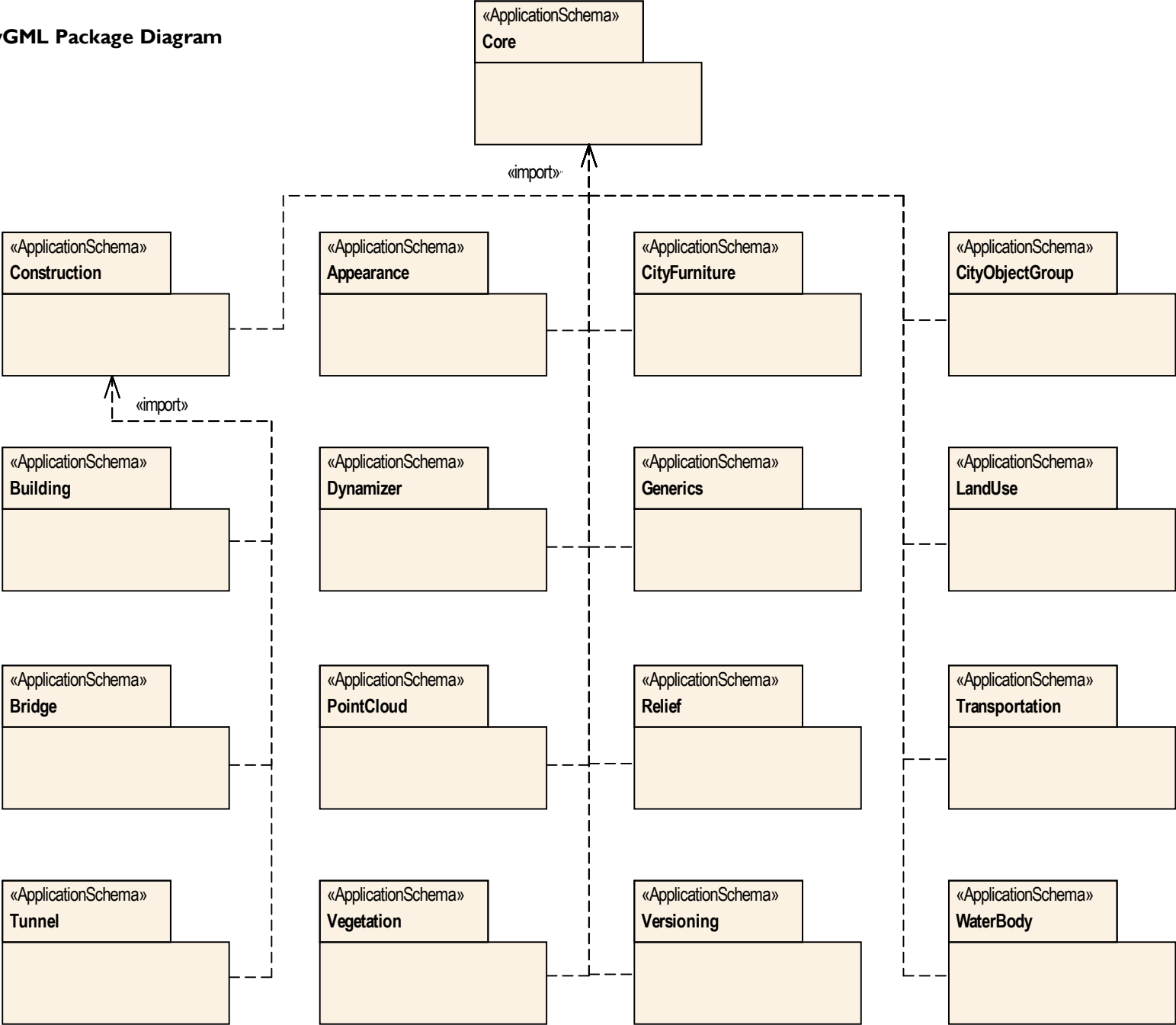
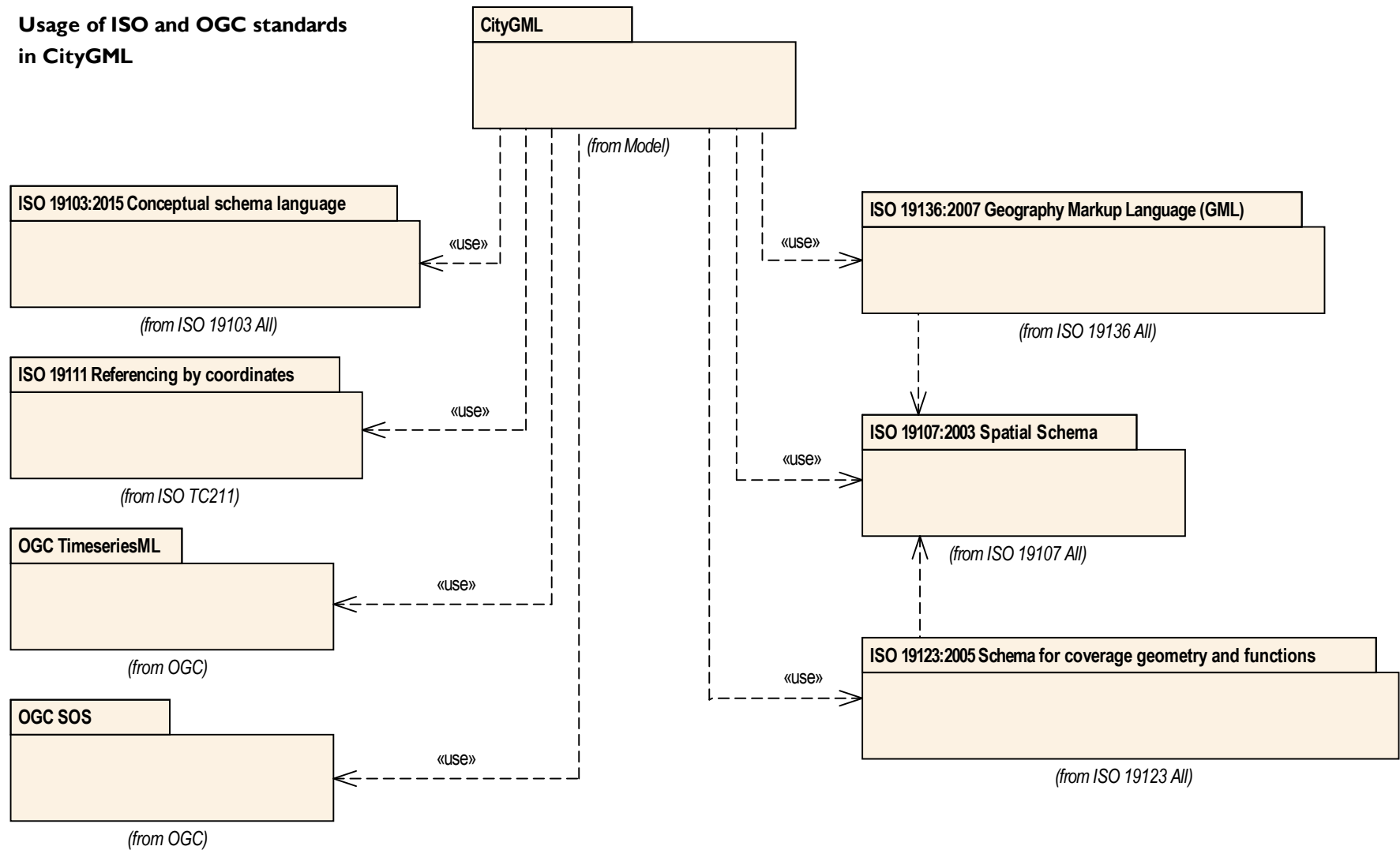
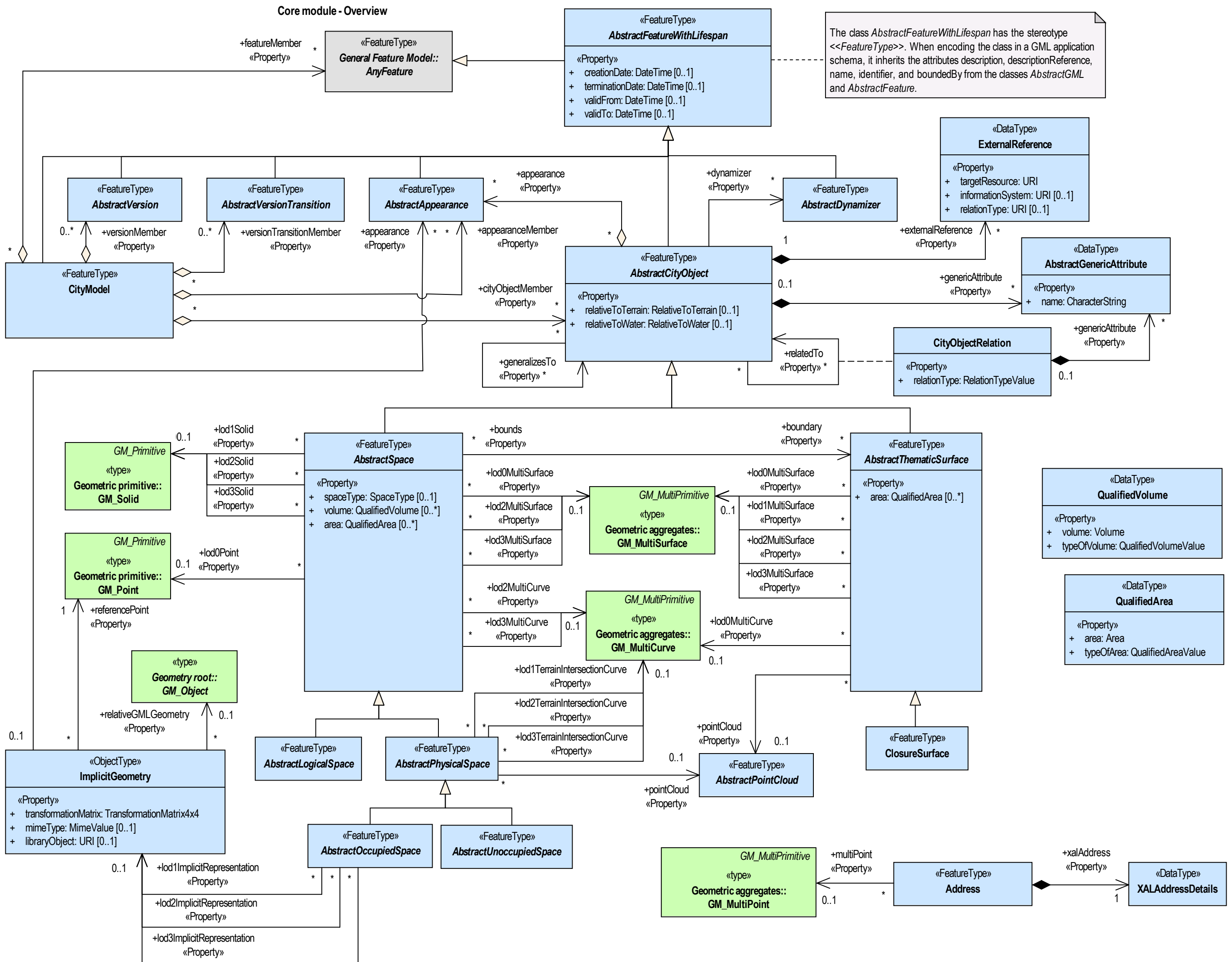


CityGML Package Diagram

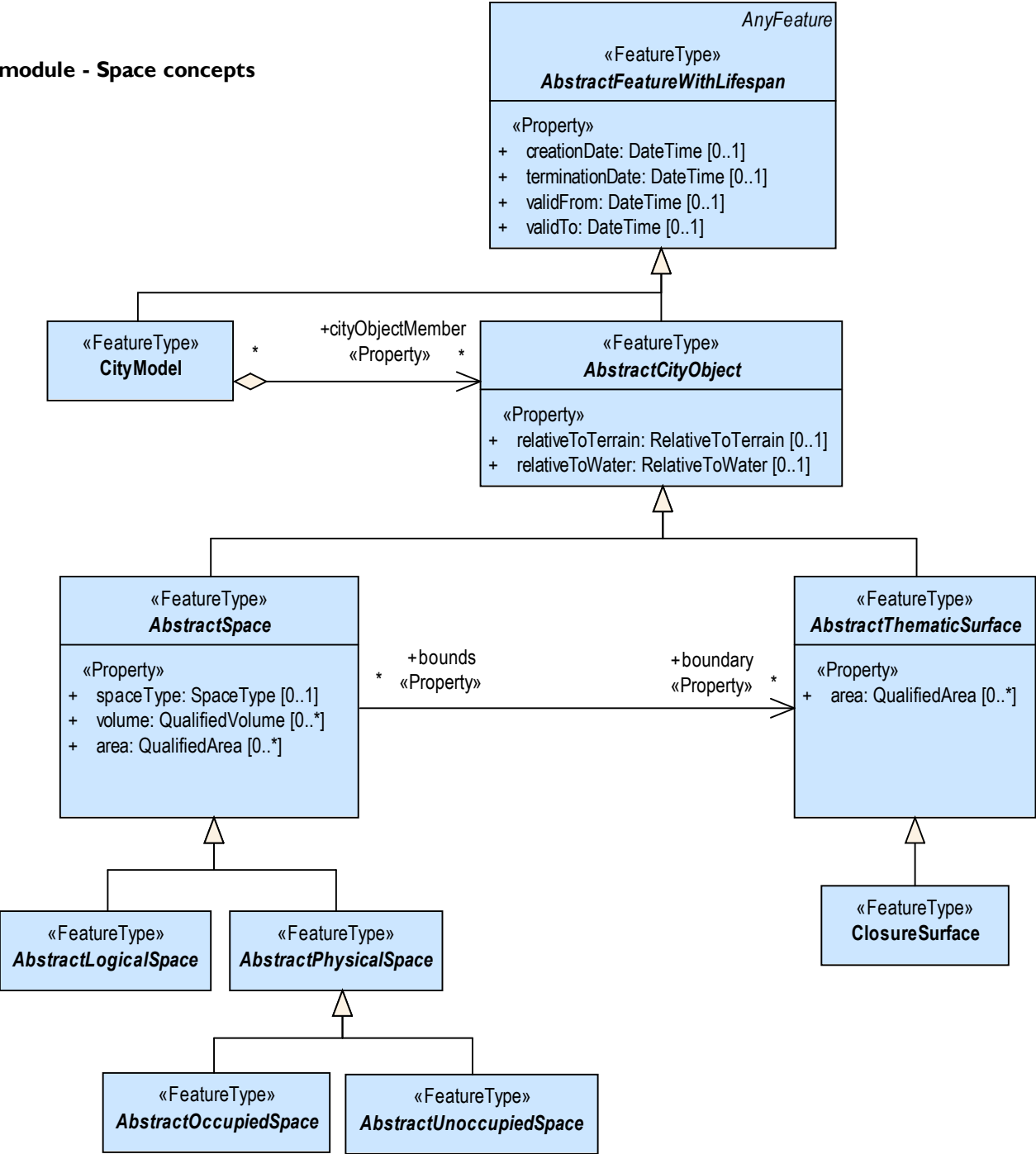


## Usage of ISO and OGC standards in CityGML

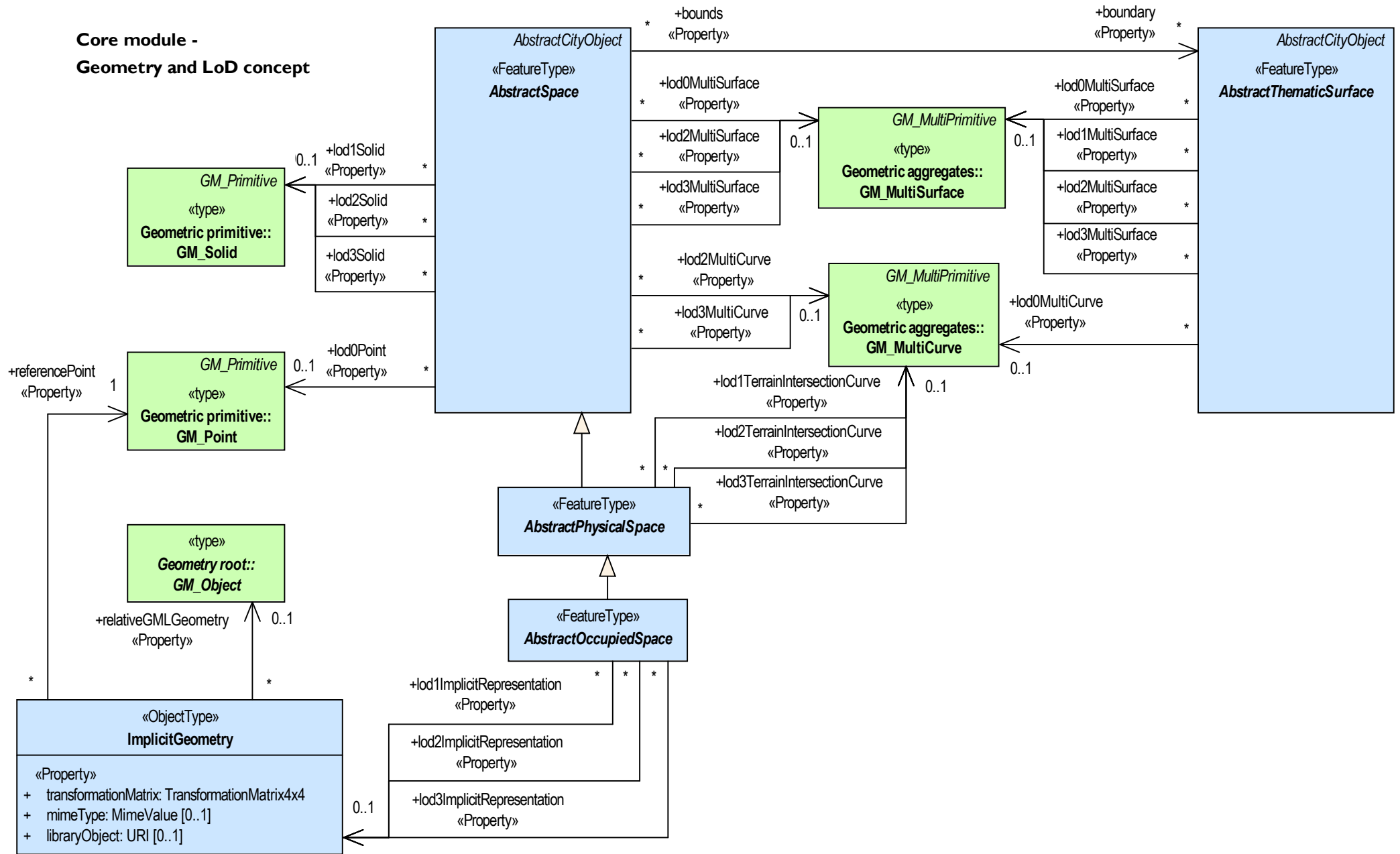




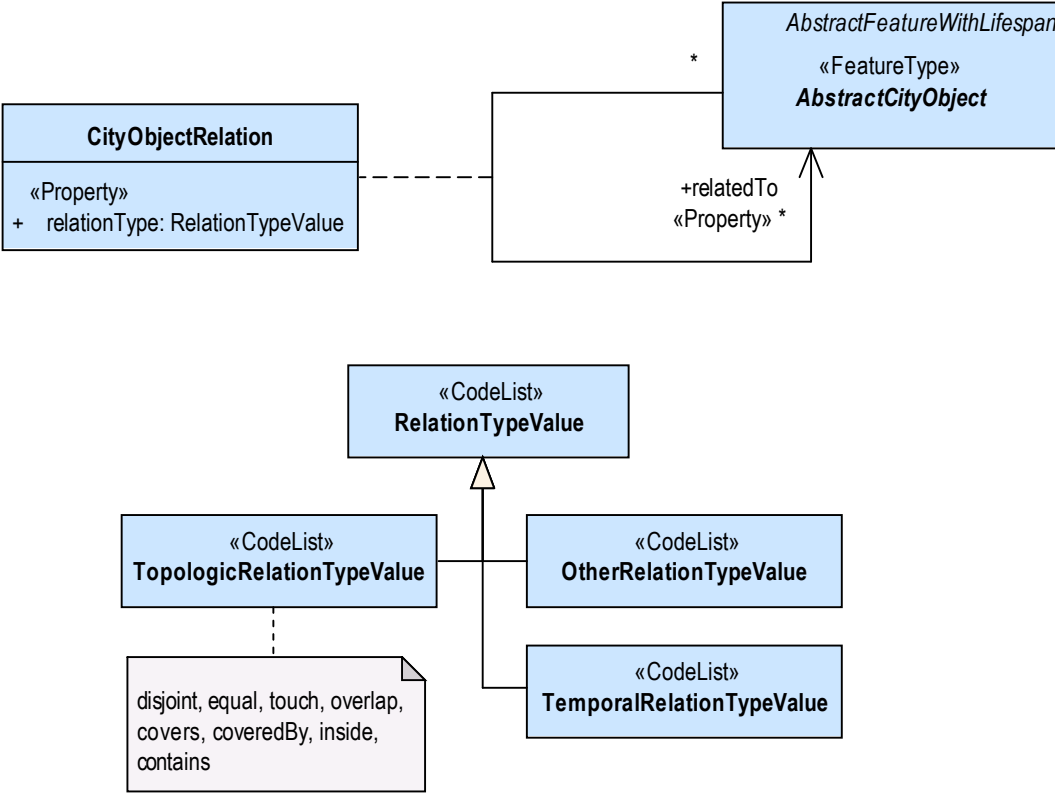
Core module - Space concepts



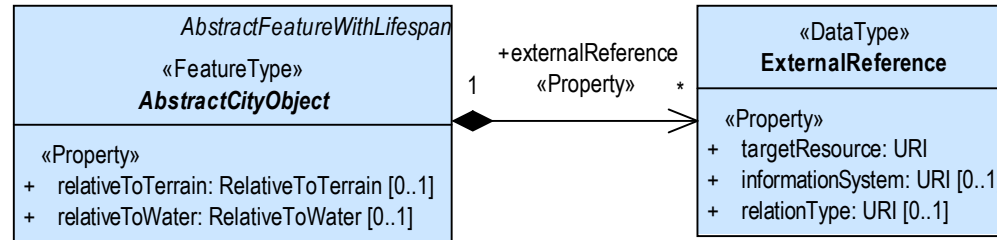
**Core module -  
Geometry and LoD concept**



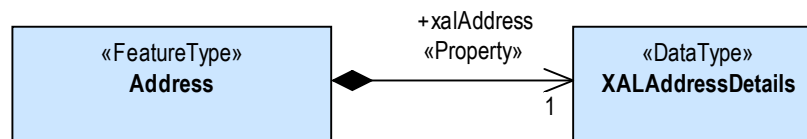
Core module - City object relations



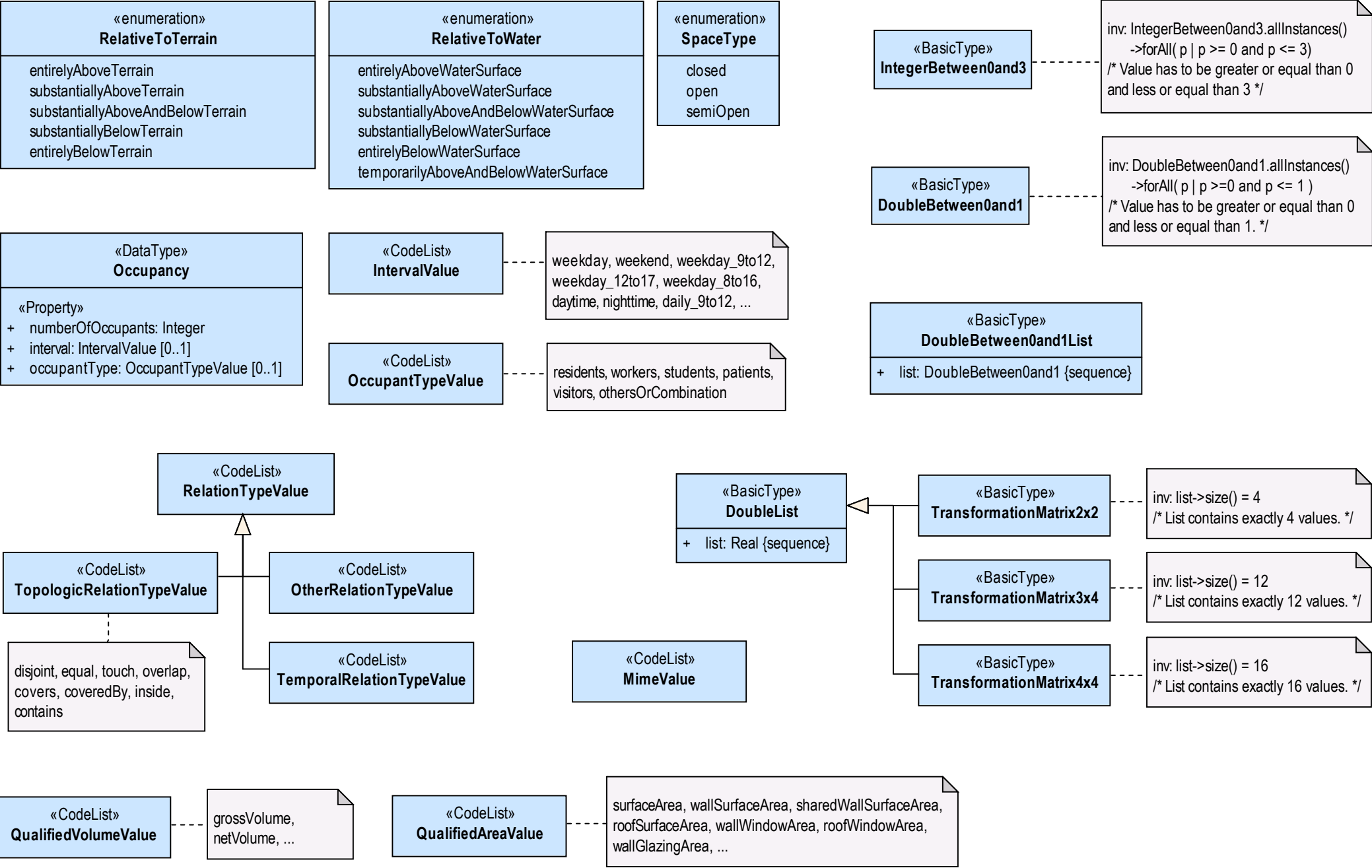
## Core module - Miscellaneous



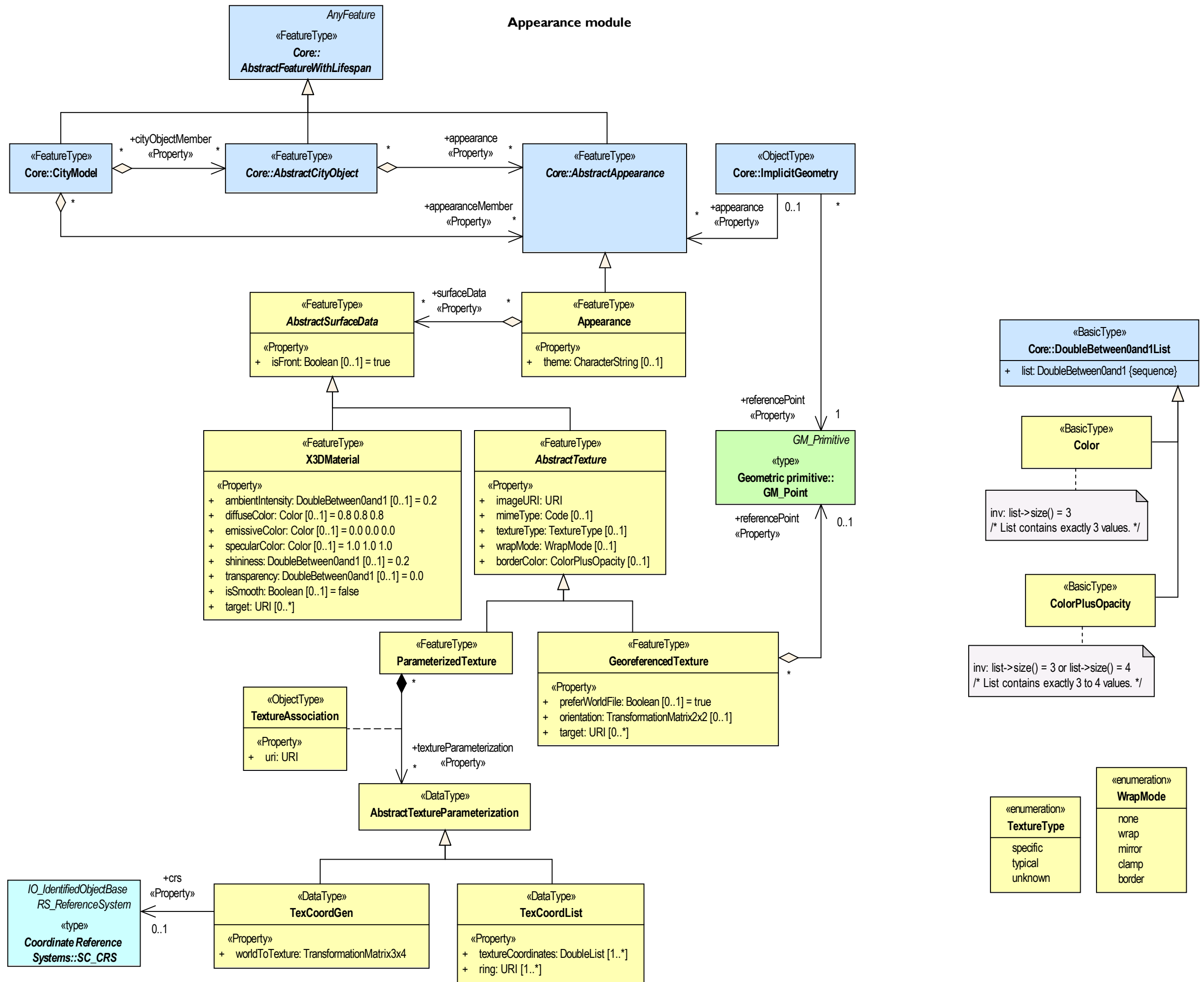
**ExternalReference** is now extended by an optional **relationType** which can link to some external definition of the type of relation (e.g. the *sameAs* relation from OWL). Hence, **ExternalReferences** can now be used to express relations similar to RDF.



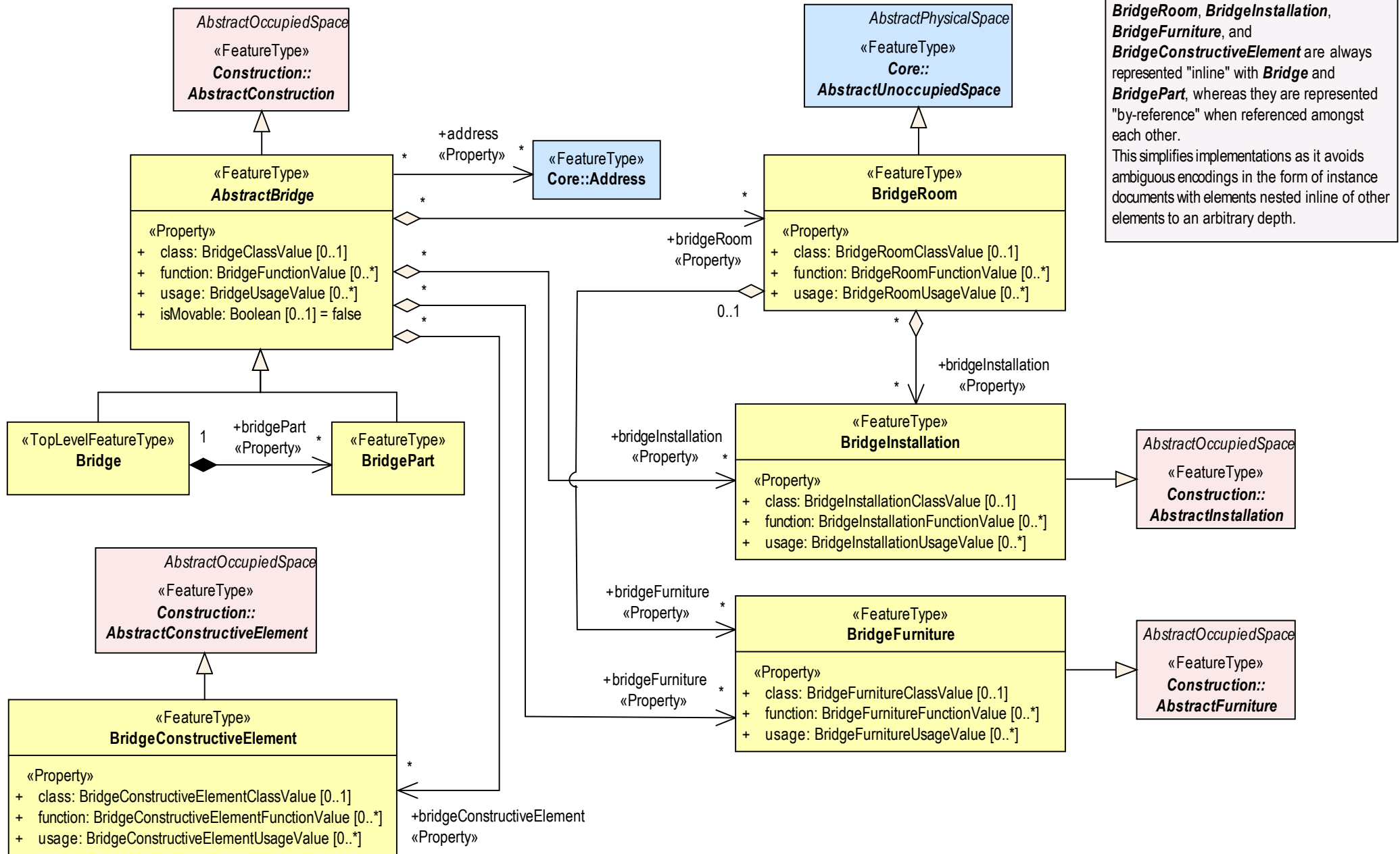
Core module - Basic Types, Enumerations, and Code lists



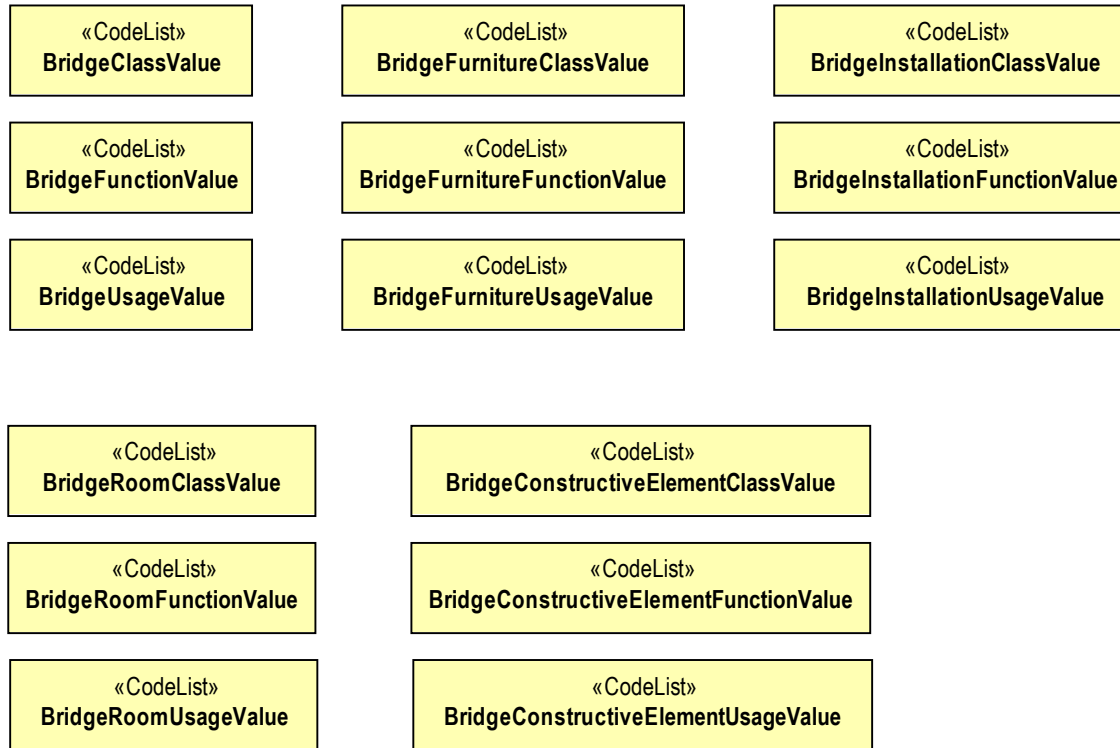


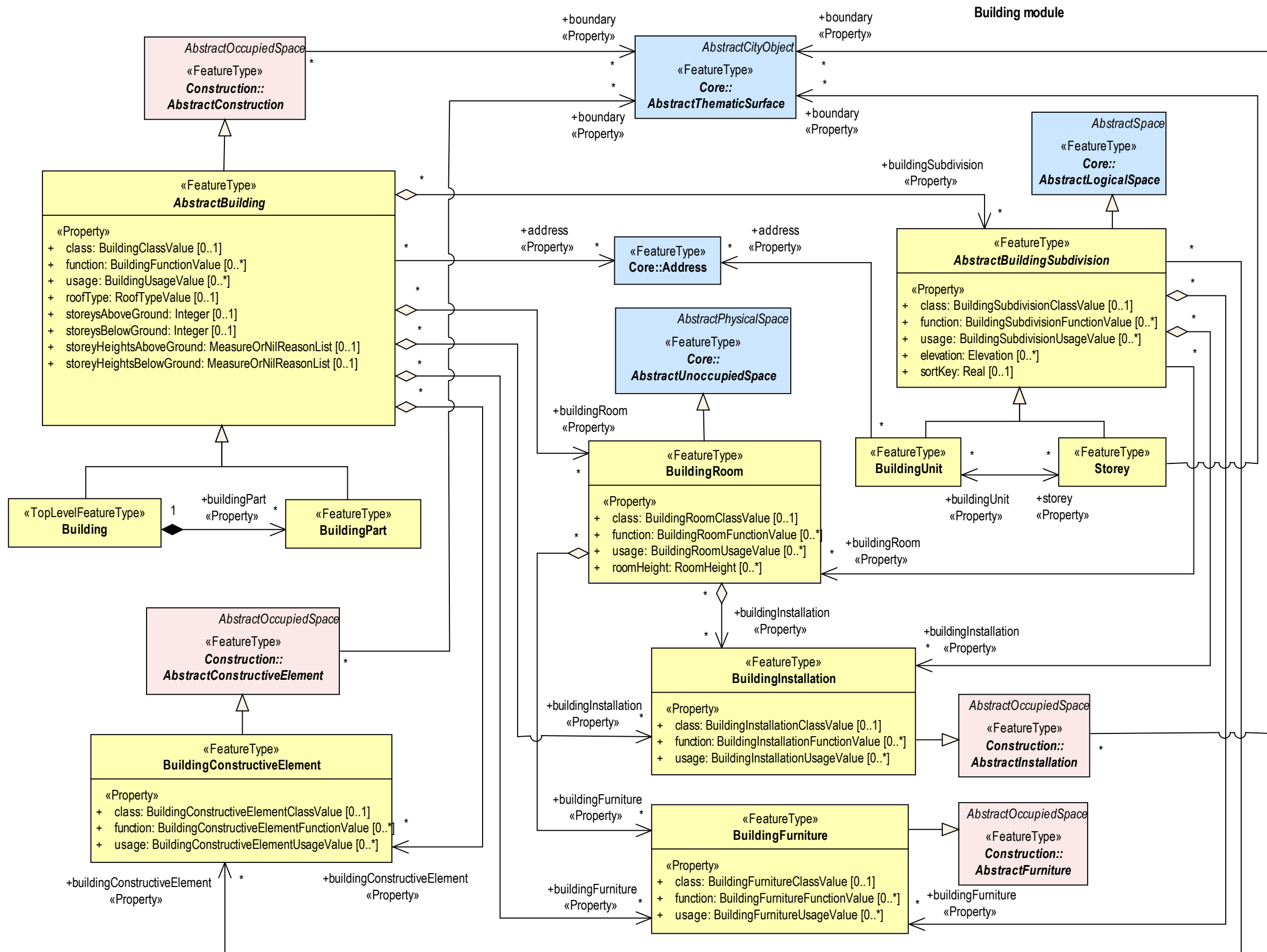


## Bridge module



## Bridge module - Code lists





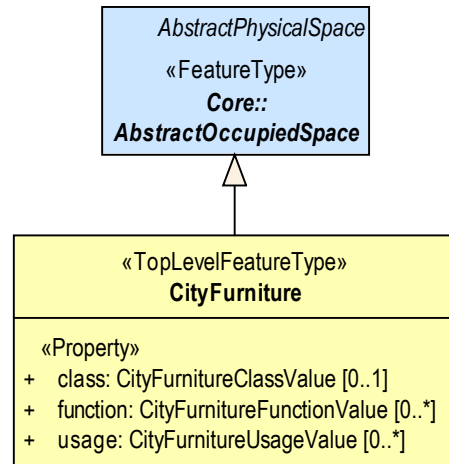
**BuildingRoom, BuildingInstallation, BuildingFurniture, and BuildingConstructiveElement** are always represented "inline" with **Building** and **BuildingPart**, whereas they are represented "by-reference" when referenced amongst each other.

This simplifies implementations as it avoids ambiguous encodings in the form of instance documents with elements nested inline of other elements to an arbitrary depth.

## Building module - Code lists

«CodeList» <b>BuildingClassValue</b>	«CodeList» <b>BuildingInstallationClassValue</b>	«CodeList» <b>BuildingFurnitureClassValue</b>	«CodeList» <b>RoofTypeValue</b>
«CodeList» <b>BuildingFunctionValue</b>	«CodeList» <b>BuildingInstallationFunctionValue</b>	«CodeList» <b>BuildingFurnitureFunctionValue</b>	
«CodeList» <b>BuildingUsageValue</b>	«CodeList» <b>BuildingInstallationUsageValue</b>	«CodeList» <b>BuildingFurnitureUsageValue</b>	
«CodeList» <b>BuildingRoomClassValue</b>	«CodeList» <b>BuildingConstructiveElementClassValue</b>	«CodeList» <b>BuildingSubdivisionClassValue</b>	
«CodeList» <b>BuildingRoomFunctionValue</b>	«CodeList» <b>BuildingConstructiveElementFunctionValue</b>	«CodeList» <b>BuildingSubdivisionFunctionValue</b>	
«CodeList» <b>BuildingRoomUsageValue</b>	«CodeList» <b>BuildingConstructiveElementUsageValue</b>	«CodeList» <b>BuildingSubdivisionUsageValue</b>	

## CityFurniture module



## CityFurniture module - Code lists

«CodeList»

**CityFurnitureClassValue**

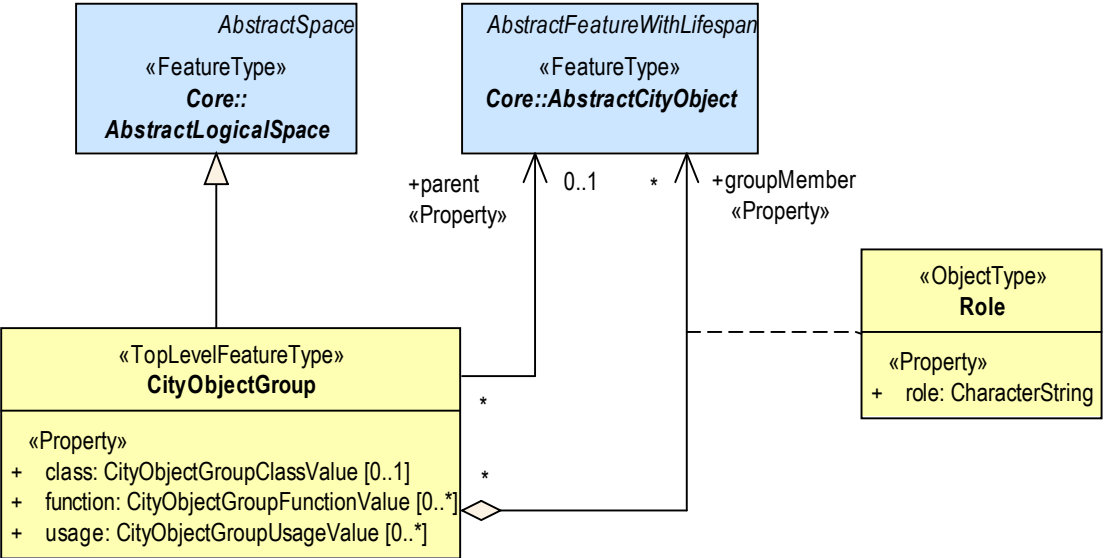
«CodeList»

**CityFurnitureFunctionValue**

«CodeList»

**CityFurnitureUsageValue**

CityObjectGroup module





## CityObjectGroup module - Code lists

«CodeList»

**CityObjectGroupClassValue**

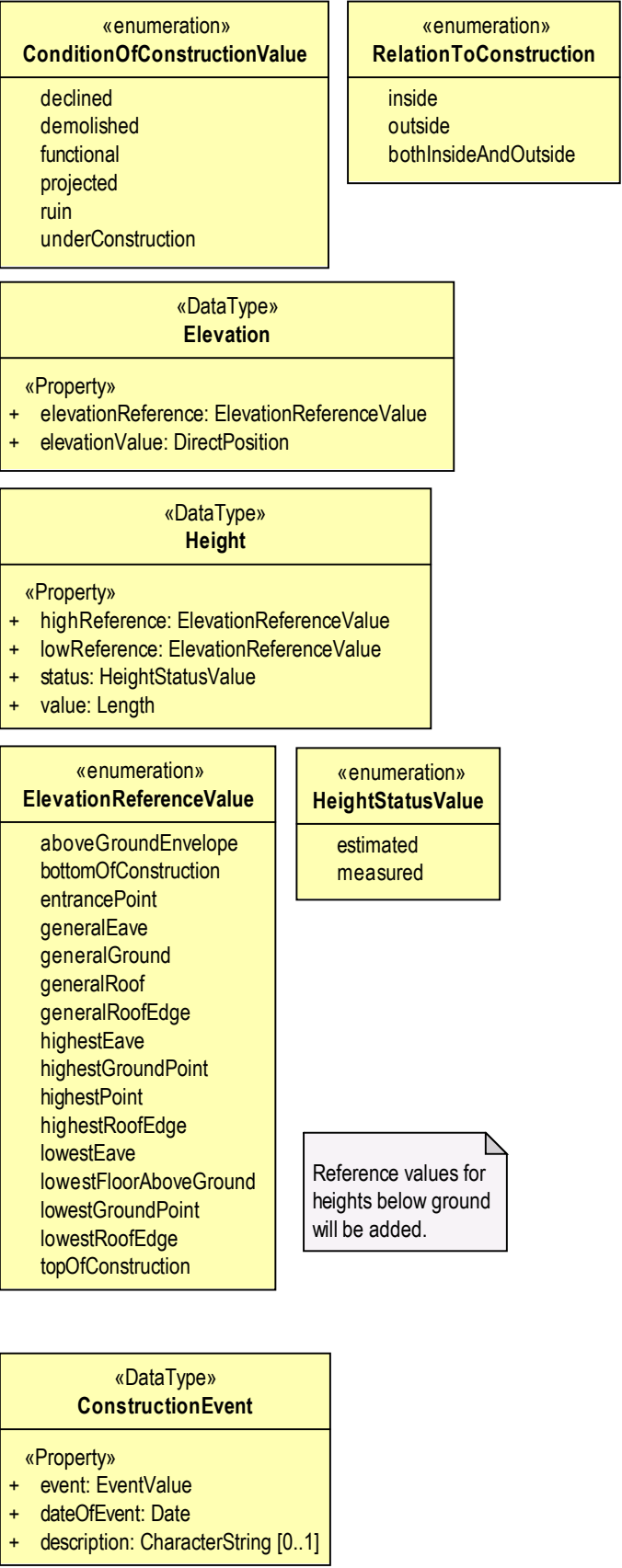
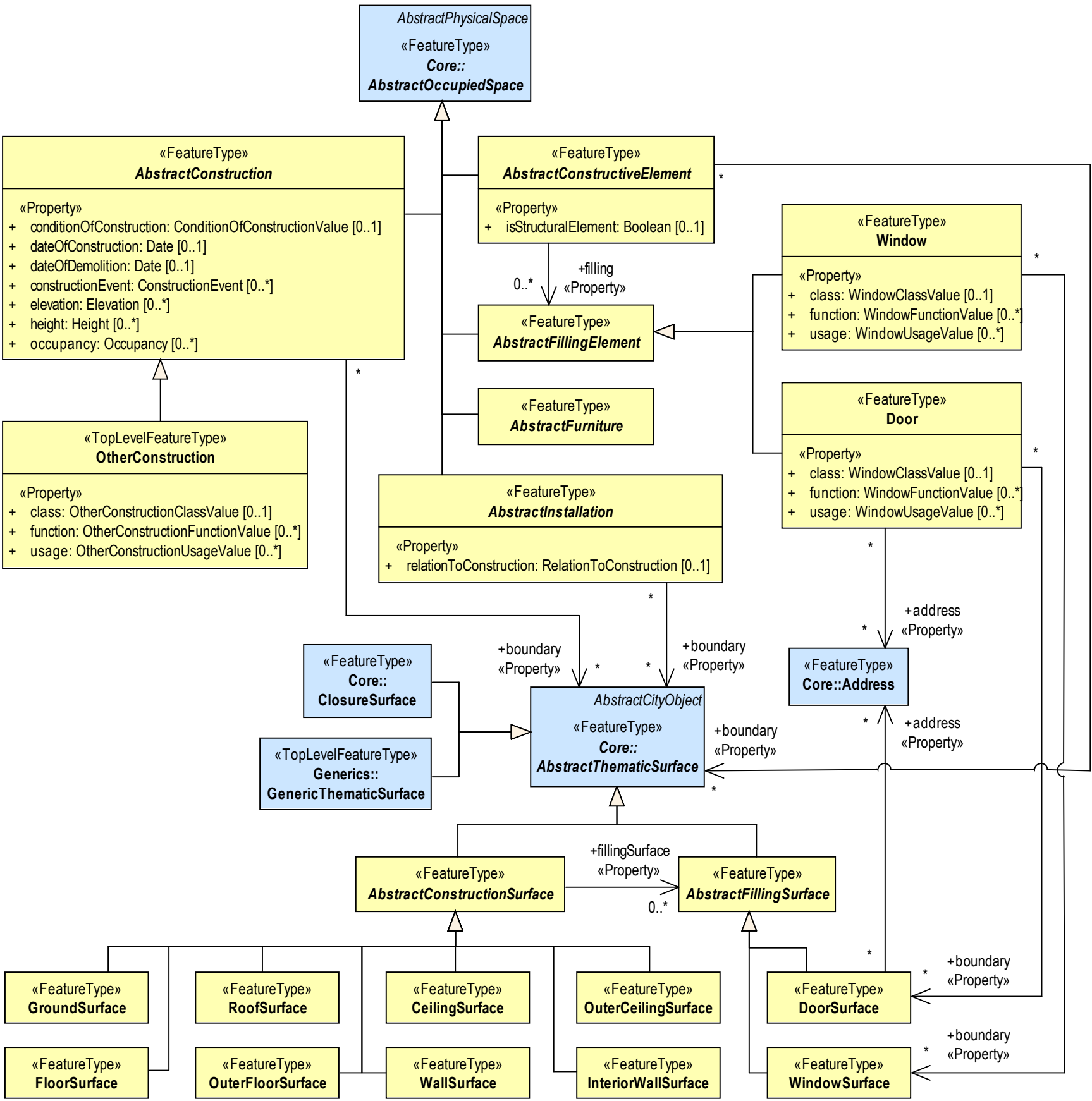
«CodeList»

**CityObjectGroupFunctionValue**

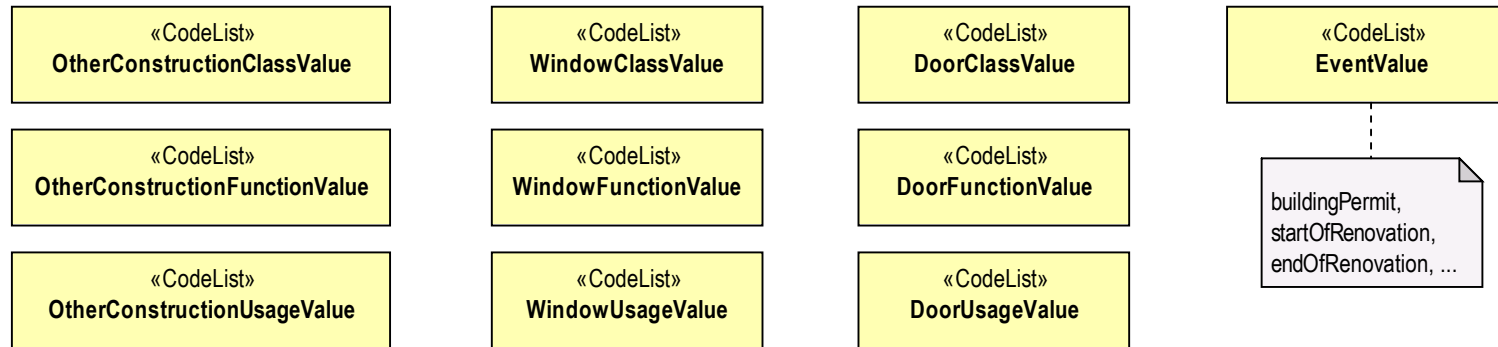
«CodeList»

**CityObjectGroupUsageValue**

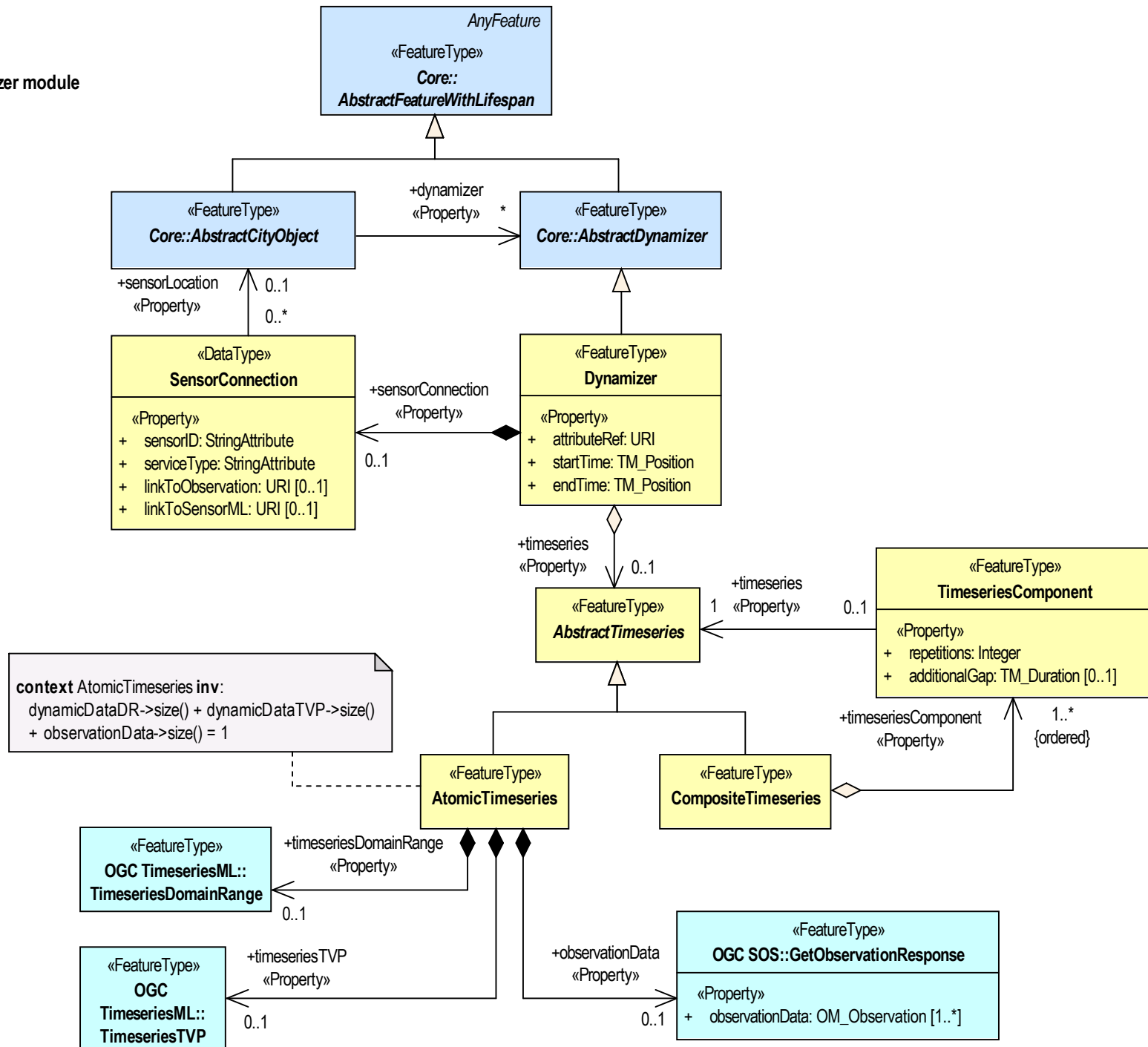
Construction module

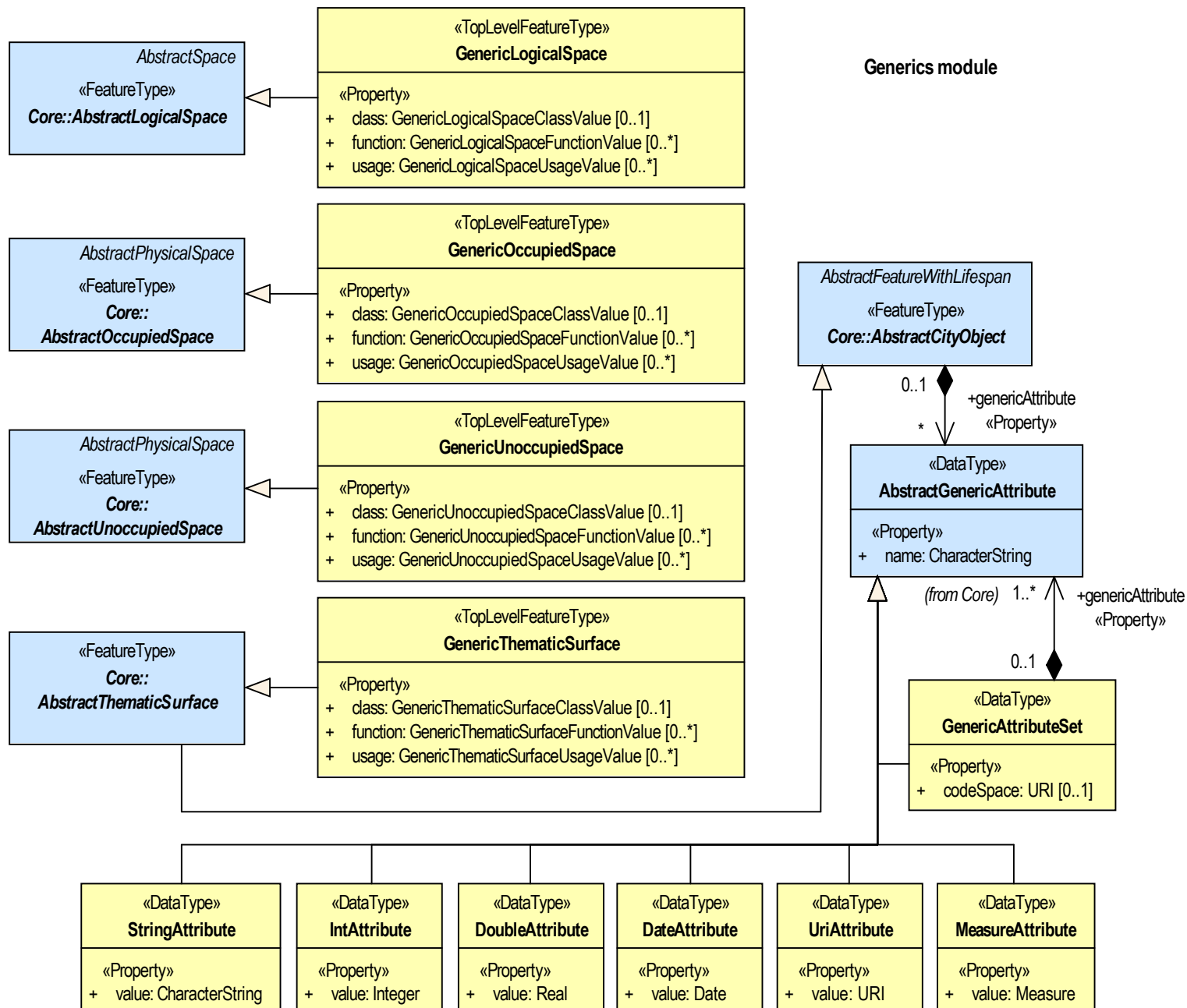


### Construction module - Code lists



## Dynamizer module





Generics module - Code lists

«CodeList»  
**GenericLogicalSpaceClassValue**

«CodeList»  
**GenericOccupiedSpaceClassValue**

«CodeList»  
**GenericUnoccupiedSpaceClassValue**

«CodeList»  
**GenericThematicSurfaceClassValue**

«CodeList»  
**GenericLogicalSpaceFunctionValue**

«CodeList»  
**GenericOccupiedSpaceFunctionValue**

«CodeList»  
**GenericUnoccupiedSpaceFunctionValue**

«CodeList»  
**GenericThematicSurfaceFunctionValue**

«CodeList»  
**GenericLogicalSpaceUsageValue**

«CodeList»  
**GenericOccupiedSpaceUsageValue**

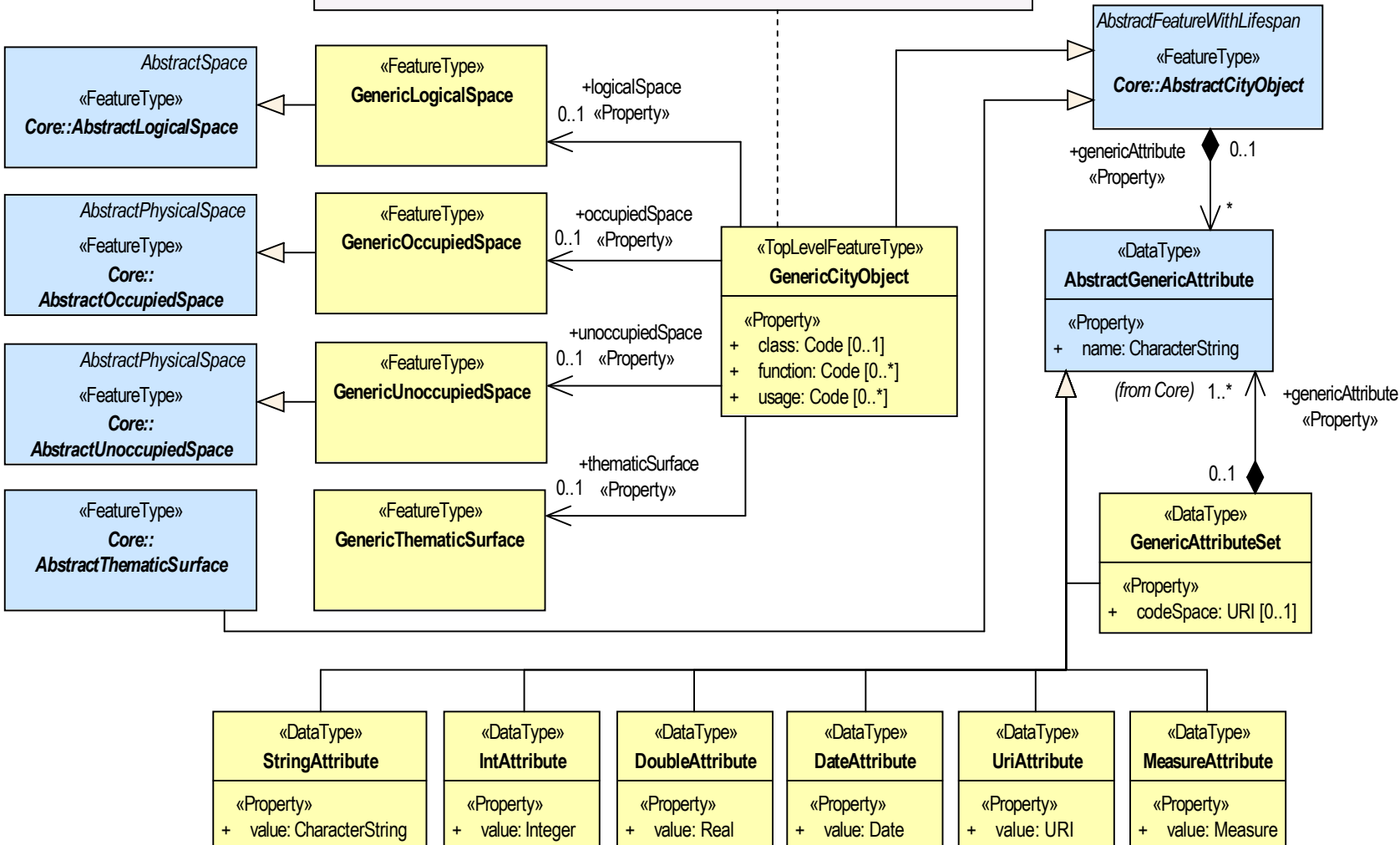
«CodeList»  
**GenericUnoccupiedSpaceUsageValue**

«CodeList»  
**GenericThematicSurfaceUsageValue**

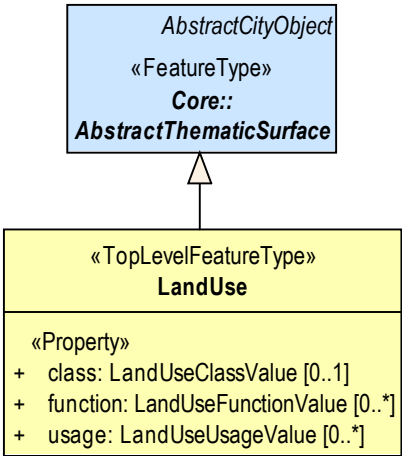
**context** GenericCityObject inv:  
 logicalSpace->size() + occupiedSpace->size()  
 + unoccupiedSpace->size() + thematicSurface->size() = 1

An instance of **GenericCityObject** can only be associated either with one instance of one of the Space classes or with one instance of **GenericThematicSurface**.  
 If several city objects are to be modelled at the same time, CityObjectGroup should be used.

## Generics module - Alternative I



LandUse module





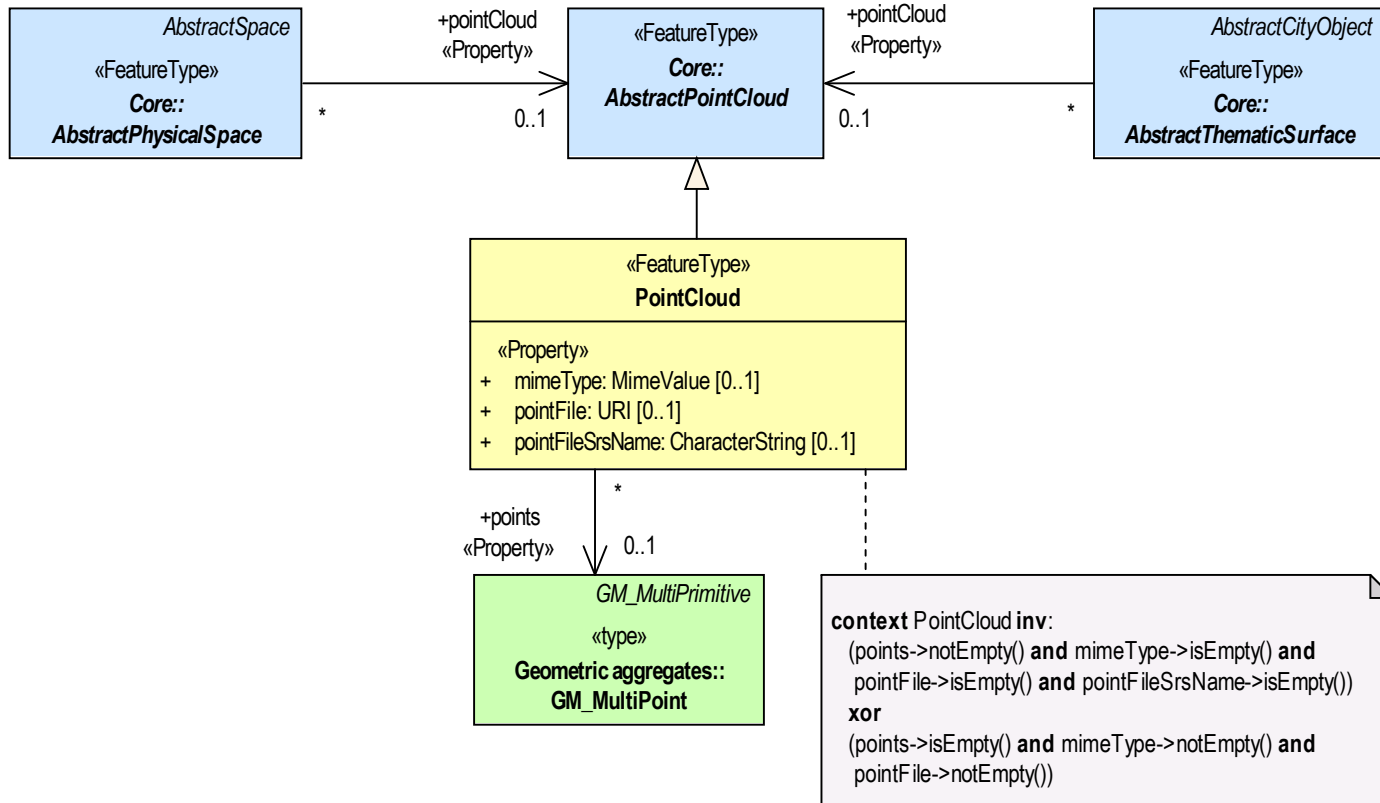
## LandUse module - Code lists

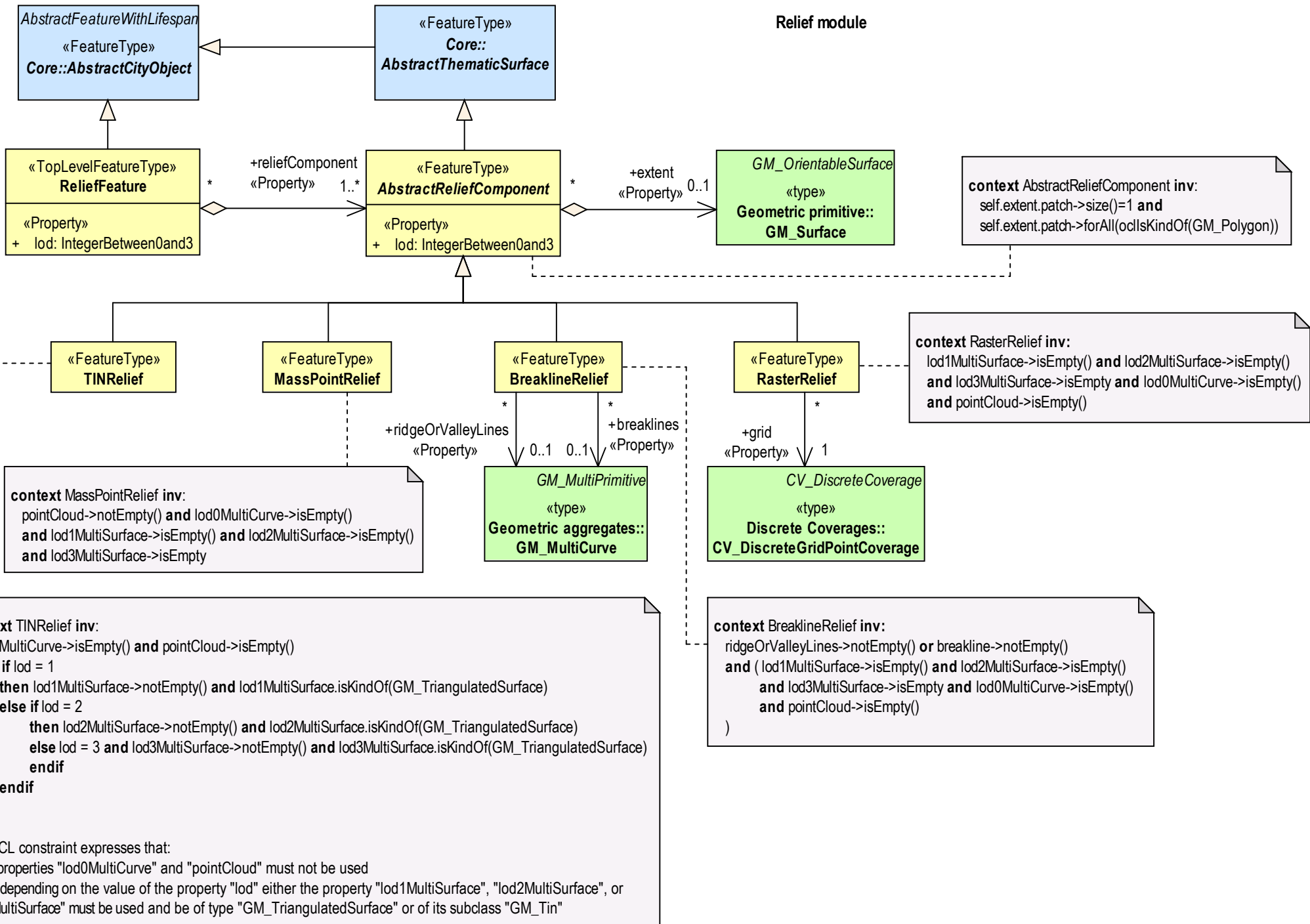
«CodeList»  
**LandUseClassValue**

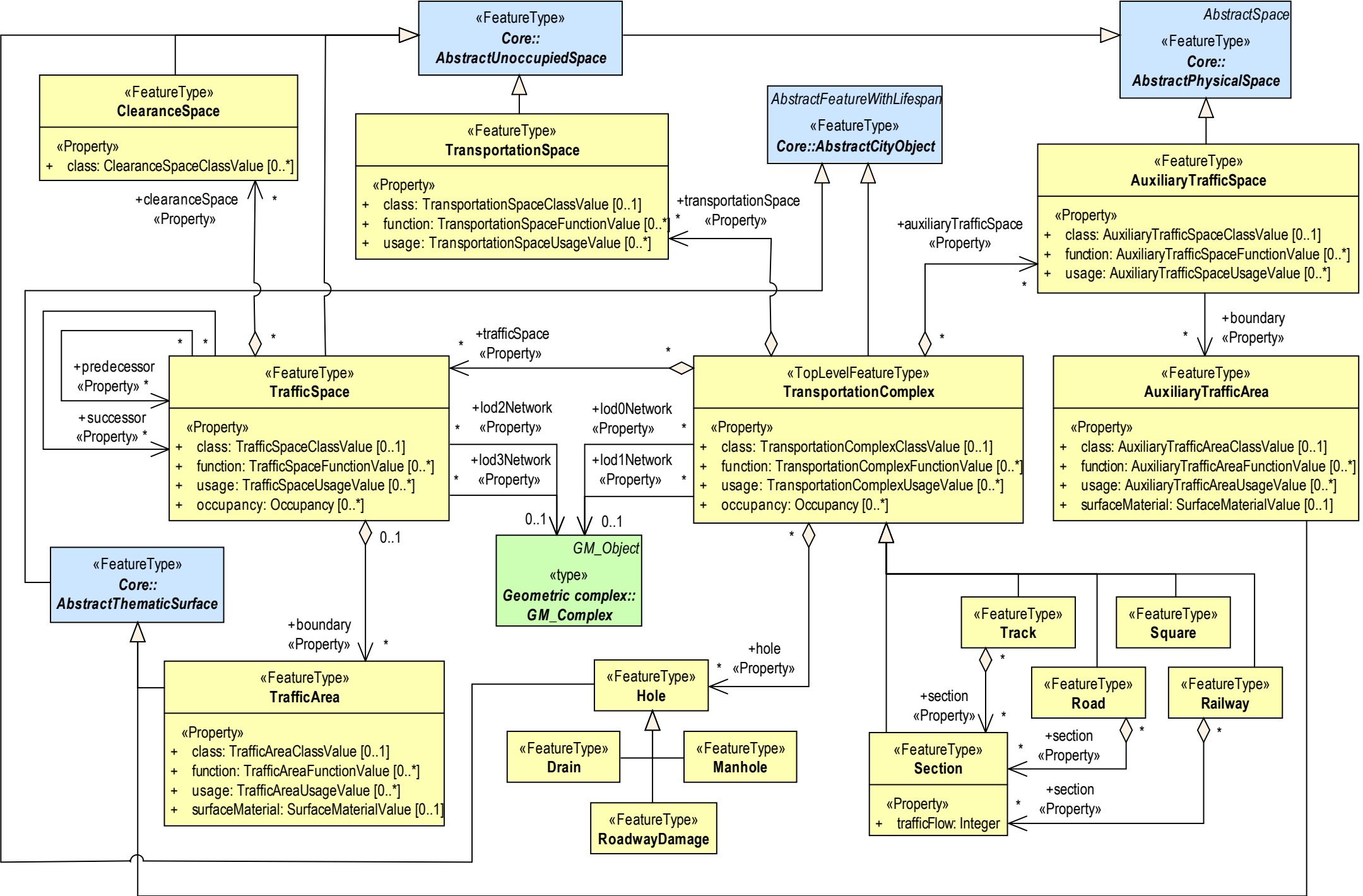
«CodeList»  
**LandUseFunctionValue**

«CodeList»  
**LandUseUsageValue**

## PointCloud module



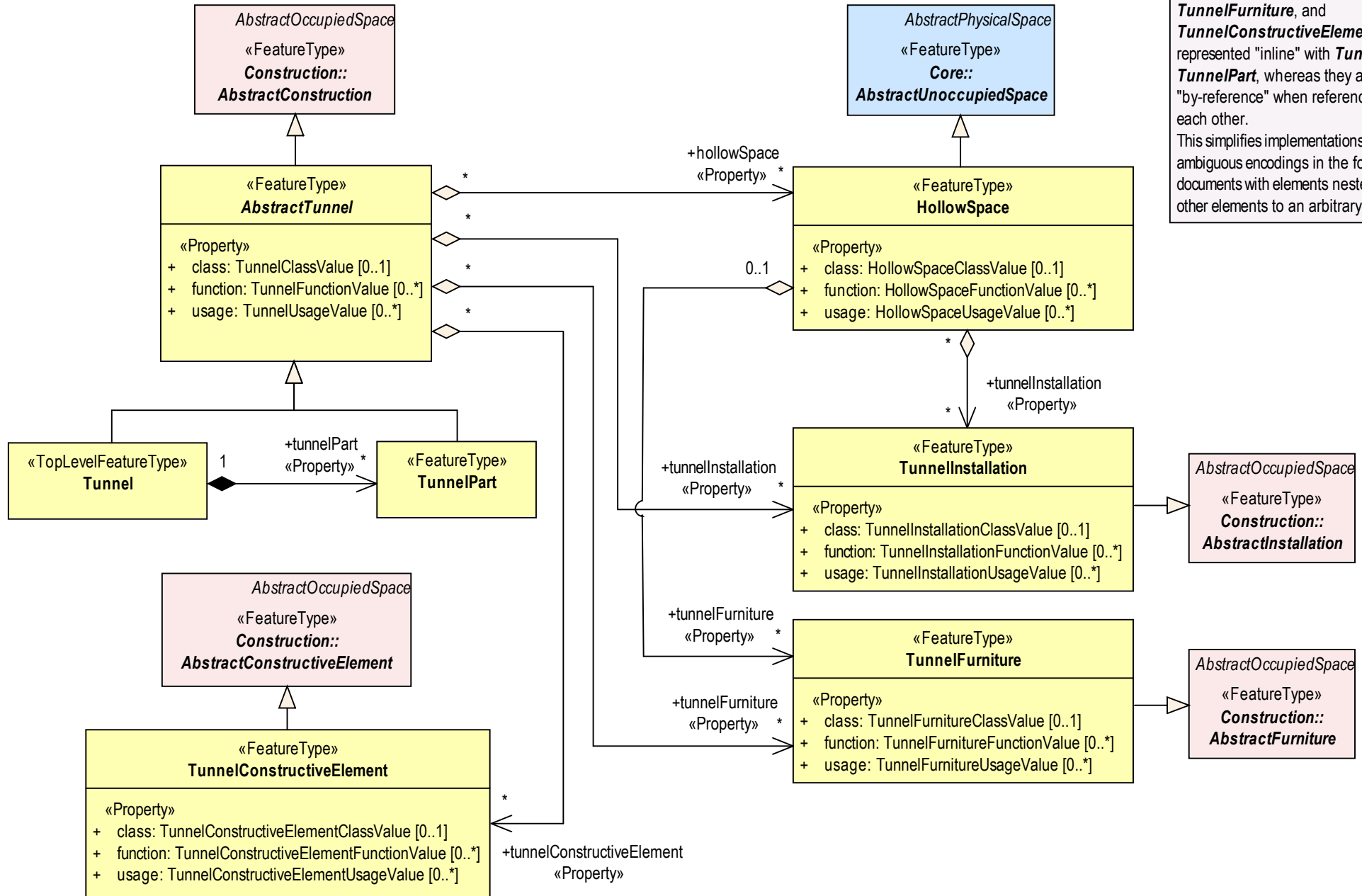




## Transportation module - Code lists

«CodeList» <b>TransportationComplexClassValue</b>	«CodeList» <b>TransportationSpaceClassValue</b>	«CodeList» <b>TrafficSpaceClassValue</b>	«CodeList» <b>TrafficAreaClassValue</b>
«CodeList» <b>TransportationComplexFunctionValue</b>	«CodeList» <b>TransportationSpaceFunctionValue</b>	«CodeList» <b>TrafficSpaceFunctionValue</b>	«CodeList» <b>TrafficAreaFunctionValue</b>
«CodeList» <b>TransportationComplexUsageValue</b>	«CodeList» <b>TransportationSpaceUsageValue</b>	«CodeList» <b>TrafficSpaceUsageValue</b>	«CodeList» <b>TrafficAreaUsageValue</b>
«CodeList» <b>AuxiliaryTrafficSpaceClassValue</b>	«CodeList» <b>AuxiliaryTrafficAreaClassValue</b>	«CodeList» <b>ClearanceSpaceClassValue</b>	«CodeList» <b>SurfaceMaterialValue</b>
«CodeList» <b>AuxiliaryTrafficSpaceFunctionValue</b>	«CodeList» <b>AuxiliaryTrafficAreaFunctionValue</b>		
«CodeList» <b>AuxiliaryTrafficSpaceUsageValue</b>	«CodeList» <b>AuxiliaryTrafficAreaUsageValue</b>		

## Tunnel module



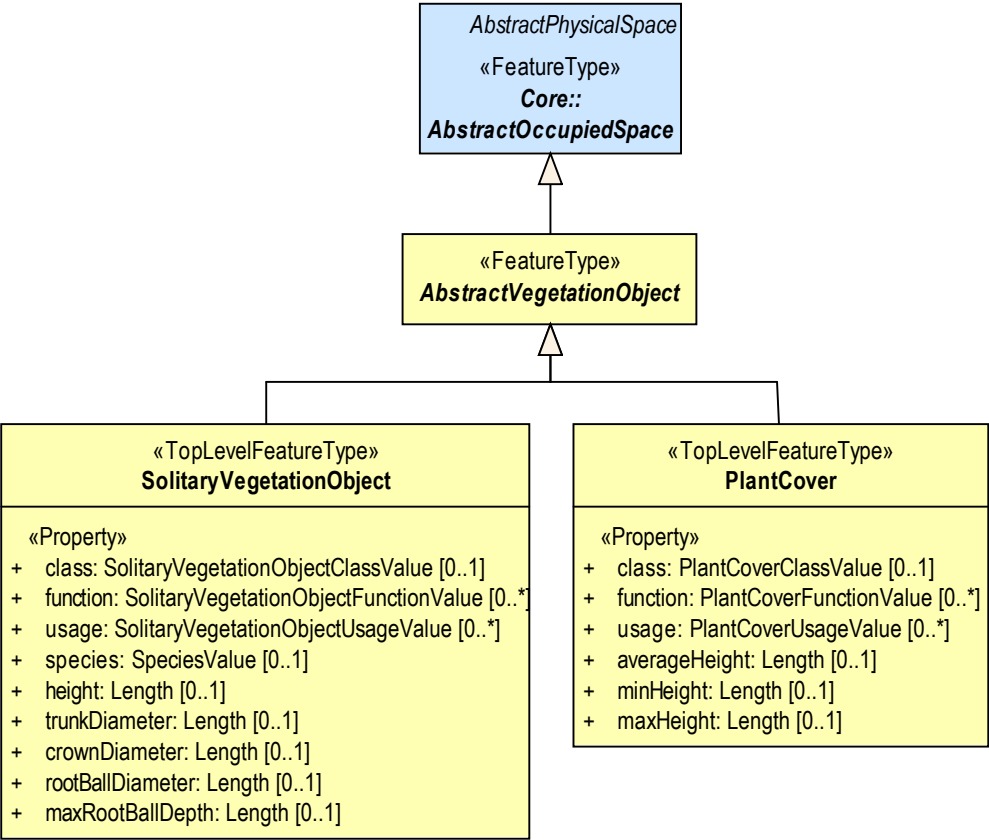
**HollowSpace**, **TunnelInstallation**, **TunnelFurniture**, and **TunnelConstructiveElement** are always represented "inline" with **Tunnel** and **TunnelPart**, whereas they are represented "by-reference" when referenced amongst each other.

This simplifies implementations as it avoids ambiguous encodings in the form of instance documents with elements nested inline of other elements to an arbitrary depth.

## Tunnel module - Code lists

«CodeList» <b>TunnelClassValue</b>	«CodeList» <b>TunnelInstallationClassValue</b>	«CodeList» <b>TunnelFurnitureClassValue</b>
«CodeList» <b>TunnelFunctionValue</b>	«CodeList» <b>TunnelInstallationFunctionValue</b>	«CodeList» <b>TunnelFurnitureFunctionValue</b>
«CodeList» <b>TunnelUsageValue</b>	«CodeList» <b>TunnelInstallationUsageValue</b>	«CodeList» <b>TunnelFurnitureUsageValue</b>
«CodeList» <b>HollowSpaceClassValue</b>	«CodeList» <b>TunnelConstructiveElementClassValue</b>	
«CodeList» <b>HollowSpaceFunctionValue</b>	«CodeList» <b>TunnelConstructiveElementFunctionValue</b>	
«CodeList» <b>HollowSpaceUsageValue</b>	«CodeList» <b>TunnelConstructiveElementUsageValue</b>	

Vegetation module





## Vegetation module - Code lists

«CodeList»  
**SolitaryVegetationObjectClassValue**

«CodeList»  
**SolitaryVegetationObjectFunctionValue**

«CodeList»  
**SolitaryVegetationObjectUsageValue**

«CodeList»  
**SpeciesValue**

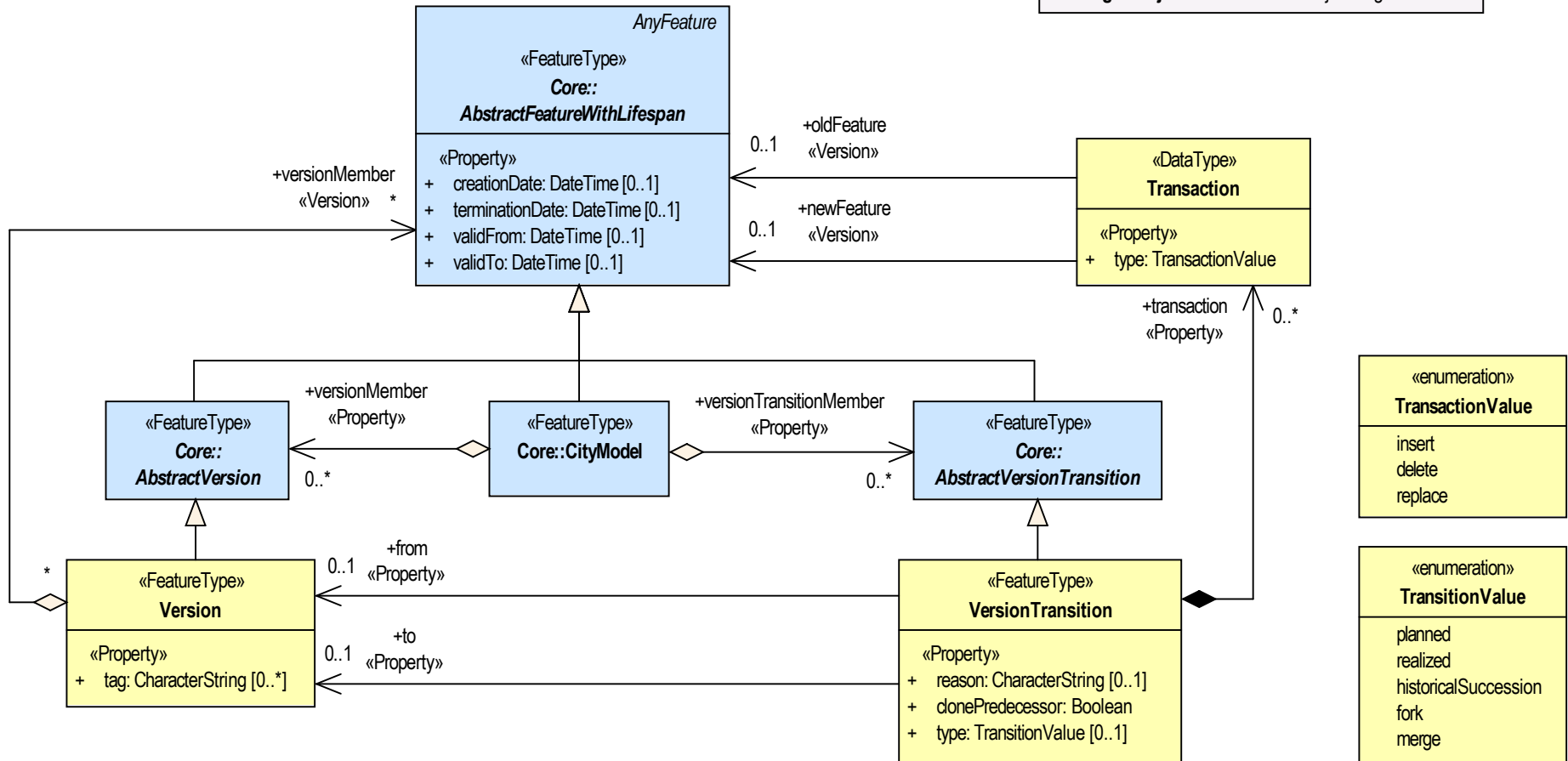
«CodeList»  
**PlantCoverClassValue**

«CodeList»  
**PlantCoverFunctionValue**

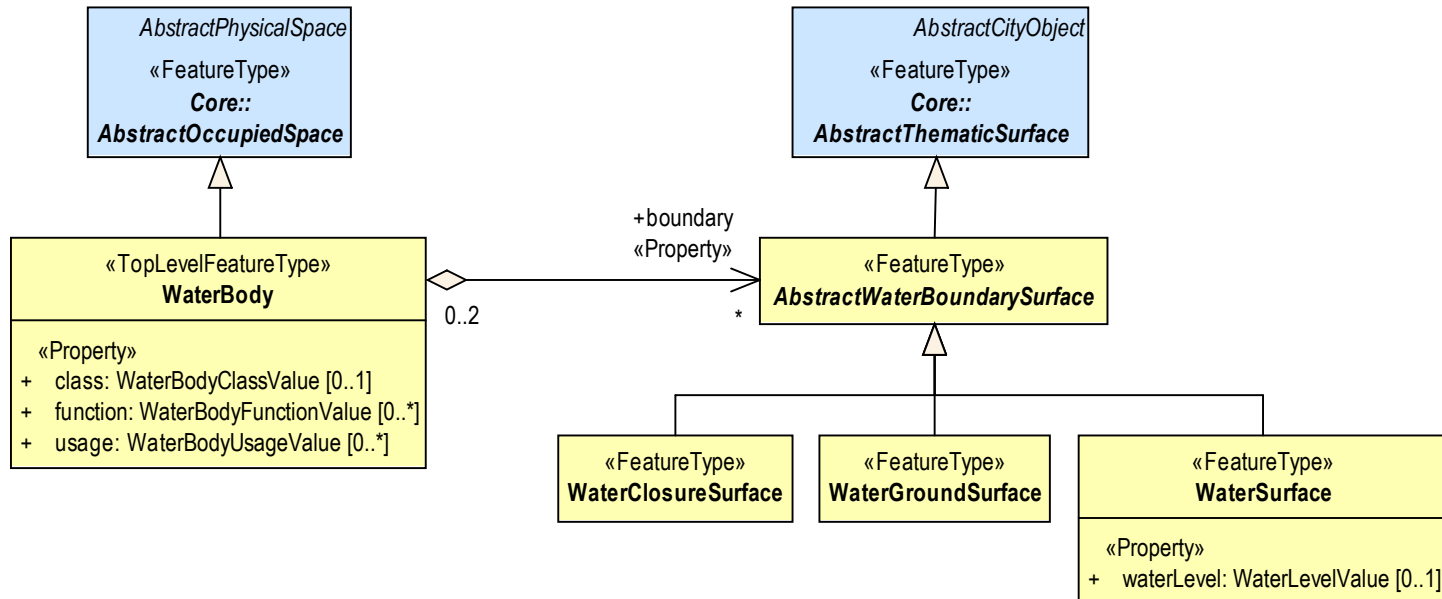
«CodeList»  
**PlantCoverUsageValue**

## Versioning module

The stereotype «Version» is adopted from INSPIRE. The stereotype is used for association roles to express that the **association refers to a specific version of the target object** and not to the object in general.



## WaterBody module



WaterBody module - Code lists

«CodeList»  
WaterBodyClassValue

«CodeList»  
WaterBodyFunctionValue

«CodeList»  
WaterBodyUsageValue

«CodeList»  
WaterLevelValue