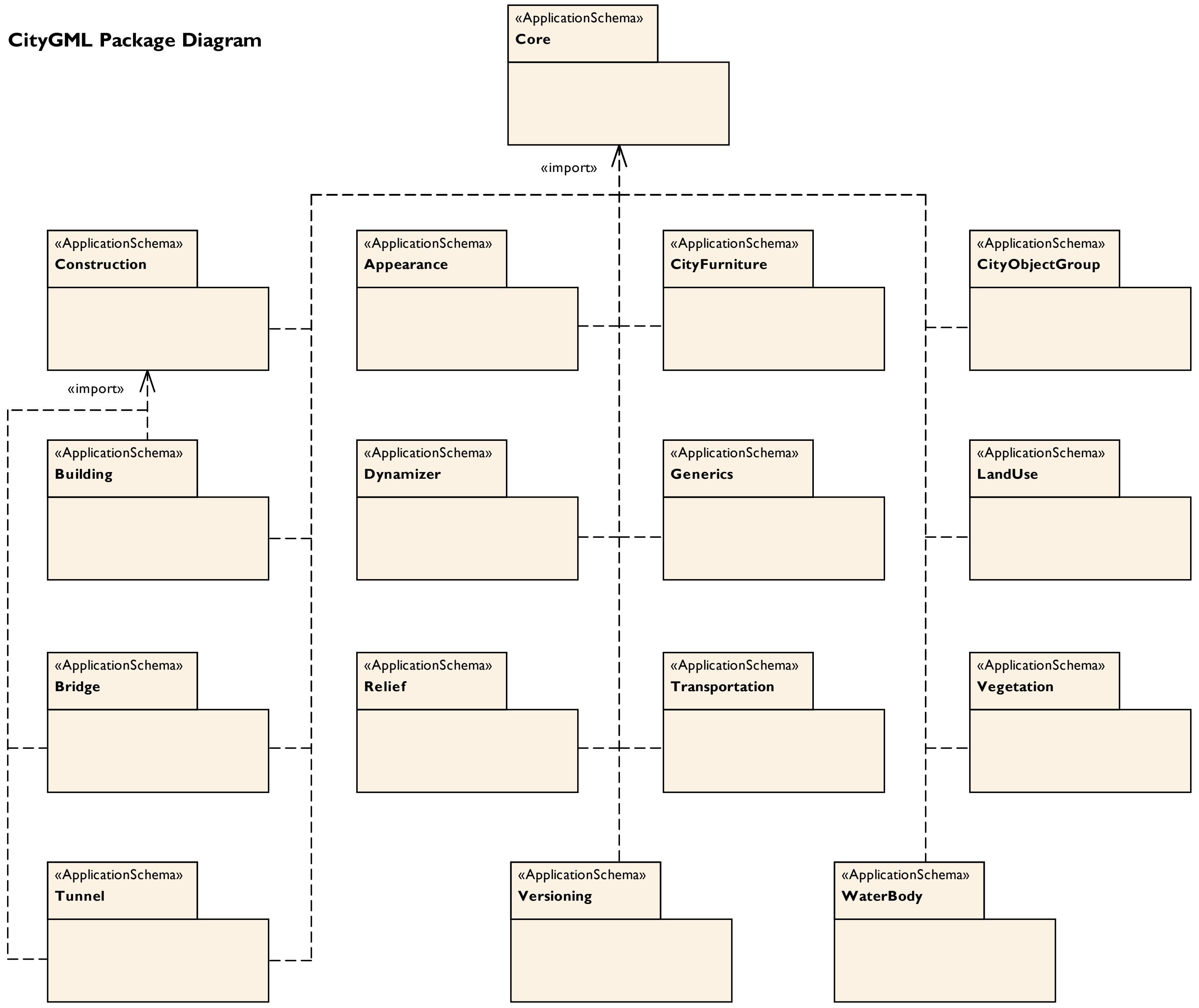
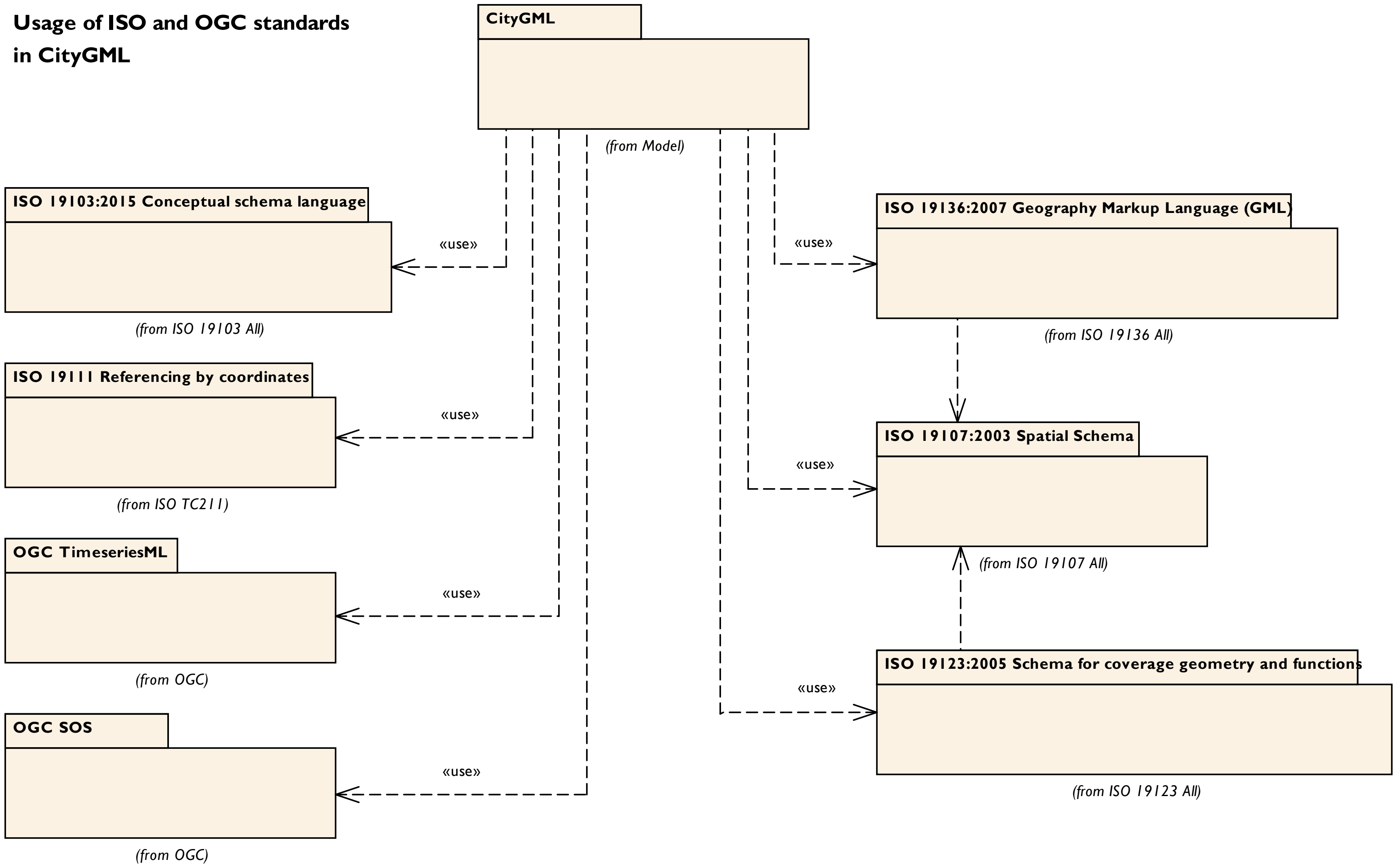
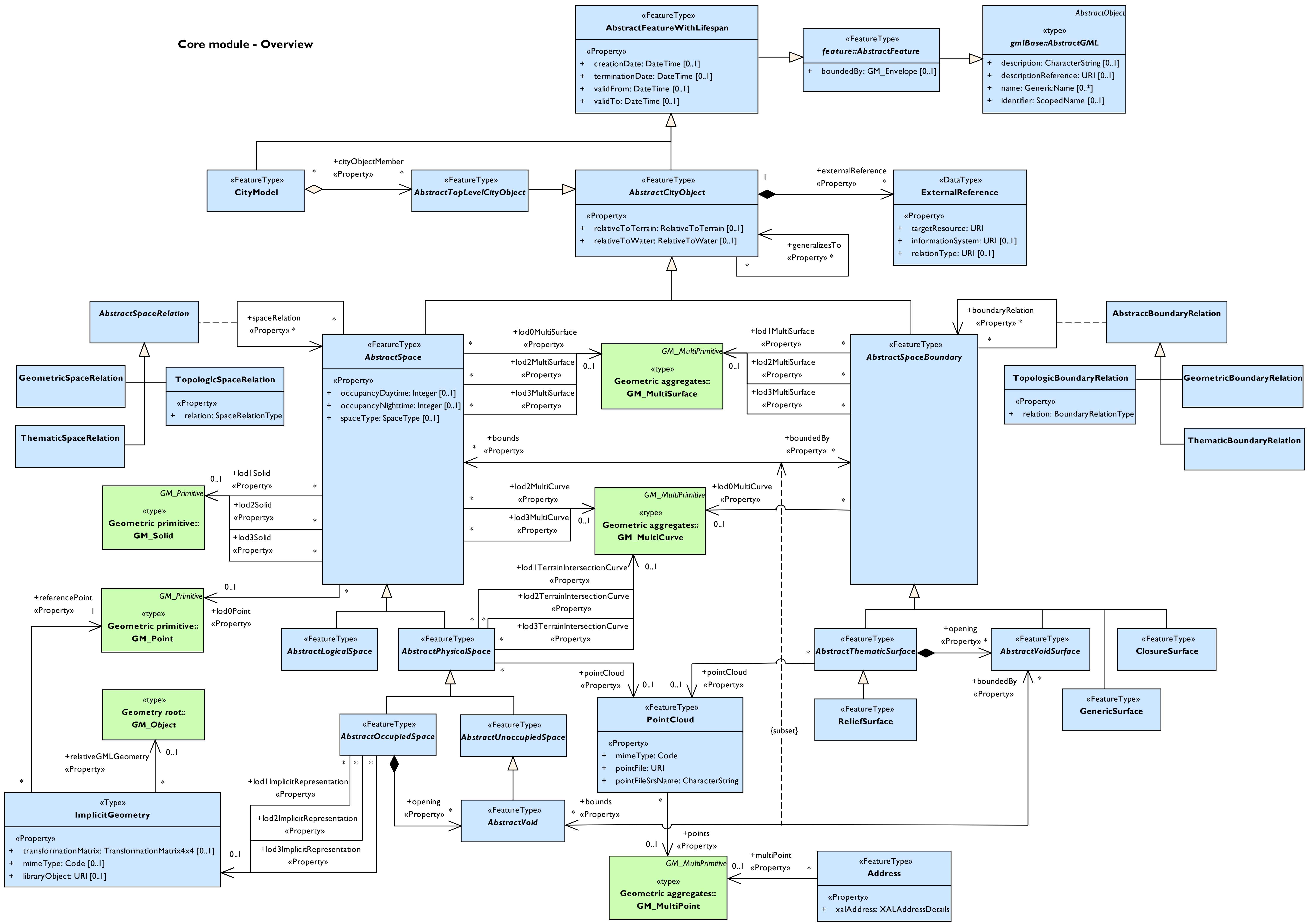
CityGML Package Diagram

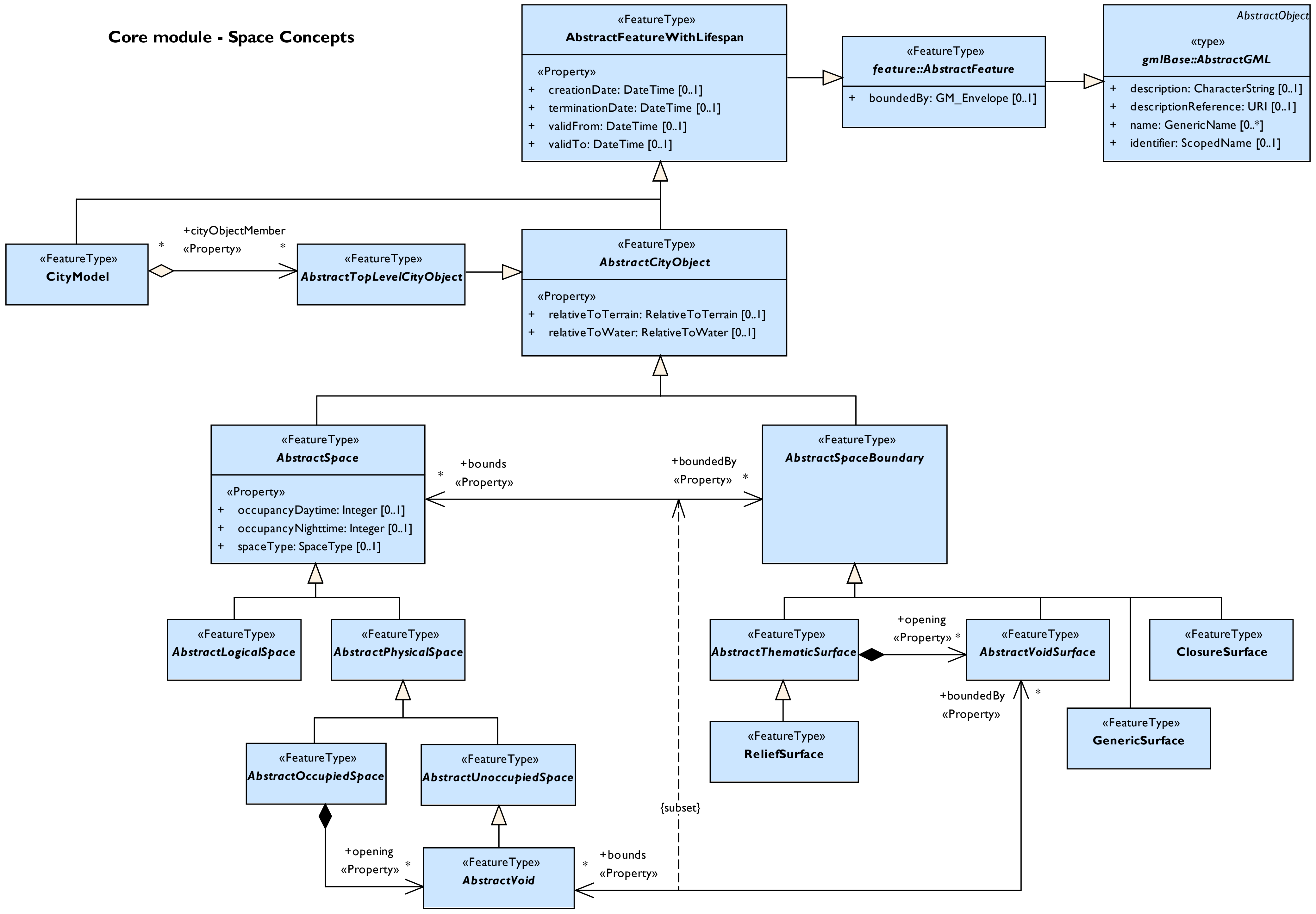



Usage of ISO and OGC standards in CityGML
CityGML
(from Model)
ISO 19103:2015 Conceptual schema language
(from ISO 19103 All)
ISO 19111 Referencing by coordinates
(from ISO TC211)
OGC TimeseriesML
(from OGC)
OGC SOS
(from OGC)
ISO 19136:2007 Geography Markup Language (GML)
(from ISO 19136 All)
ISO 19107:2003 Spatial Schema
(from ISO 19107 All)
ISO 19123:2005 Schema for coverage geometry and functions
(from ISO 19123 All)
«use»

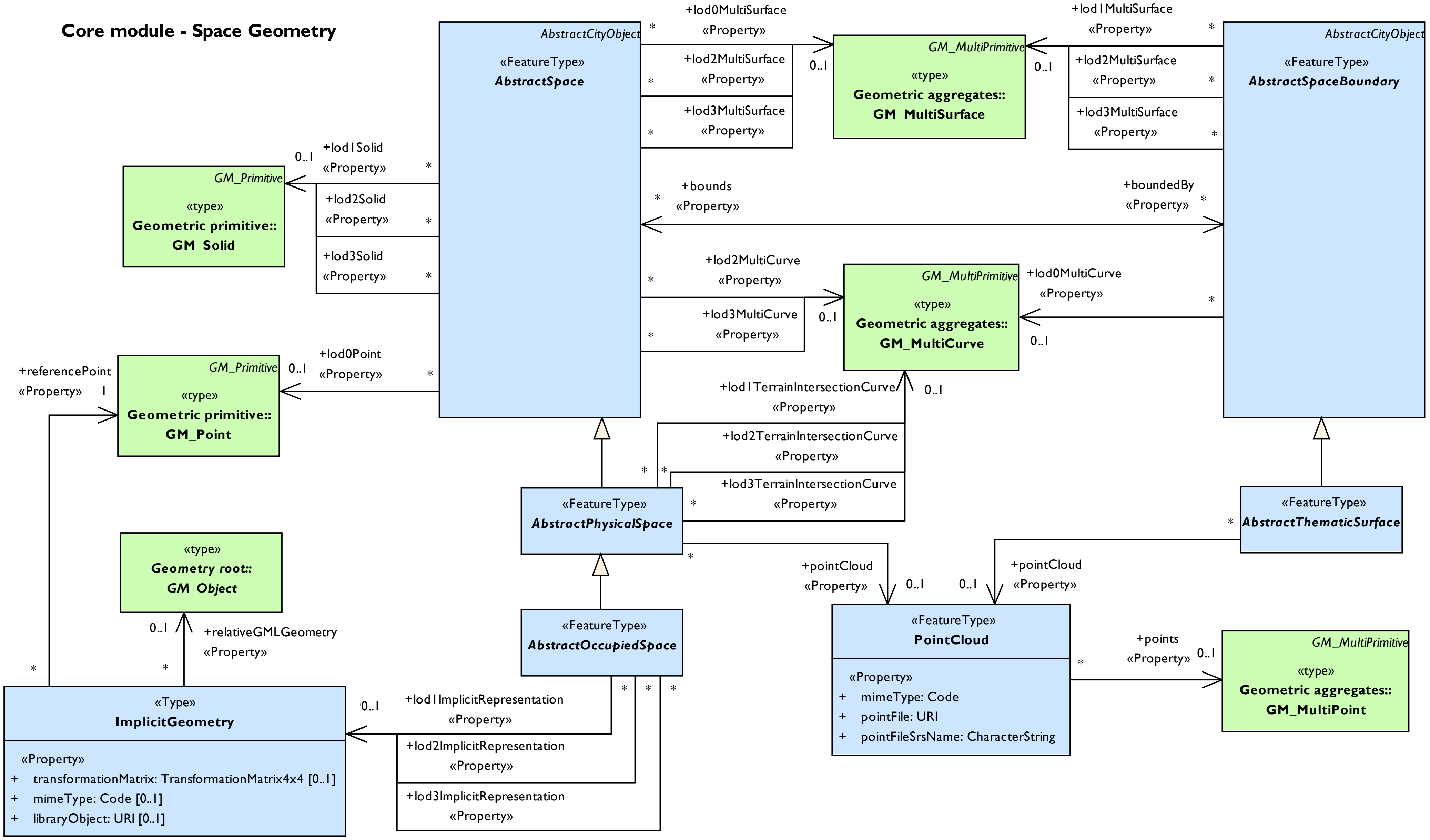
Core module - Overview



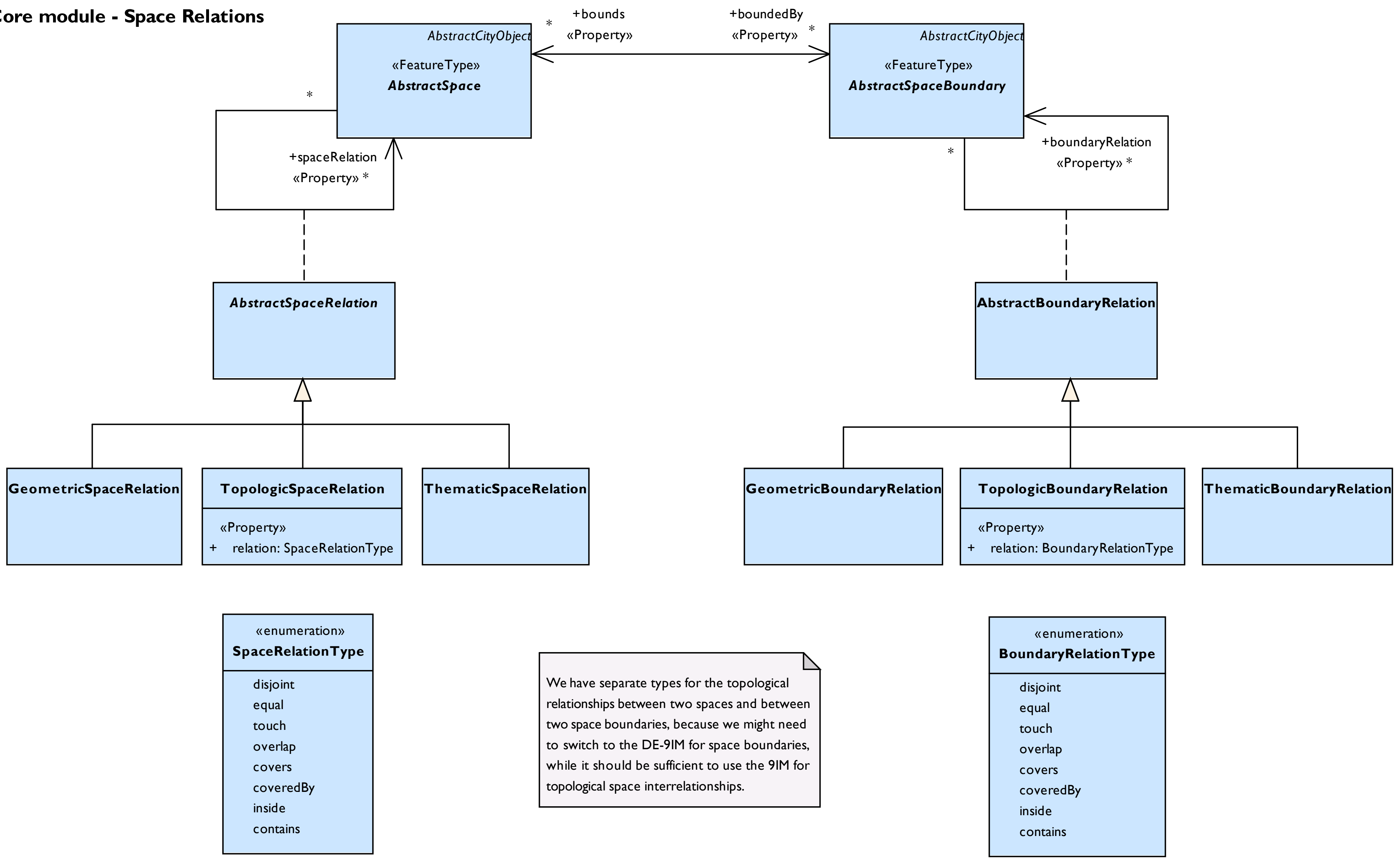
Core module - Space Concepts



Core module - Space Geometry

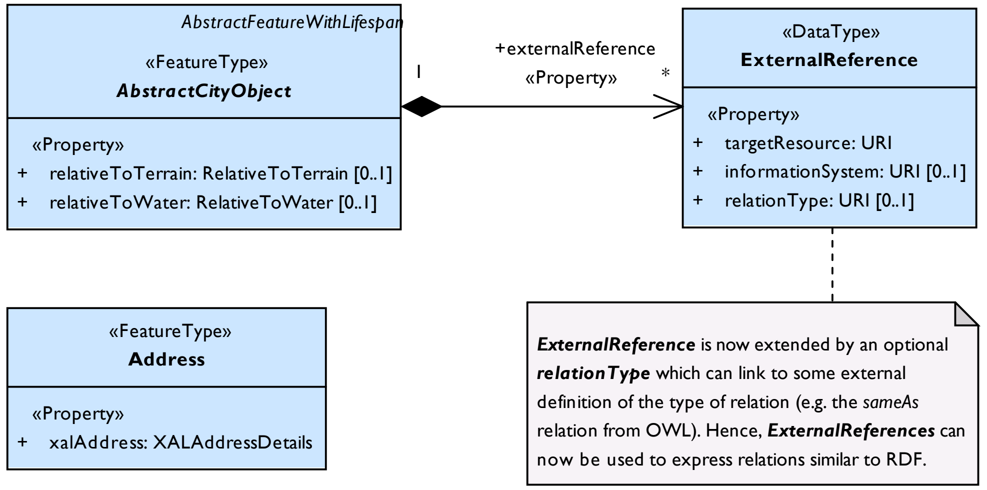

AbstractCityObject
«FeatureType»
AbstractSpace
+lod0MultiSurface «Property»
+lod2MultiSurface «Property»
+lod3MultiSurface «Property»
GM_MultiPrimitive
«type»
Geometric aggregates:: GM_MultiSurface
+lod1MultiSurface «Property»
AbstractCityObject
«FeatureType»
AbstractSpaceBoundary
+lod1Solid «Property»
+lod2Solid «Property»
+lod3Solid «Property»
GM_Primitive
«type»
Geometric primitive:: GM_Solid
+bounds «Property»
+boundedBy «Property»
+lod2MultiCurve «Property»
+lod3MultiCurve «Property»
GM_MultiPrimitive
«type»
Geometric aggregates:: GM_MultiCurve
+lod0MultiCurve «Property»
+referencePoint «Property»
GM_Primitive
«type»
Geometric primitive:: GM_Point
+lod0Point «Property»
+lod1TerrainIntersectionCurve «Property»
+lod2TerrainIntersectionCurve «Property»
+lod3TerrainIntersectionCurve «Property»
«FeatureType»
AbstractPhysicalSpace
«FeatureType»
AbstractThematicSurface
«type»
Geometry root:: GM_Object
+relativeGMLGeometry «Property»
+pointCloud «Property»
+pointCloud «Property»
«FeatureType»
AbstractOccupiedSpace
«FeatureType»
PointCloud
«Property»
+ mimeType: Code
+ pointFile: URI
+ pointFileSrsName: CharacterString
+points «Property»
GM_MultiPrimitive
«type»
Geometric aggregates:: GM_MultiPoint
«Type»
ImplicitGeometry
«Property»
+ transformationMatrix: TransformationMatrix4x4 [0..1]
+ mimeType: Code [0..1]
+ libraryObject: URI [0..1]
+lod1ImplicitRepresentation «Property»
+lod2ImplicitRepresentation «Property»
+lod3ImplicitRepresentation «Property»
0..1
*
1

Core module - Space Relations



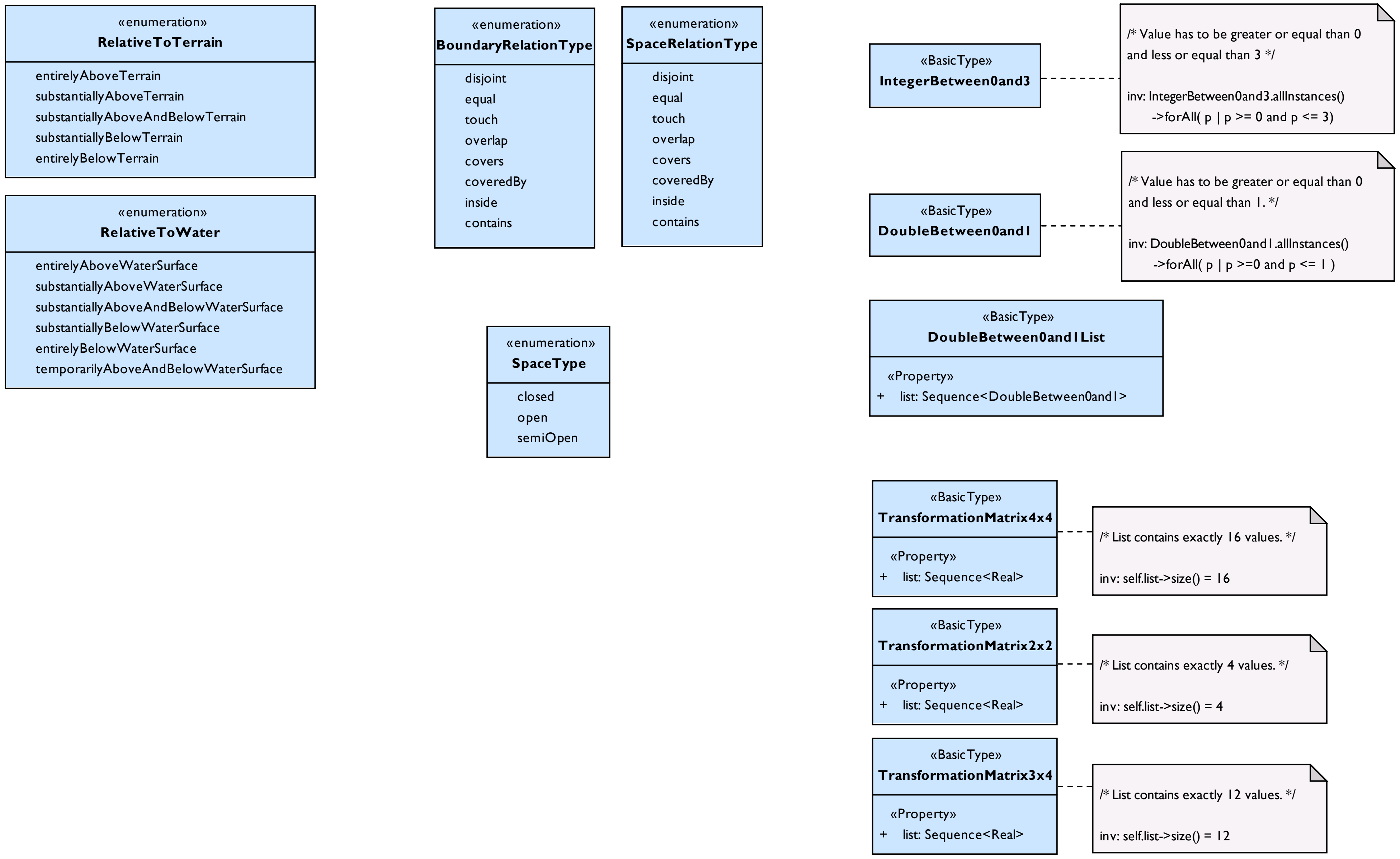
We have separate types for the topological relationships between two spaces and between two space boundaries, because we might need to switch to the DE-9IM for space boundaries, while it should be sufficient to use the 9IM for topological space interrelationships.

Core module - Miscellaneous

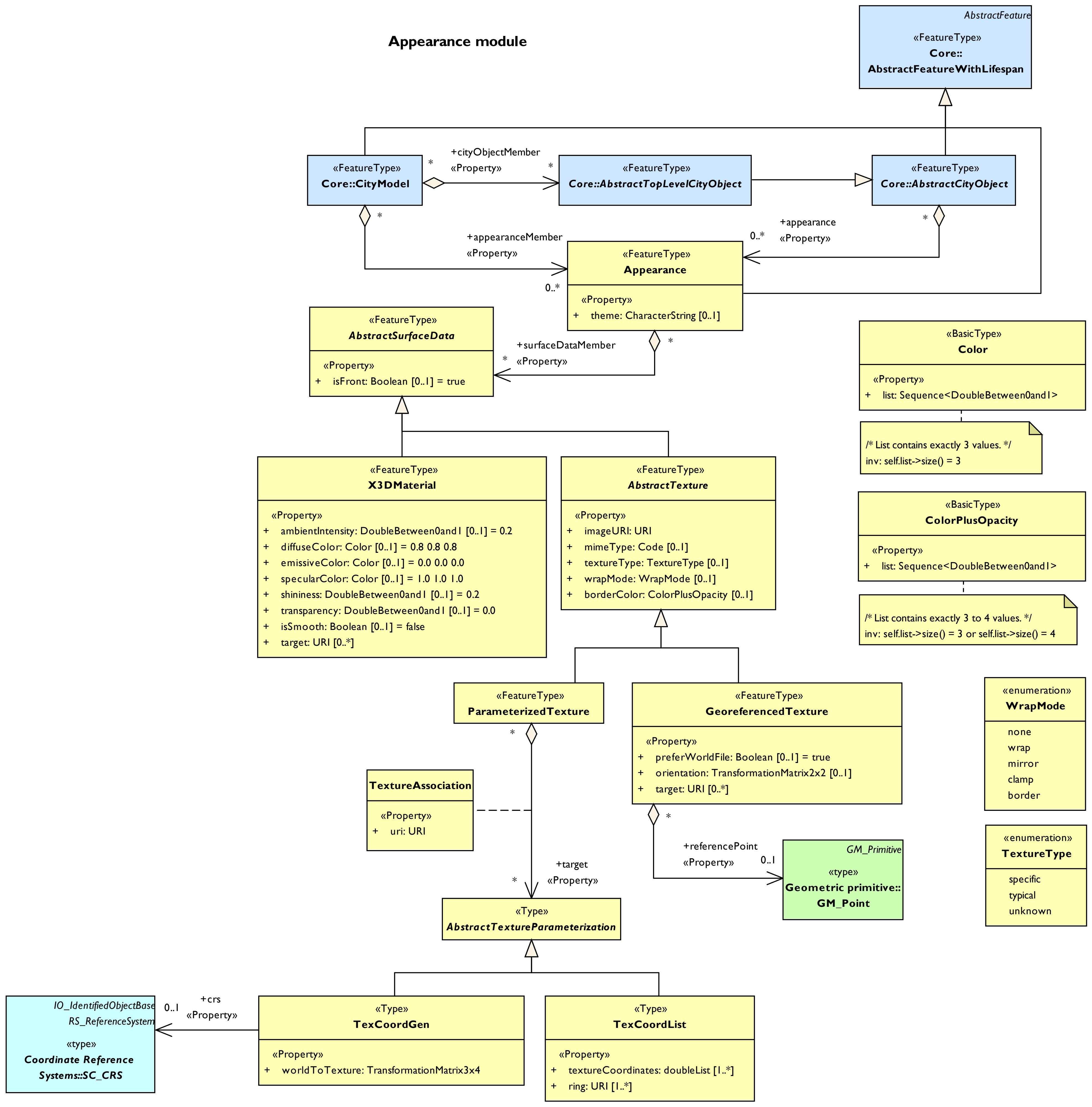

AbstractFeatureWithLifespan
«FeatureType»
AbstractCityObject
«Property»
+ relativeToTerrain: RelativeToTerrain [0..1]
+ relativeToWater: RelativeToWater [0..1]
1
+externalReference
«Property»
*
«DataType»
ExternalReference
«Property»
+ targetResource: URI
+ informationSystem: URI [0..1]
+ relationType: URI [0..1]
«FeatureType»
Address
«Property»
+ xalAddress: XALAddressDetails

ExternalReference is now extended by an optional ***relationType*** which can link to some external definition of the type of relation (e.g. the *sameAs* relation from OWL). Hence, ***ExternalReferences*** can now be used to express relations similar to RDF.

Core module - Basic Types and Enumerations

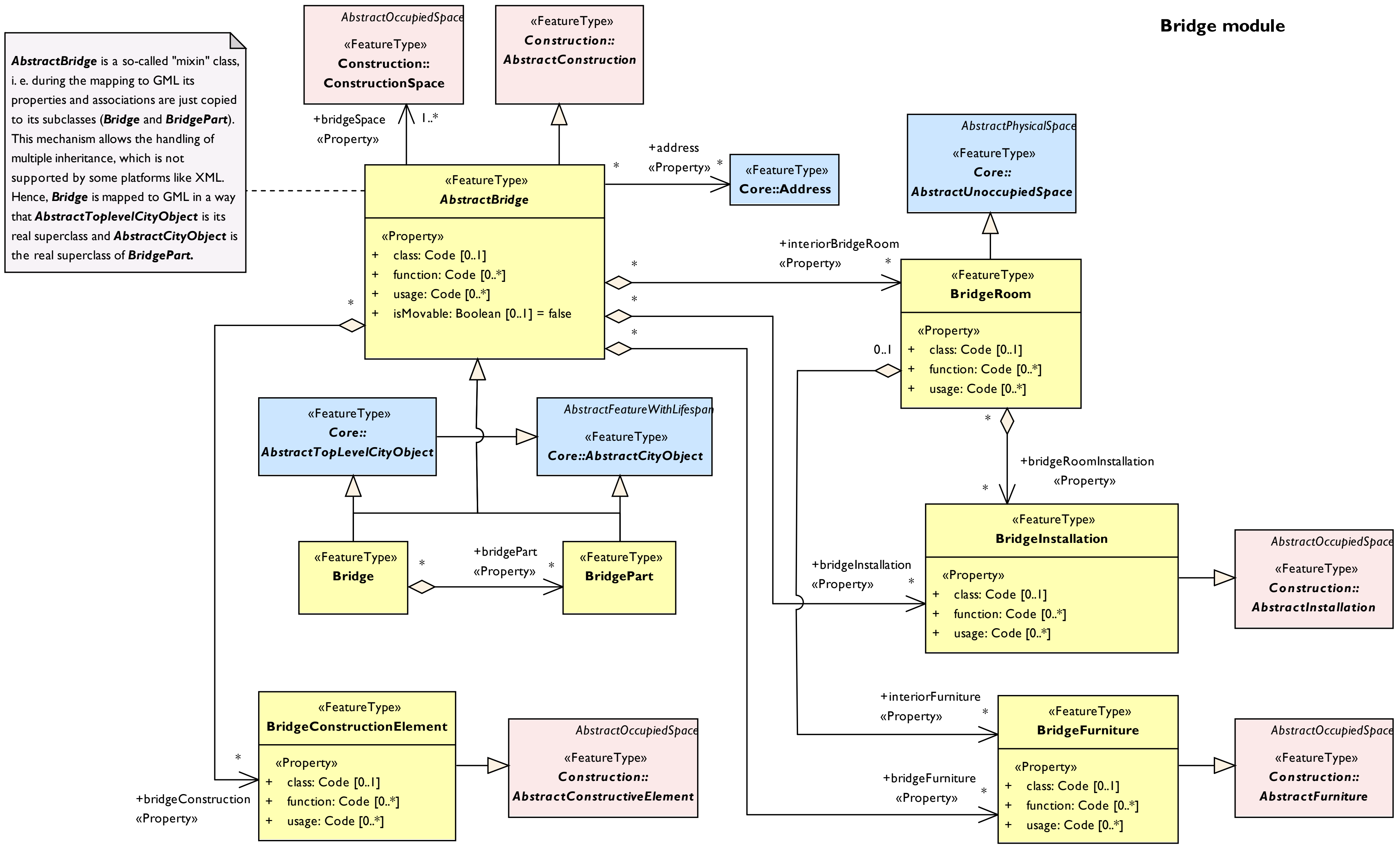

«enumeration»
RelativeToTerrain
entirelyAboveTerrain
substantiallyAboveTerrain
substantiallyAboveAndBelowTerrain
substantiallyBelowTerrain
entirelyBelowTerrain
«enumeration»
RelativeToWater
entirelyAboveWaterSurface
substantiallyAboveWaterSurface
substantiallyAboveAndBelowWaterSurface
substantiallyBelowWaterSurface
entirelyBelowWaterSurface
temporarilyAboveAndBelowWaterSurface
«enumeration»
BoundaryRelationType
disjoint
equal
touch
overlap
covers
coveredBy
inside
contains
«enumeration»
SpaceRelationType
disjoint
equal
touch
overlap
covers
coveredBy
inside
contains
«enumeration»
SpaceType
closed
open
semiOpen
«BasicType»
IntegerBetween0and3
/* Value has to be greater or equal than 0 and less or equal than 3 */
inv: IntegerBetween0and3.allInstances()
->forAll(p | p >= 0 and p <= 3)
«BasicType»
DoubleBetween0and1
/* Value has to be greater or equal than 0 and less or equal than 1. */
inv: DoubleBetween0and1.allInstances()
->forAll(p | p >=0 and p <= 1)
«BasicType»
DoubleBetween0and1List
«Property»
+ list: Sequence<DoubleBetween0and1>
«BasicType»
TransformationMatrix4x4
«Property»
+ list: Sequence<Real>
/* List contains exactly 16 values. */
inv: self.list->size() = 16
«BasicType»
TransformationMatrix2x2
«Property»
+ list: Sequence<Real>
/* List contains exactly 4 values. */
inv: self.list->size() = 4
«BasicType»
TransformationMatrix3x4
«Property»
+ list: Sequence<Real>
/* List contains exactly 12 values. */
inv: self.list->size() = 12

Appearance module

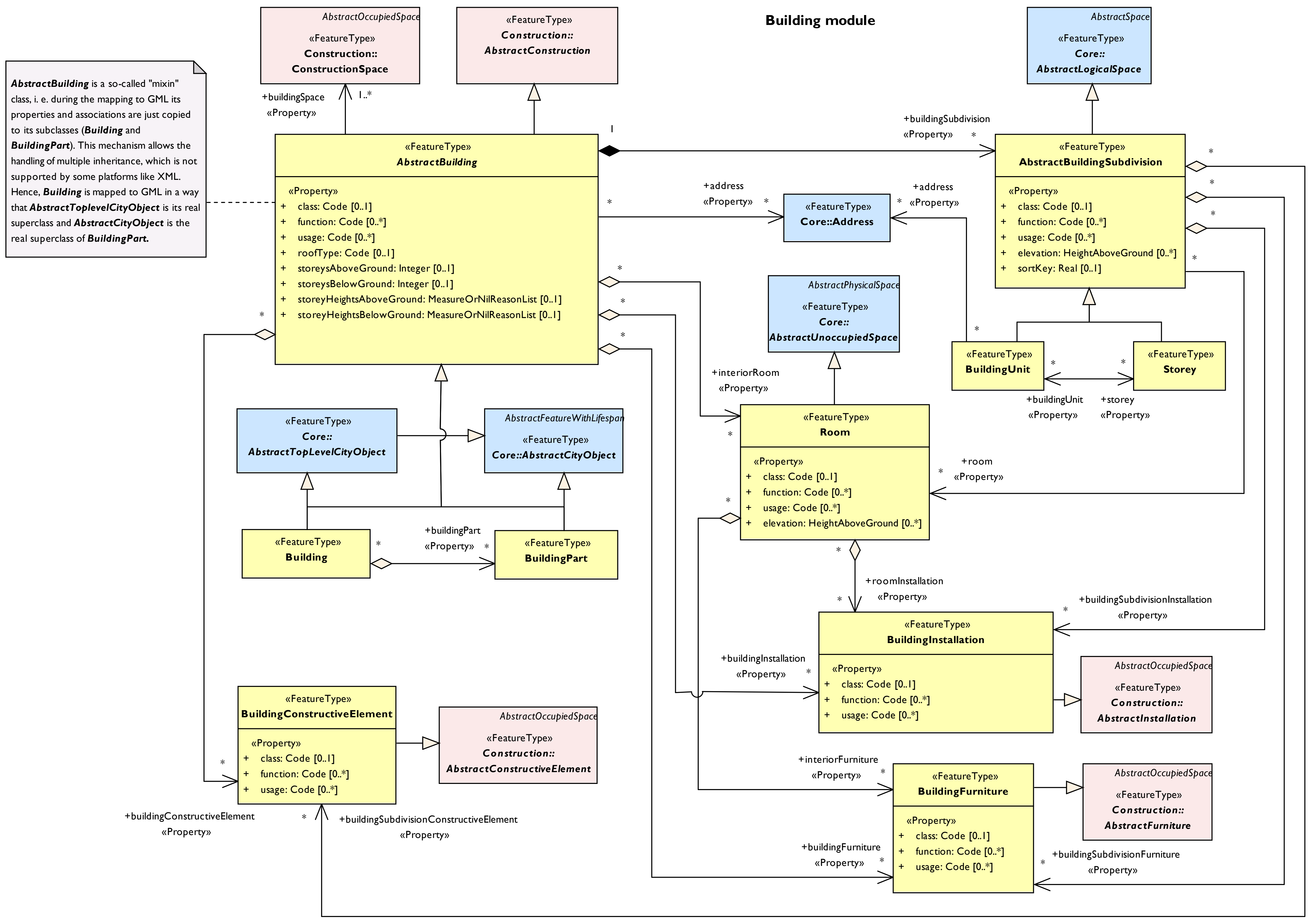


AbstractFeature
«FeatureType»
Core::AbstractFeatureWithLifespan
«FeatureType»
Core::CityModel
+cityObjectMember
«Property»
*
*
«FeatureType»
Core::AbstractTopLevelCityObject
«FeatureType»
Core::AbstractCityObject
*
+appearanceMember
«Property»
0..*
+appearance
«Property»
0..*
*
«FeatureType»
Appearance
«Property»
+ theme: CharacterString [0..1]
«FeatureType»
AbstractSurfaceData
«Property»
+ isFront: Boolean [0..1] = true
+surfaceDataMember
«Property»
*
*
«BasicType»
Color
«Property»
+ list: Sequence<DoubleBetween0and1>
/* List contains exactly 3 values. */
inv: self.list->size() = 3
«FeatureType»
X3DMaterial
«Property»
+ ambientIntensity: DoubleBetween0and1 [0..1] = 0.2
+ diffuseColor: Color [0..1] = 0.8 0.8 0.8
+ emissiveColor: Color [0..1] = 0.0 0.0 0.0
+ specularColor: Color [0..1] = 1.0 1.0 1.0
+ shininess: DoubleBetween0and1 [0..1] = 0.2
+ transparency: DoubleBetween0and1 [0..1] = 0.0
+ isSmooth: Boolean [0..1] = false
+ target: URI [0..*]
«FeatureType»
AbstractTexture
«Property»
+ imageURI: URI
+ mimeType: Code [0..1]
+ textureType: TextureType [0..1]
+ wrapMode: WrapMode [0..1]
+ borderColor: ColorPlusOpacity [0..1]
«BasicType»
ColorPlusOpacity
«Property»
+ list: Sequence<DoubleBetween0and1>
/* List contains exactly 3 to 4 values. */
inv: self.list->size() = 3 or self.list->size() = 4
«FeatureType»
ParameterizedTexture
*
«FeatureType»
GeoreferencedTexture
«Property»
+ preferWorldFile: Boolean [0..1] = true
+ orientation: TransformationMatrix2x2 [0..1]
+ target: URI [0..*]
«enumeration»
WrapMode
none
wrap
mirror
clamp
border
TextureAssociation
«Property»
+ uri: URI
*
+referencePoint
«Property»
0..1
GM_Primitive
«type»
Geometric primitive::GM_Point
«enumeration»
TextureType
specific
typical
unknown
+target
«Property»
*
«Type»
AbstractTextureParameterization
IO_IdentifiedObjectBase
RS_ReferenceSystem
«type»
Coordinate Reference Systems::SC_CRS
0..1
+crs
«Property»
«Type»
TexCoordGen
«Property»
+ worldToTexture: TransformationMatrix3x4
«Type»
TexCoordList
«Property»
+ textureCoordinates: doubleList [1..*]
+ ring: URI [1..*]

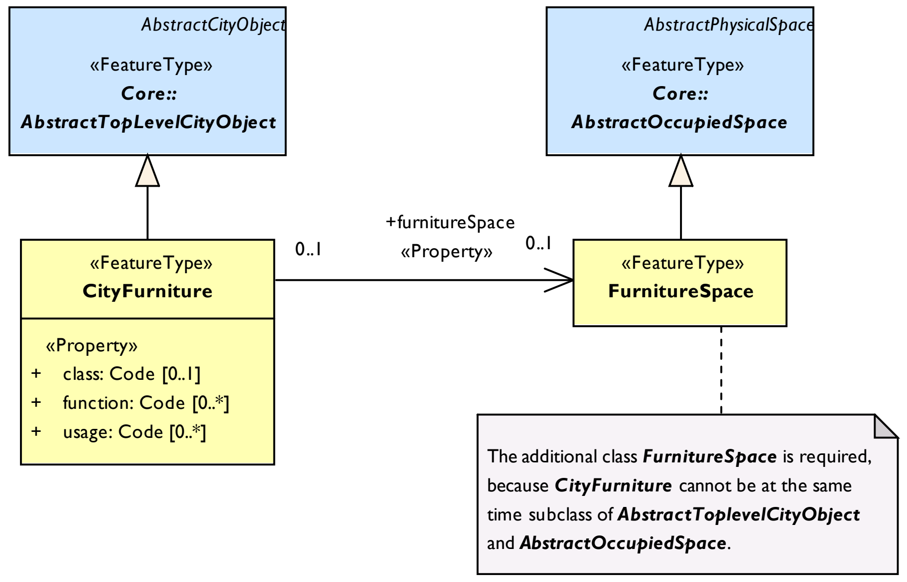
Bridge module


AbstractBridge is a so-called "mixin" class, i. e. during the mapping to GML its properties and associations are just copied to its subclasses (***Bridge*** and ***BridgePart***). This mechanism allows the handling of multiple inheritance, which is not supported by some platforms like XML. Hence, ***Bridge*** is mapped to GML in a way that ***AbstractToplevelCityObject*** is its real superclass and ***AbstractCityObject*** is the real superclass of ***BridgePart.***

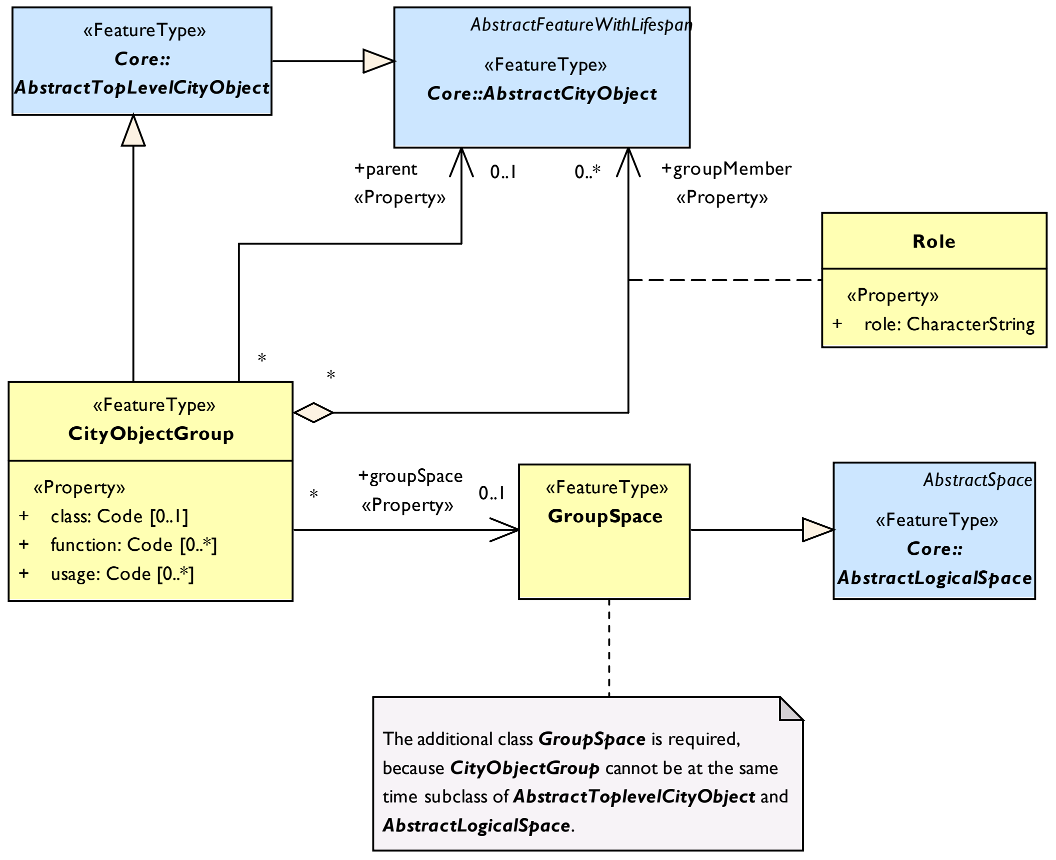
Building module


AbstractBuilding is a so-called "mixin" class, i. e. during the mapping to GML its properties and associations are just copied to its subclasses (Building and BuildingPart). This mechanism allows the handling of multiple inheritance, which is not supported by some platforms like XML. Hence, Building is mapped to GML in a way that AbstractToplevelCityObject is its real superclass and AbstractCityObject is the real superclass of BuildingPart.
AbstractOccupiedSpace «FeatureType» Construction:: ConstructionSpace
«FeatureType» Construction:: AbstractConstruction
AbstractSpace «FeatureType» Core:: AbstractLogicalSpace
+buildingSpace «Property» 1..*
1
+buildingSubdivision «Property» *
«FeatureType» AbstractBuilding
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
+ roofType: Code [0..1]
+ storeysAboveGround: Integer [0..1]
+ storeysBelowGround: Integer [0..1]
+ storeyHeightsAboveGround: MeasureOrNilReasonList [0..1]
+ storeyHeightsBelowGround: MeasureOrNilReasonList [0..1]
«FeatureType» AbstractBuildingSubdivision
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
+ elevation: HeightAboveGround [0..*]
+ sortKey: Real [0..1]
+address «Property» *
«FeatureType» Core::Address
+address «Property» *
AbstractPhysicalSpace «FeatureType» Core:: AbstractUnoccupiedSpace
«FeatureType» BuildingUnit
«FeatureType» Storey
+buildingUnit «Property» *
+storey «Property» *
+interiorRoom «Property» *
«FeatureType» Room
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
+ elevation: HeightAboveGround [0..*]
+room «Property» *
«FeatureType» Core:: AbstractTopLevelCityObject
AbstractFeatureWithLifespan «FeatureType» Core::AbstractCityObject
«FeatureType» Building
«FeatureType» BuildingPart
+buildingPart «Property» *
+roomInstallation «Property» *
+buildingSubdivisionInstallation «Property» *
«FeatureType» BuildingInstallation
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
+buildingInstallation «Property» *
AbstractOccupiedSpace «FeatureType» Construction:: AbstractInstallation
«FeatureType» BuildingConstructiveElement
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
AbstractOccupiedSpace «FeatureType» Construction:: AbstractConstructiveElement
+interiorFurniture «Property» *
«FeatureType» BuildingFurniture
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
AbstractOccupiedSpace «FeatureType» Construction:: AbstractFurniture
+buildingConstructiveElement «Property» *
+buildingSubdivisionConstructiveElement «Property» *
+buildingFurniture «Property» *
+buildingSubdivisionFurniture «Property» *

CityFurniture module

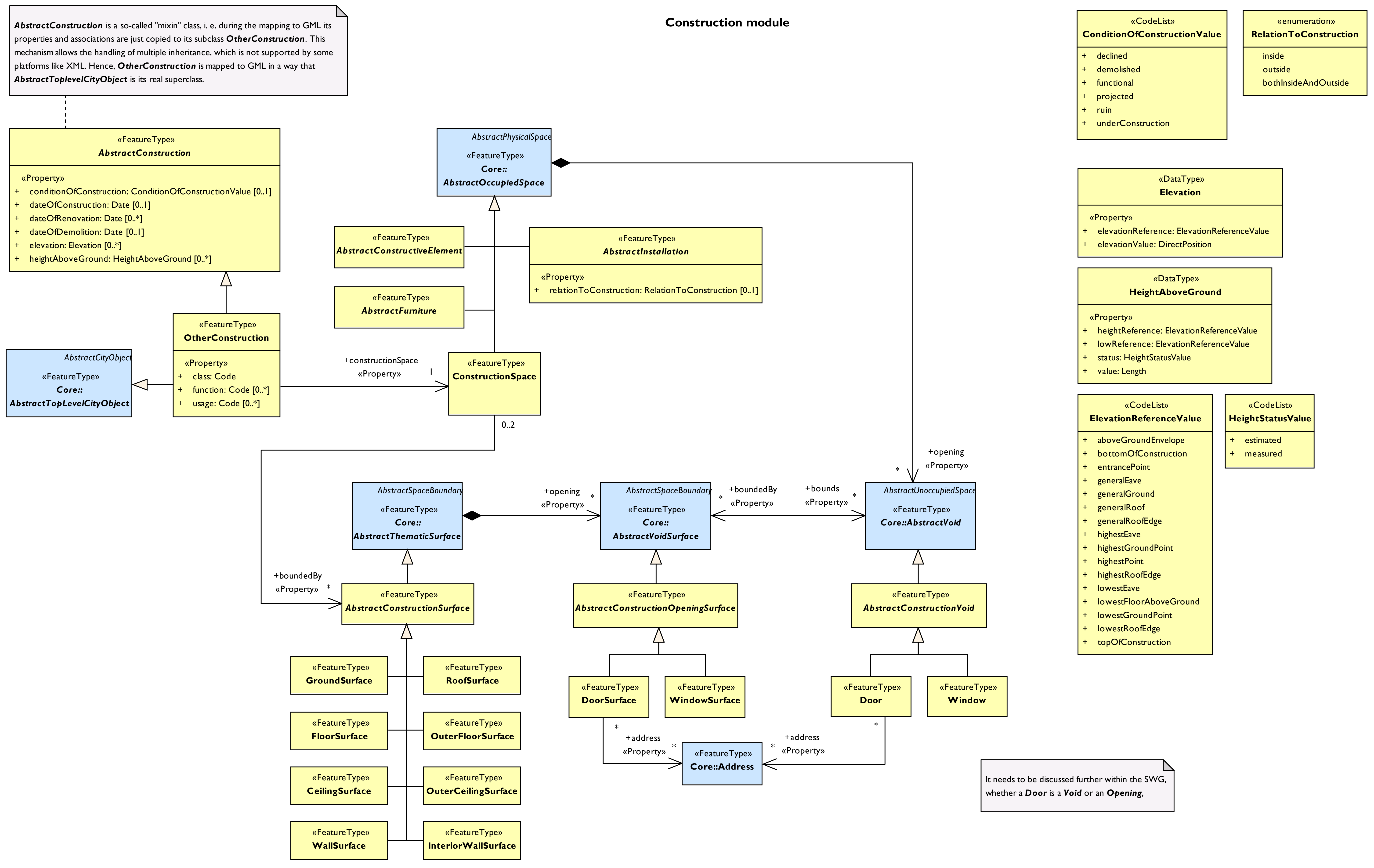


CityObjectGroup module



The additional class ***GroupSpace*** is required, because ***CityObjectGroup*** cannot be at the same time subclass of ***AbstractToplevelCityObject*** and ***AbstractLogicalSpace***.

Construction module

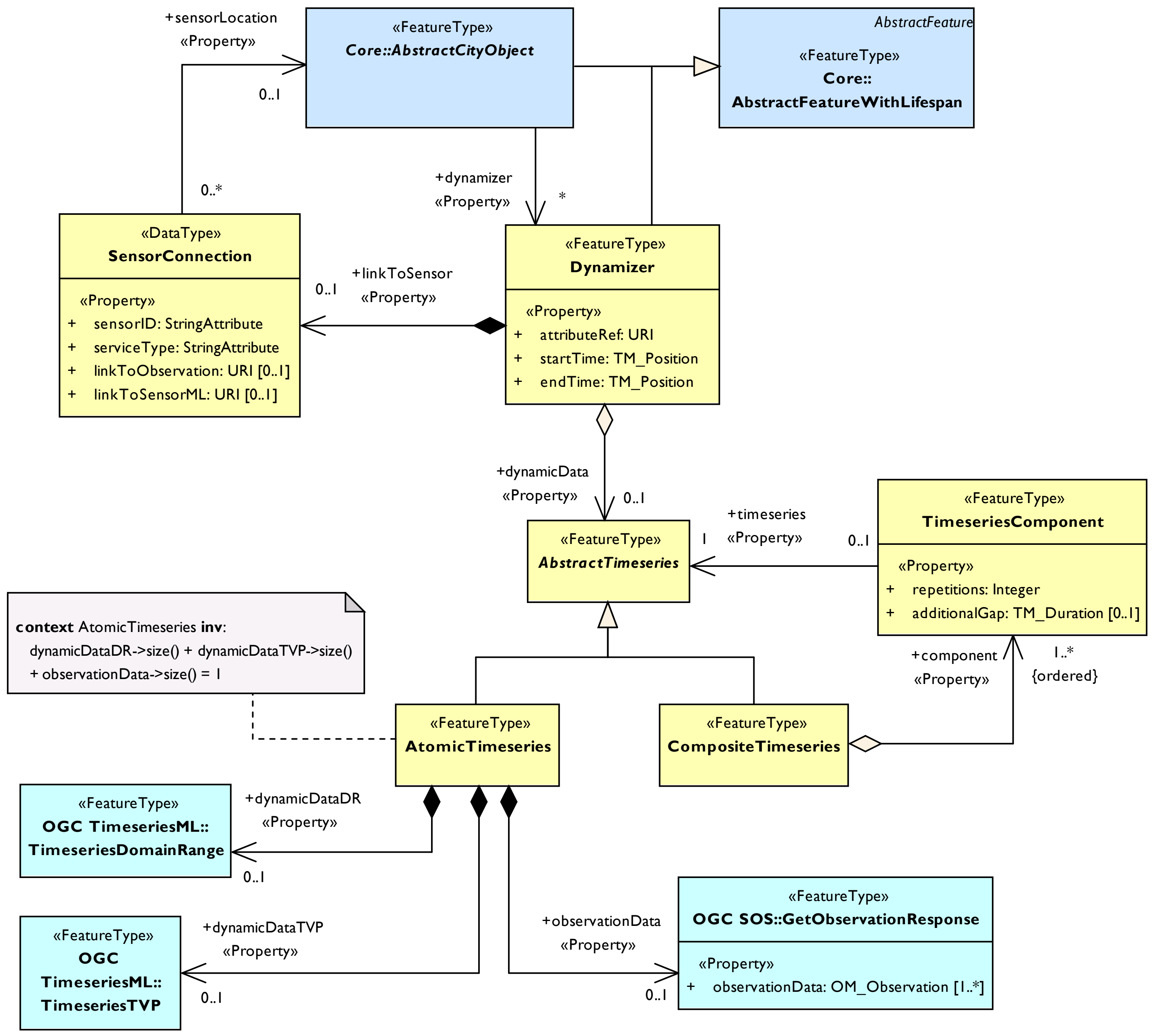


AbstractConstruction is a so-called "mixin" class, i. e. during the mapping to GML its properties and associations are just copied to its subclass ***OtherConstruction***. This mechanism allows the handling of multiple inheritance, which is not supported by some platforms like XML. Hence, ***OtherConstruction*** is mapped to GML in a way that ***AbstractToplevelCityObject*** is its real superclass.

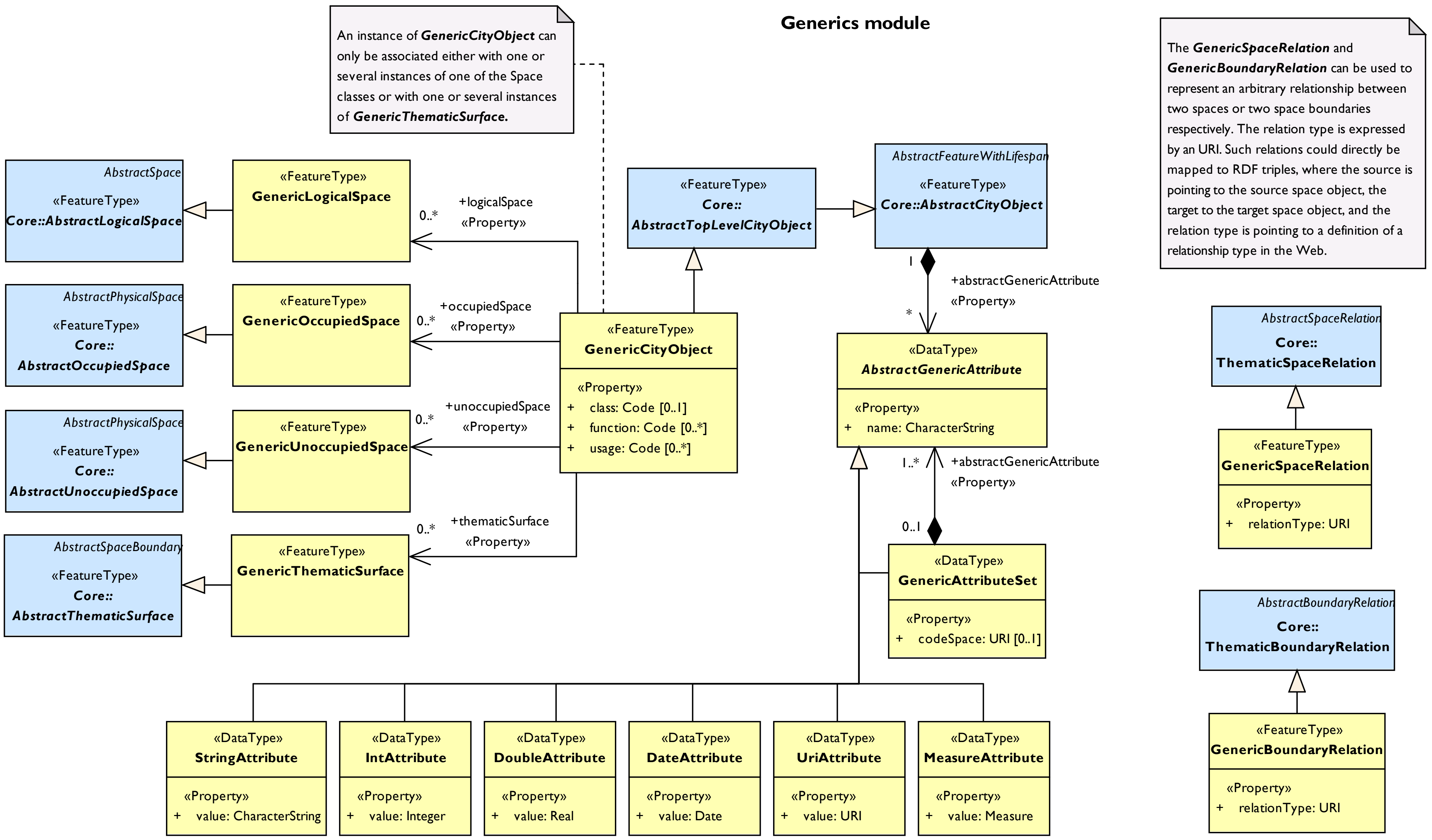
It needs to be discussed further within the SWG, whether a ***Door*** is a ***Void*** or an ***Opening***.

«FeatureType» AbstractConstruction
«Property»
+ conditionOfConstruction: ConditionOfConstructionValue [0..1]
+ dateOfConstruction: Date [0..1]
+ dateOfRenovation: Date [0..*]
+ dateOfDemolition: Date [0..1]
+ elevation: Elevation [0..*]
+ heightAboveGround: HeightAboveGround [0..*]
«FeatureType» OtherConstruction
«Property»
+ class: Code
+ function: Code [0..*]
+ usage: Code [0..*]
AbstractCityObject
«FeatureType» Core::AbstractTopLevelCityObject
+constructionSpace «Property» 1
«FeatureType» ConstructionSpace
0..2
AbstractPhysicalSpace
«FeatureType» Core::AbstractOccupiedSpace
«FeatureType» AbstractConstructiveElement
«FeatureType» AbstractFurniture
«FeatureType» AbstractInstallation
«Property»
+ relationToConstruction: RelationToConstruction [0..1]
«CodeList» ConditionOfConstructionValue
+ declined
+ demolished
+ functional
+ projected
+ ruin
+ underConstruction
«enumeration» RelationToConstruction
inside
outside
bothInsideAndOutside
«DataType» Elevation
«Property»
+ elevationReference: ElevationReferenceValue
+ elevationValue: DirectPosition
«DataType» HeightAboveGround
«Property»
+ heightReference: ElevationReferenceValue
+ lowReference: ElevationReferenceValue
+ status: HeightStatusValue
+ value: Length
«CodeList» ElevationReferenceValue
+ aboveGroundEnvelope
+ bottomOfConstruction
+ entrancePoint
+ generalEave
+ generalGround
+ generalRoof
+ generalRoofEdge
+ highestEave
+ highestGroundPoint
+ highestPoint
+ highestRoofEdge
+ lowestEave
+ lowestFloorAboveGround
+ lowestGroundPoint
+ lowestRoofEdge
+ topOfConstruction
«CodeList» HeightStatusValue
+ estimated
+ measured
+opening «Property» *
AbstractSpaceBoundary
«FeatureType» Core::AbstractThematicSurface
+opening «Property» *
AbstractSpaceBoundary
«FeatureType» Core::AbstractVoidSurface
* +boundedBy «Property»
+bounds «Property» *
AbstractUnoccupiedSpace
«FeatureType» Core::AbstractVoid
+boundedBy «Property» *
«FeatureType» AbstractConstructionSurface
«FeatureType» AbstractConstructionOpeningSurface
«FeatureType» AbstractConstructionVoid
«FeatureType» GroundSurface
«FeatureType» RoofSurface
«FeatureType» FloorSurface
«FeatureType» OuterFloorSurface
«FeatureType» CeilingSurface
«FeatureType» OuterCeilingSurface
«FeatureType» WallSurface
«FeatureType» InteriorWallSurface
«FeatureType» DoorSurface
«FeatureType» WindowSurface
«FeatureType» Door
«FeatureType» Window
* +address «Property» *
«FeatureType» Core::Address
* +address «Property» *

Dynamizer module



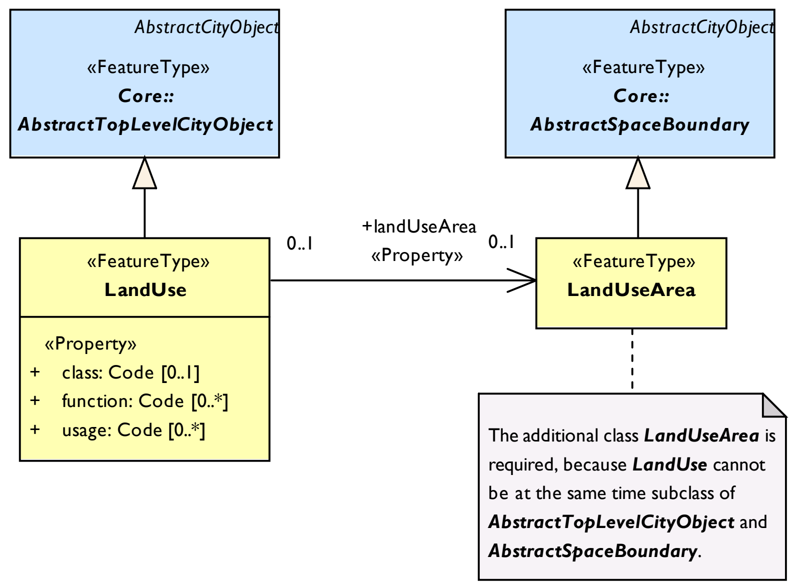
Generics module

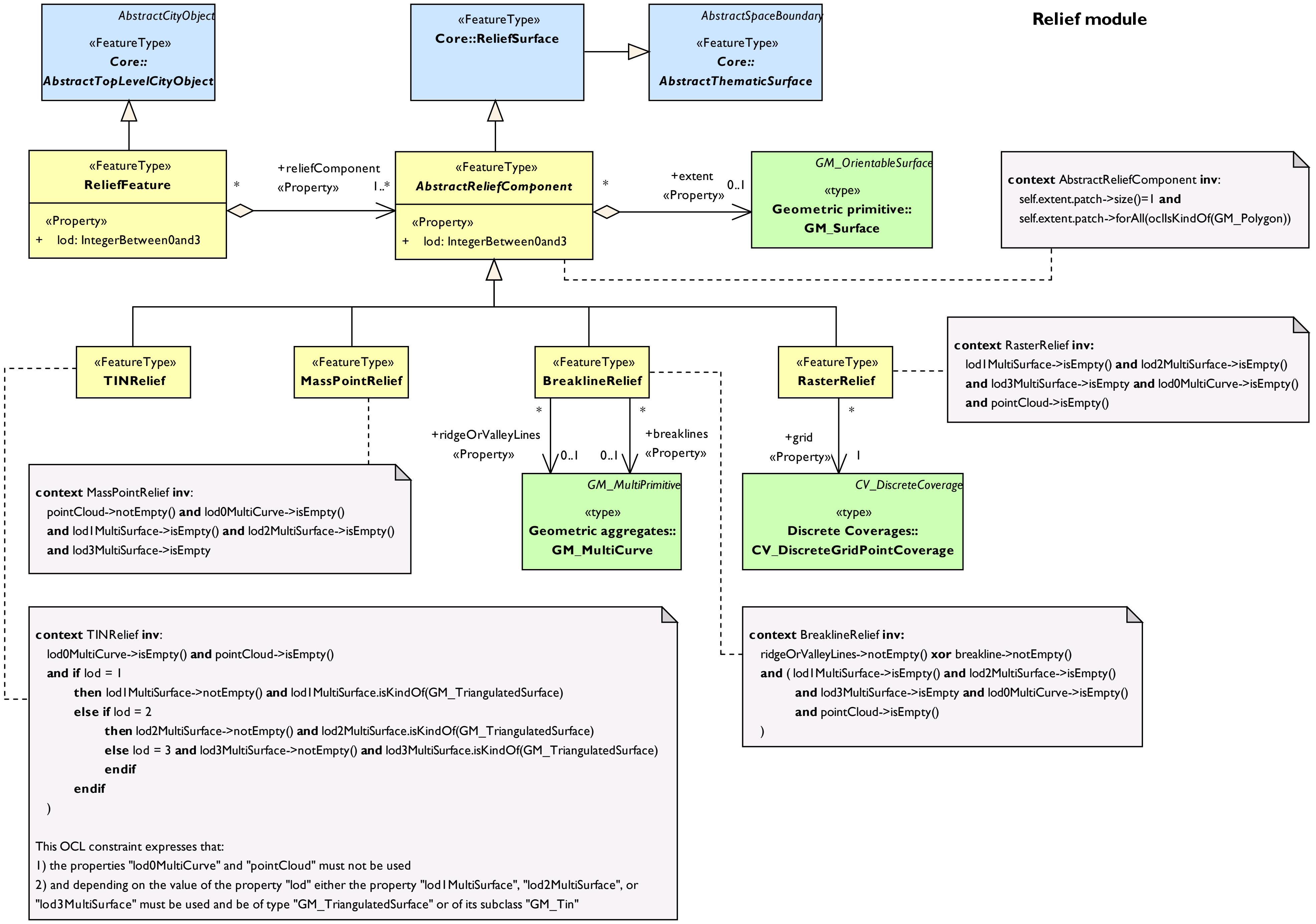


An instance of ***GenericCityObject*** can only be associated either with one or several instances of one of the Space classes or with one or several instances of ***GenericThematicSurface.***

The ***GenericSpaceRelation*** and ***GenericBoundaryRelation*** can be used to represent an arbitrary relationship between two spaces or two space boundaries respectively. The relation type is expressed by an URI. Such relations could directly be mapped to RDF triples, where the source is pointing to the source space object, the target to the target space object, and the relation type is pointing to a definition of a relationship type in the Web.

LandUse module





```
context AbstractReliefComponent inv:
   self.extent.patch->size()=1 and
   self.extent.patch->forAll(oclIsKindOf(GM_Polygon))
```

```
context RasterRelief inv:
   lod1MultiSurface->isEmpty() and lod2MultiSurface->isEmpty()
   and lod3MultiSurface->isEmpty and lod0MultiCurve->isEmpty()
   and pointCloud->isEmpty()
```

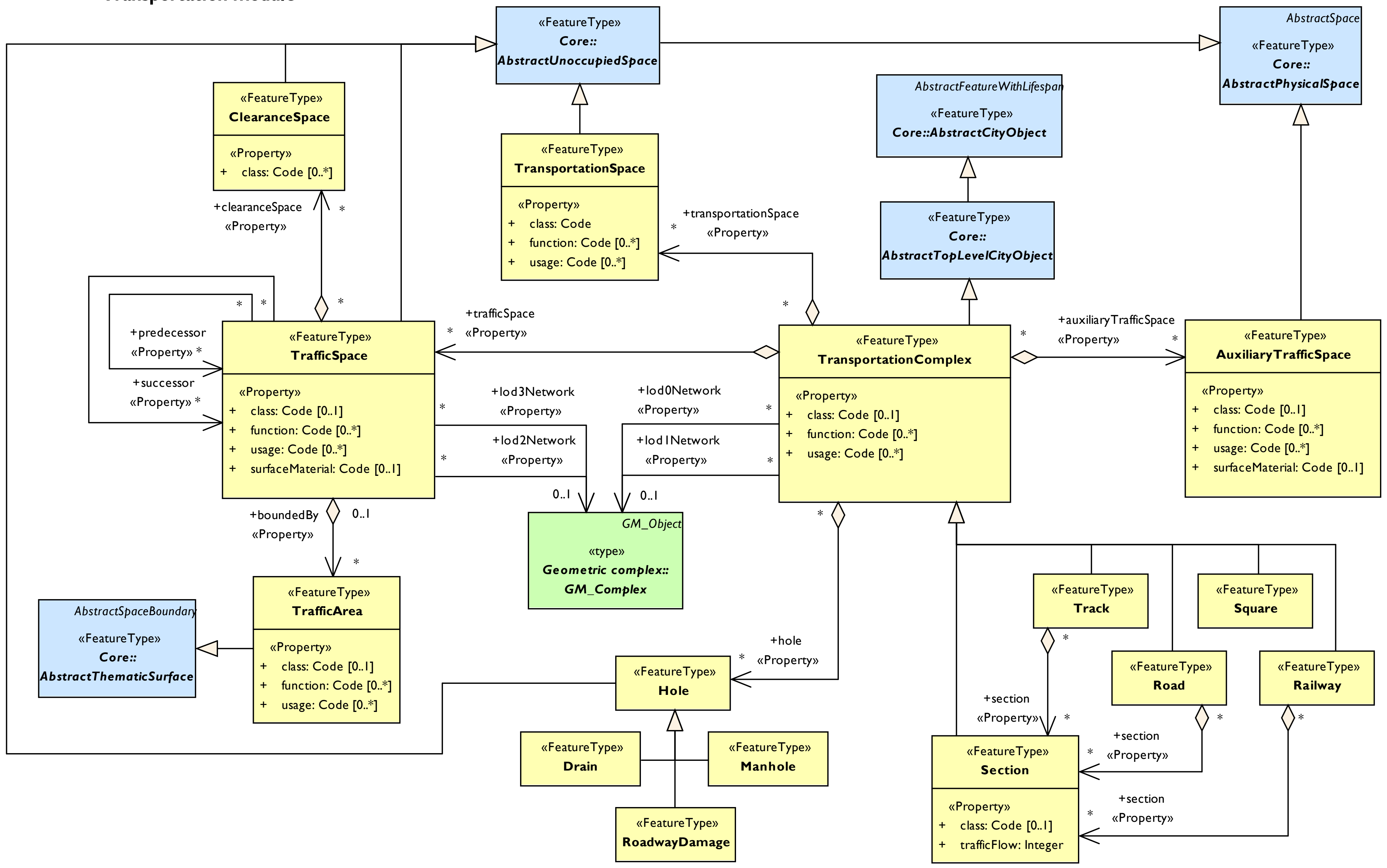
```
context MassPointRelief inv:
   pointCloud->notEmpty() and lod0MultiCurve->isEmpty()
   and lod1MultiSurface->isEmpty() and lod2MultiSurface->isEmpty()
   and lod3MultiSurface->isEmpty
```

```
context TINRelief inv:
   lod0MultiCurve->isEmpty() and pointCloud->isEmpty()
   and if lod = 1
        then lod1MultiSurface->notEmpty() and lod1MultiSurface.isKindOf(GM_TriangulatedSurface)
        else if lod = 2
              then lod2MultiSurface->notEmpty() and lod2MultiSurface.isKindOf(GM_TriangulatedSurface)
              else lod = 3 and lod3MultiSurface->notEmpty() and lod3MultiSurface.isKindOf(GM_TriangulatedSurface)
              endif
        endif
   )
```

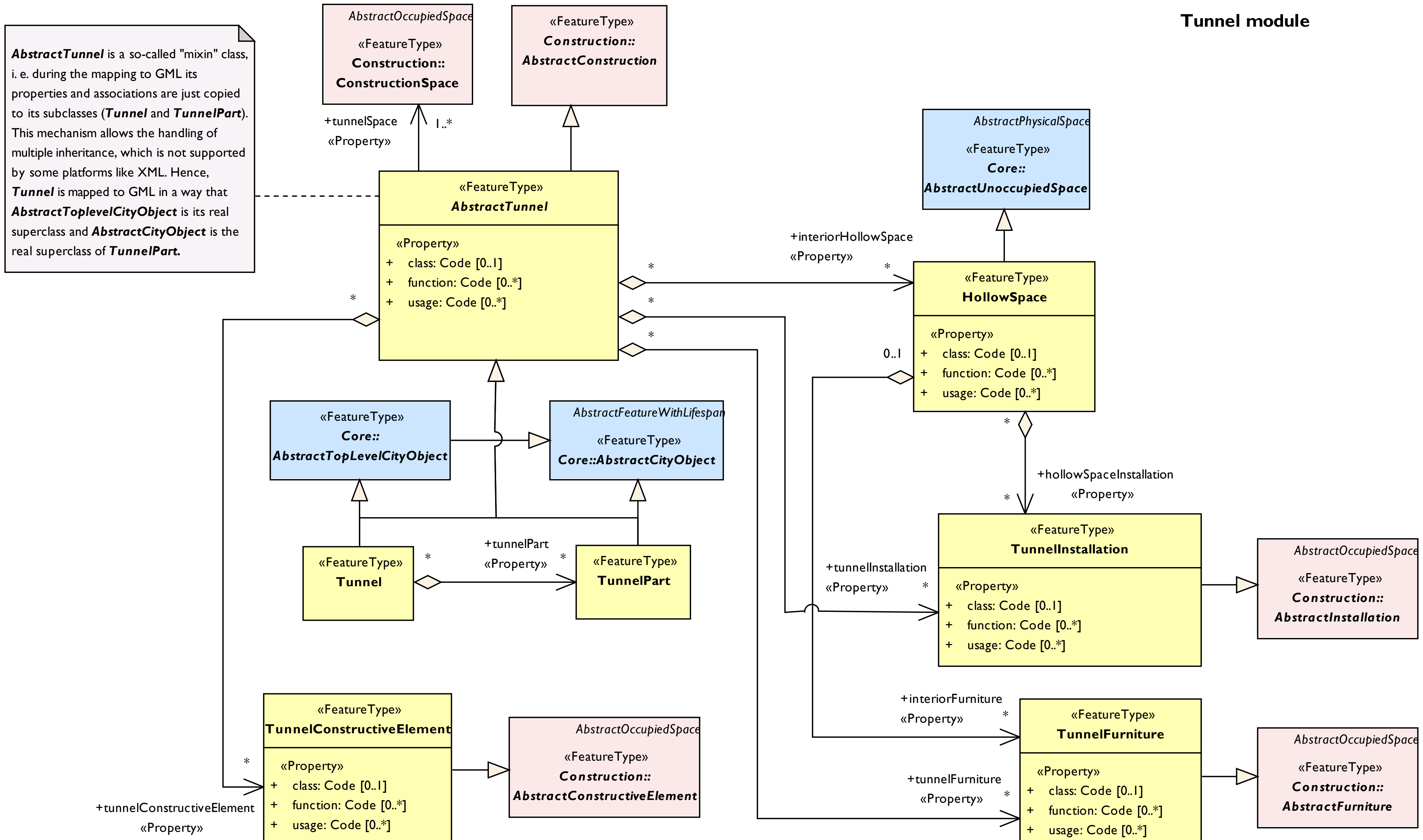
This OCL constraint expresses that:
1) the properties "lod0MultiCurve" and "pointCloud" must not be used
2) and depending on the value of the property "lod" either the property "lod1MultiSurface", "lod2MultiSurface", or "lod3MultiSurface" must be used and be of type "GM_TriangulatedSurface" or of its subclass "GM_Tin"

```
context BreaklineRelief inv:
   ridgeOrValleyLines->notEmpty() xor breakline->notEmpty()
   and ( lod1MultiSurface->isEmpty() and lod2MultiSurface->isEmpty()
         and lod3MultiSurface->isEmpty and lod0MultiCurve->isEmpty()
         and pointCloud->isEmpty()
   )
```

Transportation module

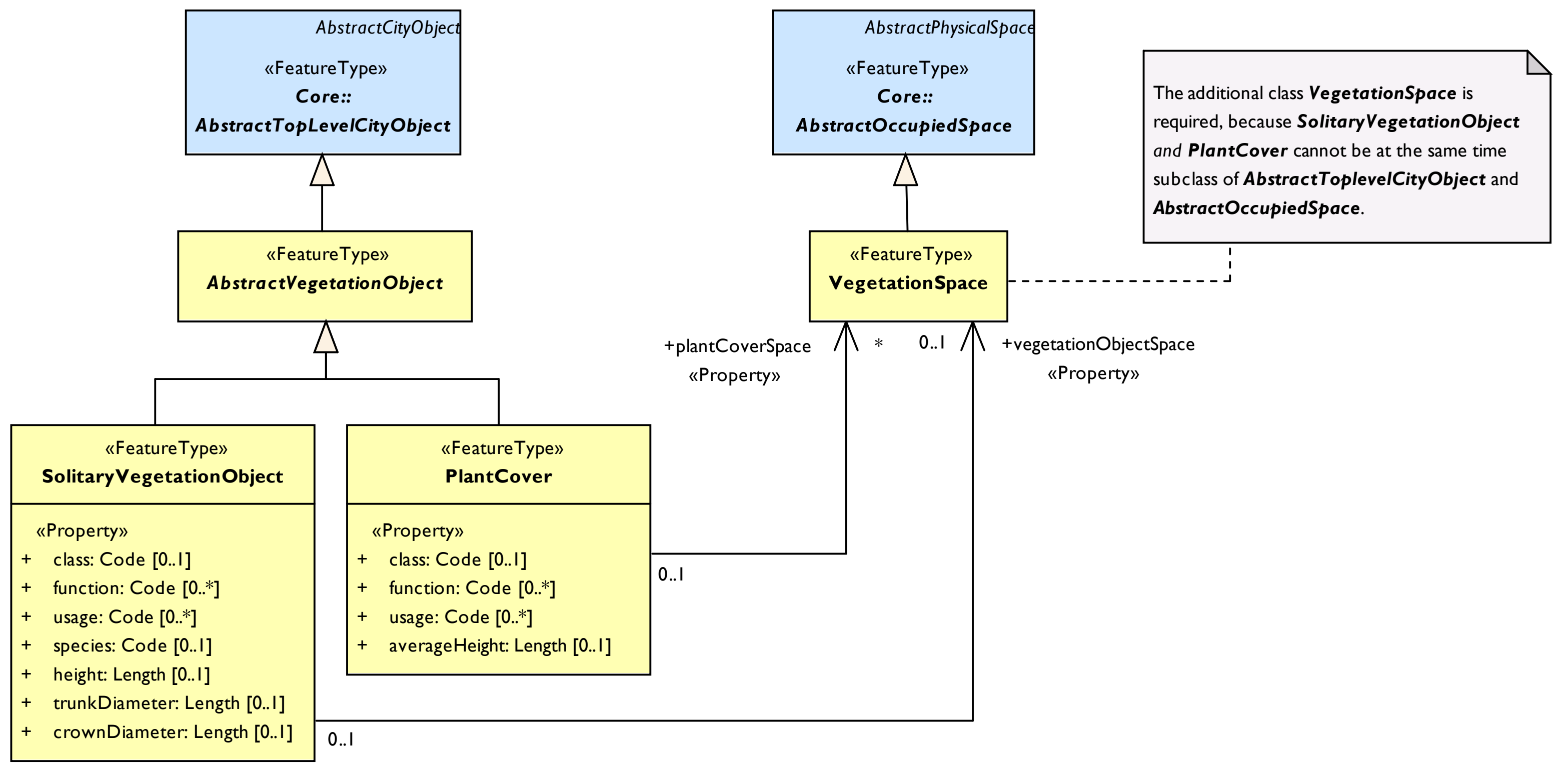


Tunnel module

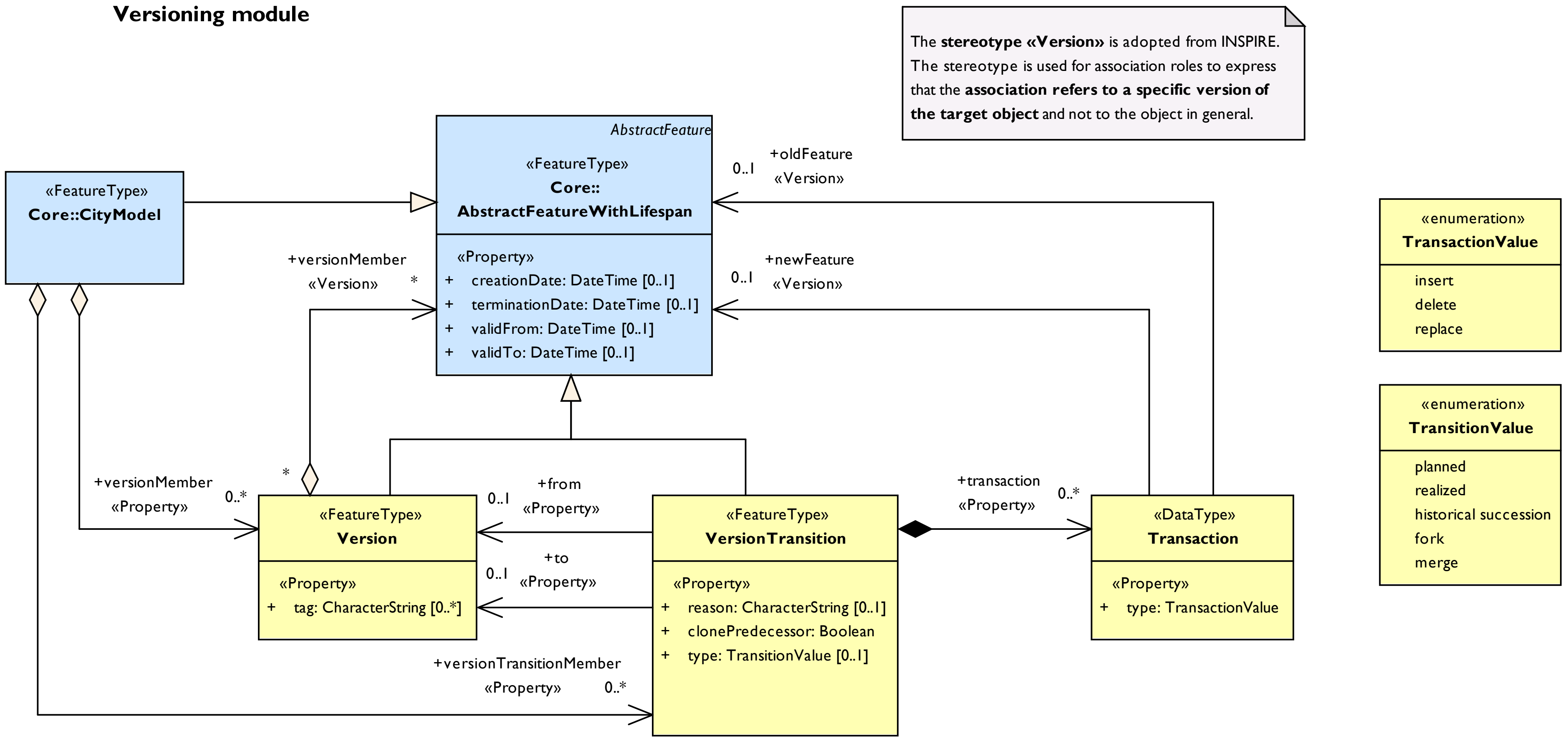


AbstractTunnel is a so-called "mixin" class, i. e. during the mapping to GML its properties and associations are just copied to its subclasses (*Tunnel* and *TunnelPart*). This mechanism allows the handling of multiple inheritance, which is not supported by some platforms like XML. Hence, *Tunnel* is mapped to GML in a way that *AbstractToplevelCityObject* is its real superclass and *AbstractCityObject* is the real superclass of *TunnelPart.*
AbstractOccupiedSpace
«FeatureType»
Construction::ConstructionSpace
«FeatureType»
Construction::AbstractConstruction
+tunnelSpace «Property»
1..*
AbstractPhysicalSpace
«FeatureType»
Core::AbstractUnoccupiedSpace
«FeatureType»
AbstractTunnel
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
+interiorHollowSpace «Property»
«FeatureType»
HollowSpace
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
0..1
«FeatureType»
Core::AbstractTopLevelCityObject
AbstractFeatureWithLifespan
«FeatureType»
Core::AbstractCityObject
+hollowSpaceInstallation «Property»
«FeatureType»
Tunnel
+tunnelPart «Property»
«FeatureType»
TunnelPart
«FeatureType»
TunnelInstallation
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
+tunnelInstallation «Property»
AbstractOccupiedSpace
«FeatureType»
Construction::AbstractInstallation
«FeatureType»
TunnelConstructiveElement
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
+tunnelConstructiveElement «Property»
AbstractOccupiedSpace
«FeatureType»
Construction::AbstractConstructiveElement
+interiorFurniture «Property»
«FeatureType»
TunnelFurniture
«Property»
+ class: Code [0..1]
+ function: Code [0..*]
+ usage: Code [0..*]
+tunnelFurniture «Property»
AbstractOccupiedSpace
«FeatureType»
Construction::AbstractFurniture

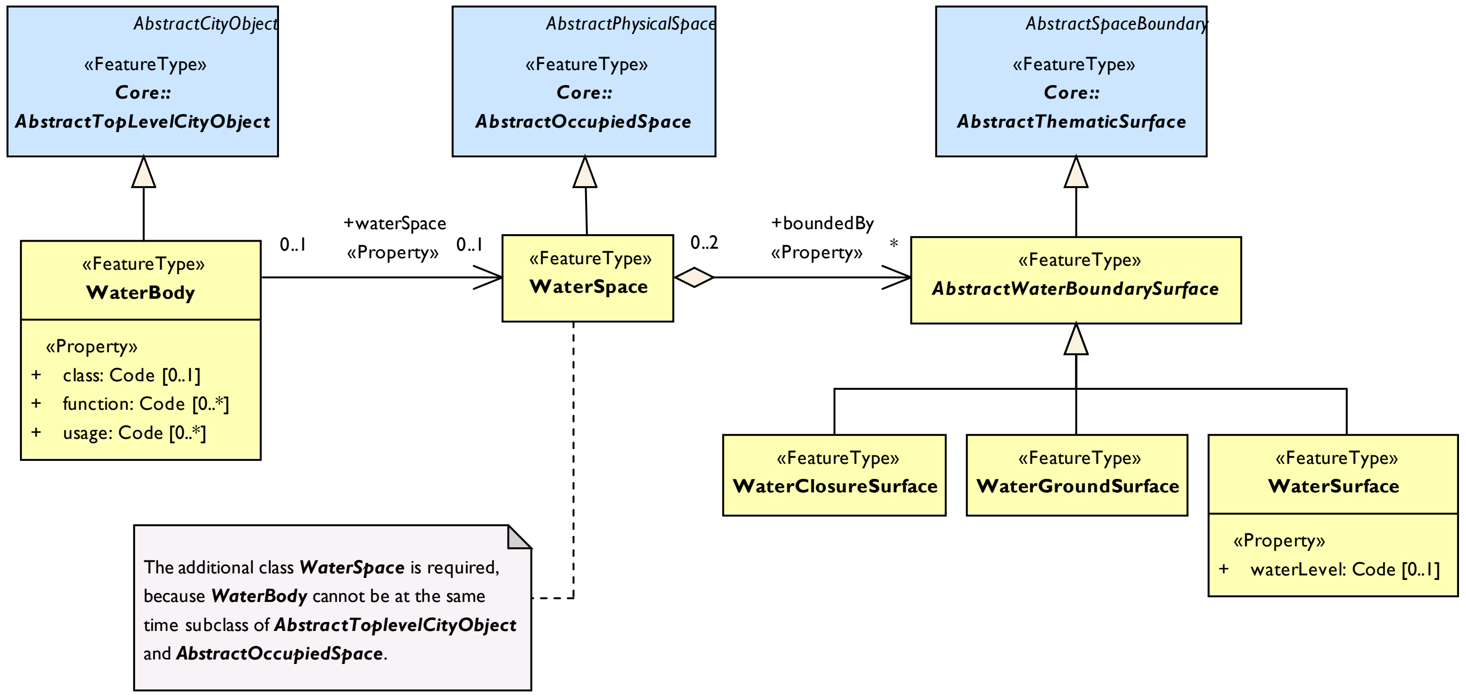
Vegetation module



Versioning module



WaterBody module



The additional class ***WaterSpace*** is required, because ***WaterBody*** cannot be at the same time subclass of ***AbstractToplevelCityObject*** and ***AbstractOccupiedSpace***.