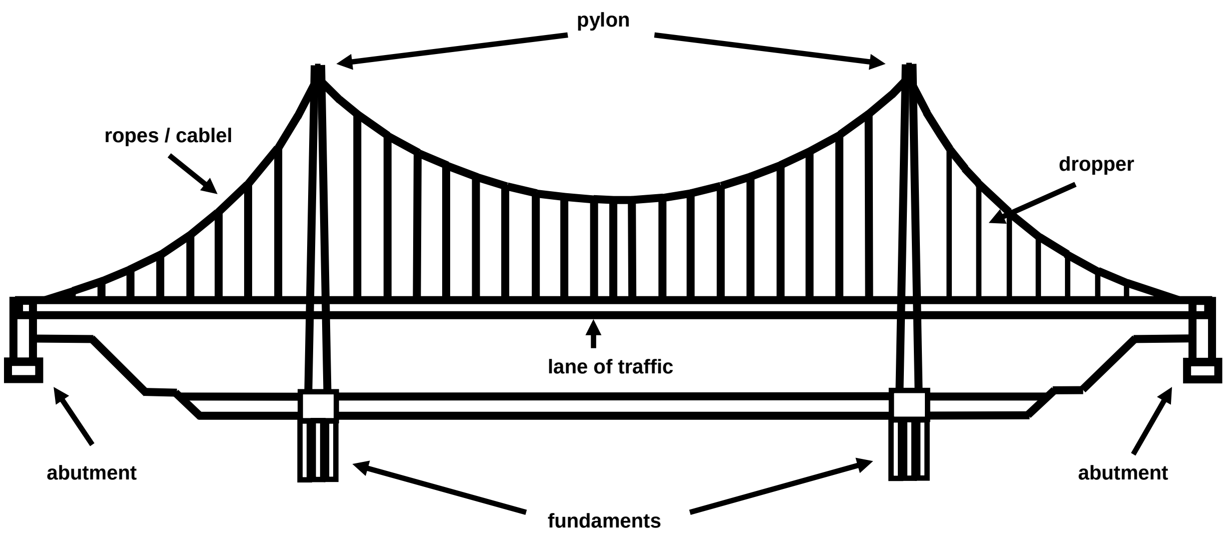


Figure 4: Construction elements of a suspension bridge

```
    <xs:complexType name="BridgeConstructionElementType">
        <xs:annotation>
            <xs:documentation>Type describing the thematic and geometric attributes and the associations of bridge construction elements.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="core:AbstractSiteType">
                <xs:sequence>
                    <xs:annotation>
                        <xs:documentation> The name will be represented by gml:name (inherited from _GML) The lodXMultiSurface must be used, if the geometry of a building is just a collection of surfaces bounding a solid, but not a topologically clean solid boundary necessary for GML3 solid boundaries. </xs:documentation>
                    <xs:annotation>
                    <xs:element name="class" type="gml:CodeType" minOccurs="0"/>
                    <xs:element name="function" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="lod1Geometry" type="gml:GeometryPropertyType" minOccurs="0"/>
                    <xs:element name="lod2Geometry" type="gml:GeometryPropertyType" minOccurs="0"/>
                    <xs:element name="lod3Geometry" type="gml:GeometryPropertyType" minOccurs="0"/>
                    <xs:element name="lod4Geometry" type="gml:GeometryPropertyType" minOccurs="0"/>
                    <xs:element name="lod1TerrainIntersection" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="lod2TerrainIntersection" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="lod3TerrainIntersection" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element name="lod4TerrainIntersection" type="gml:MultiCurvePropertyType" minOccurs="0"/>
                    <xs:element ref="_GenericApplicationPropertyOfBridgeConstructionElement" minOccurs="0"
    maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
   <!-- ===================================================================================== -->
    <xs:element name="BridgeConstructionElement" type="BridgeConstructionElementType" substitutionGroup="core:_Site"/>
   <!-- ===================================================================================== -->
    <xs:element name="_GenericApplicationPropertyOfBridgeConstructionElement" type="xs:anyType"/>
   <!-- ===================================================================================== -->
    <xs:complexType name="BridgeConstructionElementPropertyType">
        <xs:annotation>
            <xs:documentation>Denotes the relation of an _AbstractBridge to its bridge construction elements. The
    BridgeConstructionElementPropertyType element must either carry a reference to a BridgeConstructionElement object or
    contain a BridgeConstructionElement object inline, but neither both nor none.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:restriction base="gml:AssociationType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="BridgeConstructionElement"/>
                </xs:sequence>
            </xs:restriction>
        </xs:complexContent>
    </xs:complexType>
```

The instantiable class *BridgeInstallations* represents parts of the bridge which could be eliminated without collapsing of the bridge. It is about parts which are not relevant from a structural point of view. The *BridgeConstructionElements* in contrast are indispensable structurally. *BridgeInstallations* inherit its attributes and relations from the super class *_AbstractBridge*. *BridgeInstallations* occur in Levels of Detail 2 to 4 only; the LoD1 is semantically not detailed enough to contain *BridgeInstallations.* The type of geometry is very general (*gml:geometry)* and hence allows for solid, surface, linear or point geometries. Examples for the instances of the class *BridgeInstallations* are stairways, antennae, railing etc.

The class *BridgeInstallations* contains the semantic attributes *function* and *class*. The attribute *class* gives a rough classification of installations of a bridge, e.g. security, communication, or maintenance. Its type is *BridgeInstallationClassType*. The attribute *function* (type *BridgeInstallationFunctionType*) is more differentiated; values are stairs, antenna, signal light, or balustrade. The types of both attributes are *external codelists* according to the *dictionary concept of GML3.*

```
   <xs:complexType name="BridgeInstallationType">
  <!-- ===================================================================================== -->
      <xs:annotation>
         <xs:documentation>A BridgeInstallation is a part of a Bridge which has not the significance of a BridgePart. Examples
are stairs, etc. As subclass of _CityObject, a BridgeInstallation inherits all attributes and relations, in particular an id, names,
external references, generic attributes and generalization relations.</xs:documentation>
      </xs:annotation>
      <xs:complexContent>
         <xs:extension base="core:AbstractCityObjectType">
            <xs:sequence>
                <xs:element name="class" type="gml:CodeType" minOccurs="0"/>
                <xs:element name="function" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="lod2Geometry" type="gml:GeometryPropertyType" minOccurs="0"/>
                <xs:element name="lod3Geometry" type="gml:GeometryPropertyType" minOccurs="0"/>
                <xs:element name="lod4Geometry" type="gml:GeometryPropertyType" minOccurs="0"/>
                <xs:element ref="_GenericApplicationPropertyOfBridgeInstallation" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<!-- ========================================================================================== -->
<xs:element name="BridgeInstallation" type="BridgeInstallationType" substitutionGroup="core:_CityObject"/>
<!-- ========================================================================================== -->
<xs:element name="_GenericApplicationPropertyOfBridgeInstallation" type="xs:anyType" abstract="true"/>
<!-- ========================================================================================== -->
<xs:complexType name="BridgeInstallationPropertyType">
    <xs:annotation>
        <xs:documentation>Denotes the relation of an _AbstractBridge to its bridge installations. The
BridgePBridgeInstallationPropertyTypeartPropertyType element must either carry a reference to a BridgeInstallation object or
contain a BridgeInstallation object inline, but neither both nor none.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="gml:AssociationType">
            <xs:sequence minOccurs="0">
                <xs:element ref="BridgeInstallation"/>
            </xs:sequence>
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The class *IntBridgeInstallation* is modeled in analogy to the class *IntBuildingInstallation* of the building model and inherits its attributes and relations from the abstract class *_CityObject*. This class represents features inside a bridge; examples are stairways, railings and heaters. The class *IntBridgeInstallation* contains the semantic attributes *class*, *function* and *usage*, the values of which are defined in analogy to the values of the corresponding attributes of *IntBuildingInstallation*: *class* (type *IntBridgeInstallationClassType*) again gives a rough classification, whereas *function* (type *IntBridgeInstallationFunctionType)* is more detailed and *usage* (type *IntBridgeInstallationUsageType*) denotes a usage which differs from the function initially intended. These types again are defined by *external code lists* according to the instructions of the *dictionary concept of GML3.*

```
<xs:complexType name="IntBridgeInstallationType">
    <xs:annotation>
        <xs:documentation>An IntBridgeInstallation is an interior part of a Bridge which has a specific function or semantic
meaning. Examples are interior stairs, railings, radiators or pipes. As subclass of _CityObject, an IntBridgeInstallation inherits
all attributes and relations, in particular an id, names, external references, generic attributes and generalization relations.
</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="core:AbstractCityObjectType">
            <xs:sequence>
                <xs:element name="class" type="gml:CodeType" minOccurs="0"/>
                <xs:element name="function" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="usage" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="lod4Geometry" type="gml:GeometryPropertyType" minOccurs="0"/>
                <xs:element ref="_GenericApplicationPropertyOfIntBridgeInstallation" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<!-- ========================================================================================== -->
<xs:element name="IntBridgeInstallation" type="IntBridgeInstallationType" substitutionGroup="core:_CityObject"/>
<!-- ========================================================================================== -->
<xs:element name="_GenericApplicationPropertyOfIntBridgeInstallation" type="xs:anyType" abstract="true"/>
<!-- ========================================================================================== -->
<xs:complexType name="IntBridgeInstallationPropertyType">
    <xs:annotation>
        <xs:documentation>Denotes the relation of an _AbstractBridge to its interior bridge installations. The
IntBridgeInstallationPropertyType element
            must either carry a reference to a IntBridgeInstallation object or contain a IntBridgeInstallation object inline, but
neither both nor
            none.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:restriction base="gml:AssociationType">
            <xs:sequence minOccurs="0">
                <xs:element ref="IntBridgeInstallation"/>
            </xs:sequence>
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
```

The class *BridgeRoom* was designed in analogy to the class *Room* of the *Building* module. Hence, the types of its attributes *class, function* and *usage* (*BridgeRoomClassType, BridgeRoomFunctionType,* and *BridgeRoomUsageType)* are defined according to the building module as *external code lists* of the *dictionary concept of GML3.*

The picture in Figure 5 depicts an example for famous bridge with *BridgeRooms.*

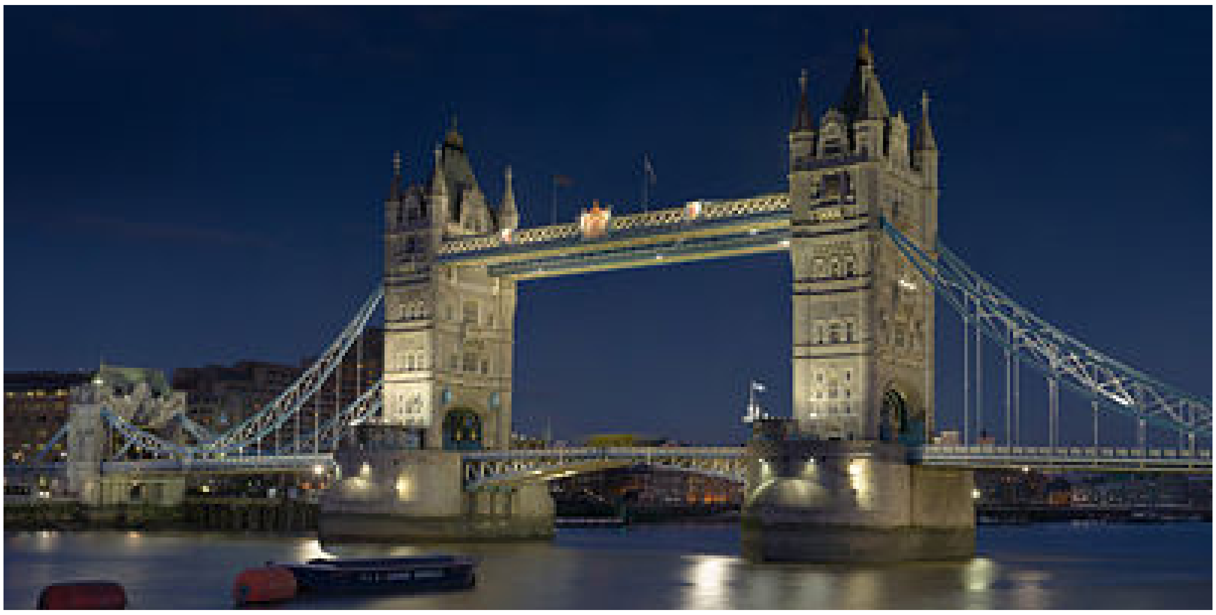


Figure 5: The Tower Bridge in London as example for a bridge with *BridgeRooms*.

```
    <xs:complexType name="BridgeRoomType">
        <xs:annotation>
            <xs:documentation>A Room is a thematic object for modelling the closed parts inside a Bridge. It has to be closed, if
necessary by using closure surfaces. The geometry may be either a solid, or a MultiSurface if the boundary is not topologically
clean. The room connectivity may be derived by detecting shared thematic openings or closure surfaces: two rooms are
connected if both use the same opening object or the same closure surface. The thematic surfaces bounding a room are
referenced by the boundedBy property. As subclass of _CityObject, a Room inherits all attributes and relations, in particular an
id, names, external references, generic attributes and generalization relations. </xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="core:AbstractCityObjectType">
                <xs:sequence>
                    <xs:element name="class" type="gml:CodeType" minOccurs="0"/>
                    <xs:element name="function" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="usage" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="lod4Solid" type="gml:SolidPropertyType" minOccurs="0"/>
                    <xs:element name="lod4MultiSurface" type="gml:MultiSurfacePropertyType" minOccurs="0"/>
                    <xs:element name="boundedBy" type="BoundarySurfacePropertyType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element name="interiorBridgeFurniture" type="InteriorBridgeFurniturePropertyType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element name="bridgeRoomInstallation" type="IntBridgeInstallationPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element ref="_GenericApplicationPropertyOfBridgeRoom" minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
```

```
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
 <!-- ============================================================================================ -->
  <xs:element name="BridgeRoom" type="BridgeRoomType" substitutionGroup="core:_CityObject"/>
 <!-- ============================================================================================ -->
```

The class *BridgeFurniture* is a subclass of the abstract class *_CityObject* and has the same design as the class *Furniture* of the *Building-Modul*. Again, the attributes *class*, *function* and *usage* (types *BridgeFurnitureClassType*, *BridgeFurnitureFunctionType*, *BridgeFurniture-UsageType*) are defined according to the instructions for *external code lists* of the *dictionary concept of GML3*.

```
  <xs:complexType name="BridgeFurnitureType">
    <xs:annotation>
      <xs:documentation>Type for bridge furnitures. As subclass of _CityObject, a BridgeFurniture inherits all attributes and relations, in particular an id, names, external references, generic attributes and generalization relations. </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="core:AbstractCityObjectType">
        <xs:sequence>
          <xs:element name="class" type="gml:CodeType" minOccurs="0"/>
          <xs:element name="function" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="usage" type="gml:CodeType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="lod4Geometry" type="gml:GeometryPropertyType" minOccurs="0"/>
          <xs:element name="lod4ImplicitRepresentation" type="core:ImplicitRepresentationPropertyType" minOccurs="0"/>
          <xs:element ref="_GenericApplicationPropertyOfBridgeFurniture" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
 <!-- ============================================================================================ -->
  <xs:element name="BridgeFurniture" type="BridgeFurnitureType" substitutionGroup="core:_CityObject"/>
 <!-- ============================================================================================ -->
```

Example

The Bridge of Rees was built as a composition of a *Bridge* feature and two additional *Bridge Parts*. Because the bridge model was developed in analogy to the building model, a bridge can be modeled as a composition of bridge parts. In the middle of Figure **6**, the *Bridge* feature is depicted, constituting the main part of the bridge. On the left and the right hand side of the *Bridge*, two *Bridge Parts* are represented, witch both complete the Bridge. In addition, there are some *Bridge Construction Elements*, which connect the *Bridge* and the *BridgeParts*. The *Bridge Construction Elements* are structurally essential. On the top of some of the Bridge Construction Elements, there are four lamps represented as *BridgeInstallations* (c.f. Figure **7**).

The CityGML file containing the Bridge of Rees can be obtained from http://www.citygmlwiki.org/index.php/CityGML_Bridge_ADE.

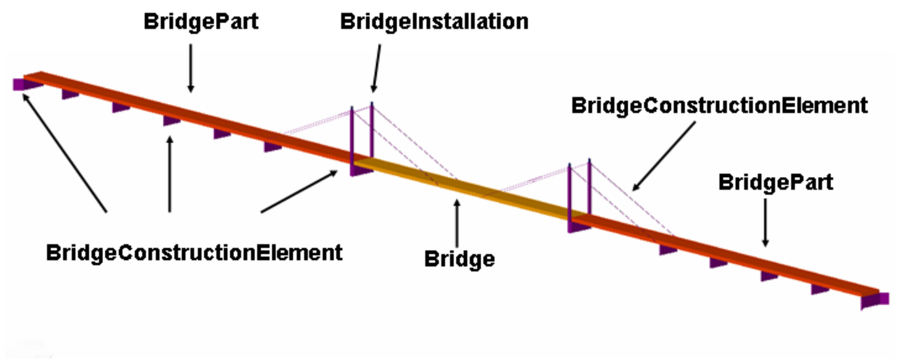


Figure 6: The Bridge of Rees, consisting of a *Bridge* feature and two *BridgePart* features

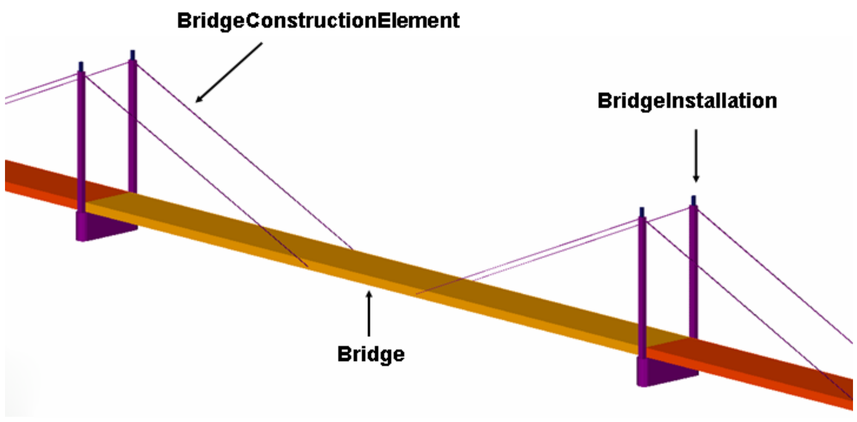


Figure 7: Lamps represented as *BridgeInstallations* on the top of *BridgeConstructionElements* of the Bridge of Rees

10.x.x Conformance requirements

Base requirements

1. If a bridge only consists of one (homogeneous) part, it shall be represented by the element *Bridge*. However, if a bridge is composed of individual structural segments, is shall be modeled as a *Bridge* element having one or more additional *BridgePart* elements. Only the geometry and non-spatial properties of the main part of the bridge should be represented within the aggregating *Bridge* element.

Referential integrity

2. An *_AbstractBridge* may consist of *BridgeParts*, which are again *_AbstractBridges* by inheritance. Thus an aggregation hierarchy between *_AbstractBridges* of arbitrary depth may be specified. In such an hierarchy, top elements are *Bridges*, while all other elements are *BridgeParts*. Each element of such a hierarchy may have all attributes and geometries of *_AbstractBridges*. It must, however, be assured that no inconsistencies occur.

Usage restriction of bridge model components according to different LODs

3. The *lodXSolid* and *lodXMultiSurface*, X ∊ [1..4], properties (*gml:SolidPropertyType* resp. *gml:MultiSurfacePropertyType*) of *_Abstract bridge* may be used to geometrically represent the exterior shell of

a bridge (as volume or surface model) within each LOD. For LOD1, either *lod1Solid* or *lod1MultiSurface* must be used, but not both. Starting from LOD2, both properties may be modeled individually.

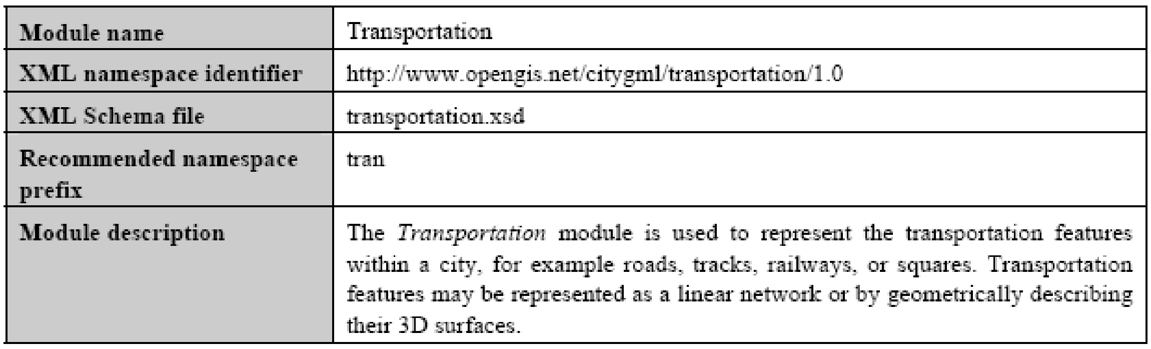
4. Starting from LOD2, the exterior shell of an *_Abstract bridge* may be semantically decomposed into *_BoundarySurface* elements using the *boundedBy* property (type: *BoundarySurfacePropertyType*) of *_AbstractBuilding*. Only *RoofSurface*, *WallSurface*, *GroundSurface*, and *ClosureSurface* as subclasses of *_BoundarySurface* are allowed. The *boundedBy* property shall not be used if the bridge is only represented in LOD1.

2. Locations where the CityGML specification shall be completed with regard to the bridge model

- Chapter 7 *Modularization*: Add the module 'Bridges' in the list of modules of the CityGML schema.

- Chapter 7.1 *CityGML core and extension modules*:

Example taken from the specification:



Module name	Transportation
XML namespace identifier	http://www.opengis.net/citygml/transportation/1.0
XML Schema file	transportation.xsd
Recommended namespace prefix	tran
Module description	The *Transportation* module is used to represent the transportation features within a city, for example roads, tracks, railways, or squares. Transportation features may be represented as a linear network or by geometrically describing their 3D surfaces.

Table for bridges to be added:

Modul name	Bridge
XML namespace identifier	http://www.opengis.net/citygml/bridge/1.1
XML Schema file	bridge.xsd
Recommended namespace prefix	brid
Module description	The *Bridge* module represents the thematic and spatial aspects of bridges, its aggregation in parts and construction elements in four level of detail.

- Chapter 7.2 *CityGML profiles*:

Pages 20-21:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.citygml.org/citygml/profiles/base/1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.citygml.org/citygml/profiles/base/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
<xs:import namespace="http://www.opengis.net/citygml/1.0"
schemaLocation="http://www.citygml.org/citygml/1.0/cityGMLBase.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/appearance/1.0"
schemaLocation="http://www.citygml.org/citygml/appearance/1.0/appearance.xsd"/>
```

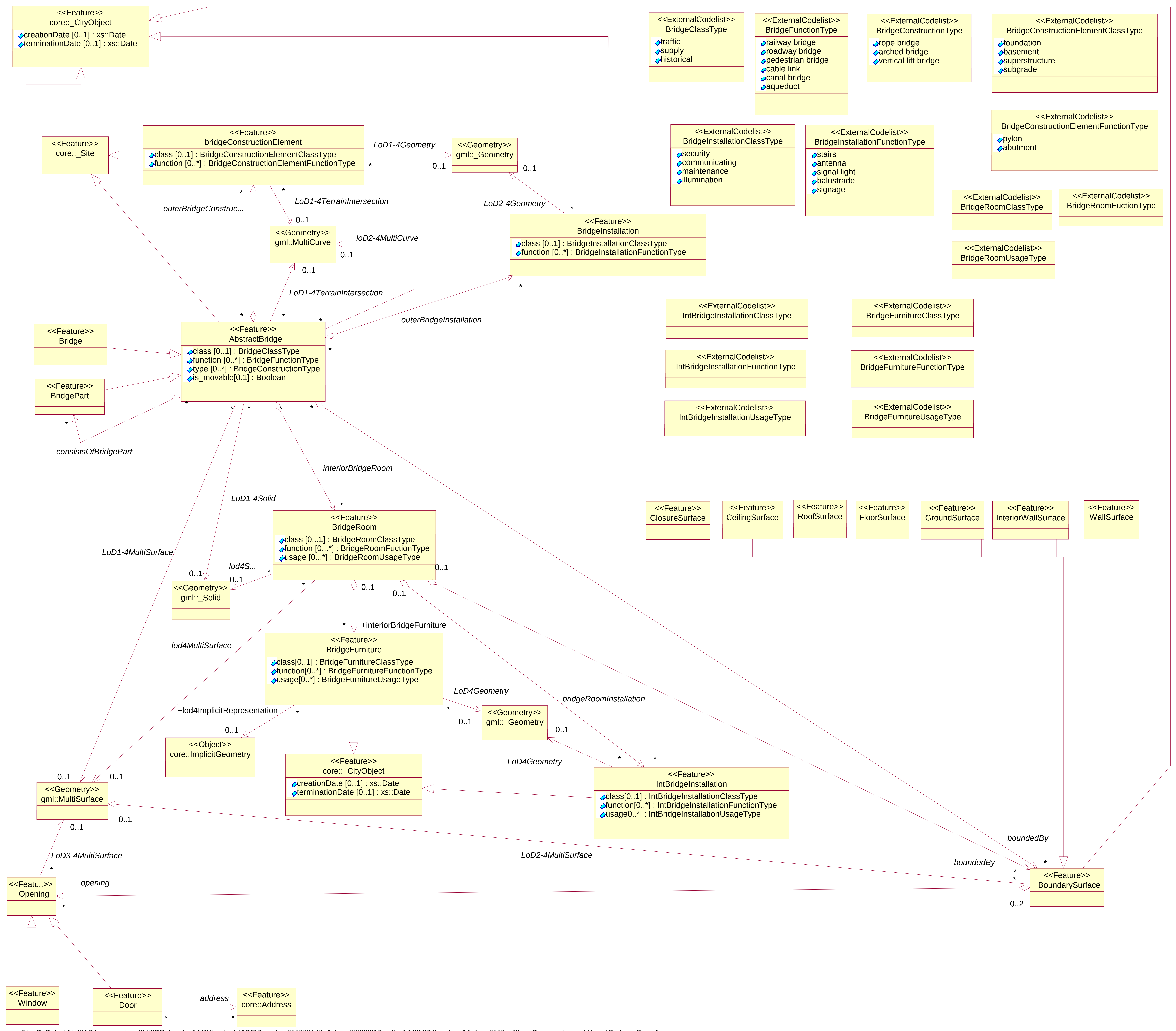
```
<xs:import namespace="http://www.opengis.net/citygml/building/1.0"
schemaLocation="http://www.citygml.org/citygml/building/1.0/building.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/cityfurniture/1.0"
schemaLocation="http://www.citygml.org/citygml/cityfurniture/1.0/cityFurniture.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/cityobjectgroup/1.0"
schemaLocation="http://www.citygml.org/citygml/cityobjectgroup/1.0/cityObjectGroup.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/genericcityobject/1.0"
schemaLocation="http://www.citygml.org/citygml/genericcityobject/1.0/genericCityObject.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/landuse/1.0"
schemaLocation="http://www.citygml.org/citygml/landuse/1.0/landUse.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/relief/1.0"
schemaLocation="http://www.citygml.org/citygml/relief/1.0/relief.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/transportation/1.0"
schemaLocation="http://www.citygml.org/citygml/transportation/1.0/transportation.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/vegetation/1.0"
schemaLocation="http://www.citygml.org/citygml/vegetation/1.0/vegetation.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/waterbody/1.0"
schemaLocation="http://www.citygml.org/citygml/waterbody/1.0/waterBody.xsd"/>
<xs:import namespace="http://www.opengis.net/citygml/texturedsurface/1.0"
schemaLocation="http://www.citygml.org/citygml/texturedsurface/1.0/texturedSurface.xsd"/>
NEW :
<xs:import namespace="http://www.opengis.net/citygml/bridge/1.1"
schemaLocation="http://www.citygml.org/citygml/ bridge /1.1/ bridge.xsd"/>
</xs:schema>

<xs:import namespace="http://www.opengis.net/citygml/subsurface/1.1"
schemaLocation="http://www.citygml.org/citygml/ bridge /1.1/ subsurface.xsd"/>
</xs:schema>
```

- Chapter 10 *Thematic model*
  addition (p.43):
  As for version 1.0 of CityGML, the following eleven thematic extension modules are defined: *Appearance*, *Building*, *CityFurniture*, *CityObjectGroup*, *GenericCityObject*, *LandUse*, *Relief*, *Transportation*, *Vegetation*, *WaterBody*, Bridge, SubSurface and *TexturedSurface [deprecated]*.

- Chapter 10.X (replace “x” by the actual section number): Add specification document for bridges (pages 1 to 11 of this document).

3. UML Diagram of the Bridge Model



File: D:\Daten\ALKIS\Pilotanwendung\3d\3DRuhrgebiet\AGStandards\ADE\Bruecken20090314\brücken_20090317.mdl 14:02:27 Sonntag, 14. Juni 2009 Class Diagram: Logical View / Bridges Page 1