

IFC Infra Overall Architecture Project

Documentation and Guidelines

Authors: André Borrmann (Project Lead), Julian Amann, Tim Chipman, Juha Hyvärinen, Thomas Liebich, Sergej Muhič, Laura Mol, Jim Plume, Paul Scarponcini

Status: FINAL (01/03/2017)

Content

1.	Introduction and Overview	3
2.	Spatial Structure.....	5
3.	Positioning and Geometry Representation	6
3.1	Geodetic reference systems	7
3.2	Terrain.....	9
3.3	Alignment & Positioning	11
3.4	String Lines Representation	15
3.5	Cross Section Representation	16
3.6	Surface Representation.....	20
3.7	Solid Geometry	22
3.8	Railway cant	24
4.	Physical element description	28
5.	Classification, Property Sets and linkage with object type libraries	32
5.1	Classification	32
5.2	Property Sets.....	34
5.3	Linkage with the buildingSMART Data Dictionary and Object Type Libraries	35
5.4	Linked Data approaches.....	37
6.	Summary and Outlook	39
	Annex 1: Spatial Structure – Explanatory Text	40
	Annex 2: Comparison OGC LandInfra and LandXML	52
	Annex 3: Element breakdown.....	53

1. Introduction and Overview

The IFC-Infra overall architecture project was initiated to provide a common basis for the upcoming projects for extending IFC for the infrastructure domain, including IFC-Road, IFC-Rail and IFC-Bridge.

It was created in response to the observation that national initiatives in the field had created diverging approaches in a number of aspects that require a unified process in order to avoid inconsistencies in the IFC data model. The project was set up to provide recommendations for developing extensions and to provide basic data structures mandatory for all infrastructure extension. The project was carried out in parallel with the IFC-Alignment 1.1 project; resulting in strong synergies and continuous synchronization of the developments. The project was also conducted in close collaboration with OGC in order to realize a harmonized conceptual model as a common basis for both IFC-Infra and InfraGML. This harmonized conceptual model will facilitate integration and conversion between both buildingSMART and OGC upcoming standards.

The recommendations for using IFC and the guidelines for further extension are documented in this buildingSMART Technical Report. The recommendations include:

- Spatial Structure (Section 2);
- Geometry representations (Section 3);
- Element breakdown structure (Section 4);
- Classification and linked data (Section 5).

The IFC-Alignment and the IFC-Overall Architecture project have also developed data structures and implementation guidelines, including:

- Alignment and Positioning (in collaboration with the IFC-Alignment project);
- Geometry representations:
 - StringLine representation;
 - CrossSection representation;
 - Surface representation;
 - Solid representation;
- Terrain (Triangulated Irregular Network).

These data structures are going to be published as IFC4.1 RC3 and will undergo the standardization process. If they are adopted by buildingSMART International, they will provide a sound foundation for specific extensions to be developed for roads and railways and be released as the official IFC4.1 release. This release will form the basis for the ongoing IFC Alignment Deployment project, where different teams of software developers and key users test the implementation of the new data structures in real use cases. Both, the RC3 and the final release are made available through buildingSMART-tech.org.

The developments have been based on high-priority use cases, which have been identified in the course of this project by means of international surveys among stakeholders in the infrastructure

domain. The use cases are documented in the report “Requirements Analysis” published on 29/06/2016.

The project team applied the following general principles during the development of recommended extensions and strongly recommend the use of these general principles in upcoming IFC extensions, such as IFC-Bridge, IFC-Road and IFC-Rail:

- minimal intervention: ensure downwards compatibility to the greatest extent possible;
- minimal extension: use existing data structure to the greatest extent possible;
- international scope: the data model should only contain elements with global validity. Seeing as the IFC standard already offers very strong extension mechanisms, such as property sets and classifications, for modeling national and/or regional concepts.

Not only does this Technical Report provide recommendations on geometry representation, but also on spatial (Section 2) and element breakdown structure (Section 4). An infrastructure project can be broken down in different manners. In general, the IFC data model provides the following breakdown structure (Figure 1):

- spatial breakdown structure
- element breakdown structure
- system breakdown structure

This document provides recommendations regarding the spatial breakdown structure (Section 2) and the element breakdown structure (Section 4).

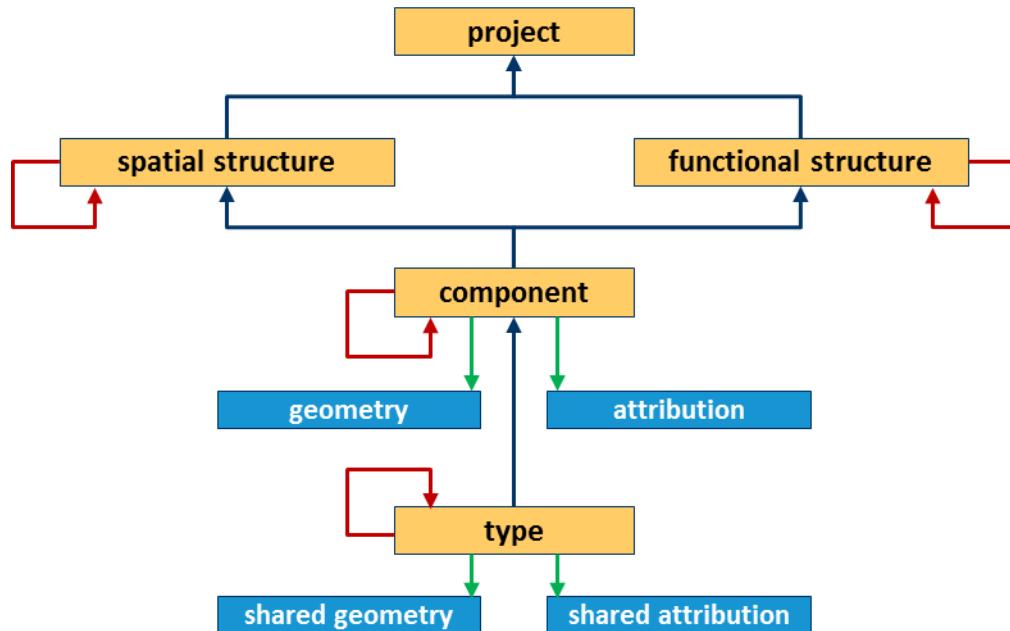


Figure 1: IFC breakdown structures

2. Spatial Structure

The spatial hierarchy in IFC creates a framework for spatial positioning of physical entities within a piece of infrastructure. For example, it allows a section of road or railway (or any physical feature, such as a drain or light pole) to be associated with a named section of that piece of infrastructure. This facilitates asset management and permits a meaningful spatial organization of an IFC model.

The following figure shows the set of spatial structure elements that may be used in an IFC model file or database to establish a spatial hierarchy for a project. Prior to the introduction of the new spatial structure elements for infrastructure (shaded in yellow in Figure 2), the spatial hierarchy (going from high to low level) was composed of *ifcSite*, *ifcBuilding*, *ifcBuildingStorey* and *ifcSpace* (shaded in green).

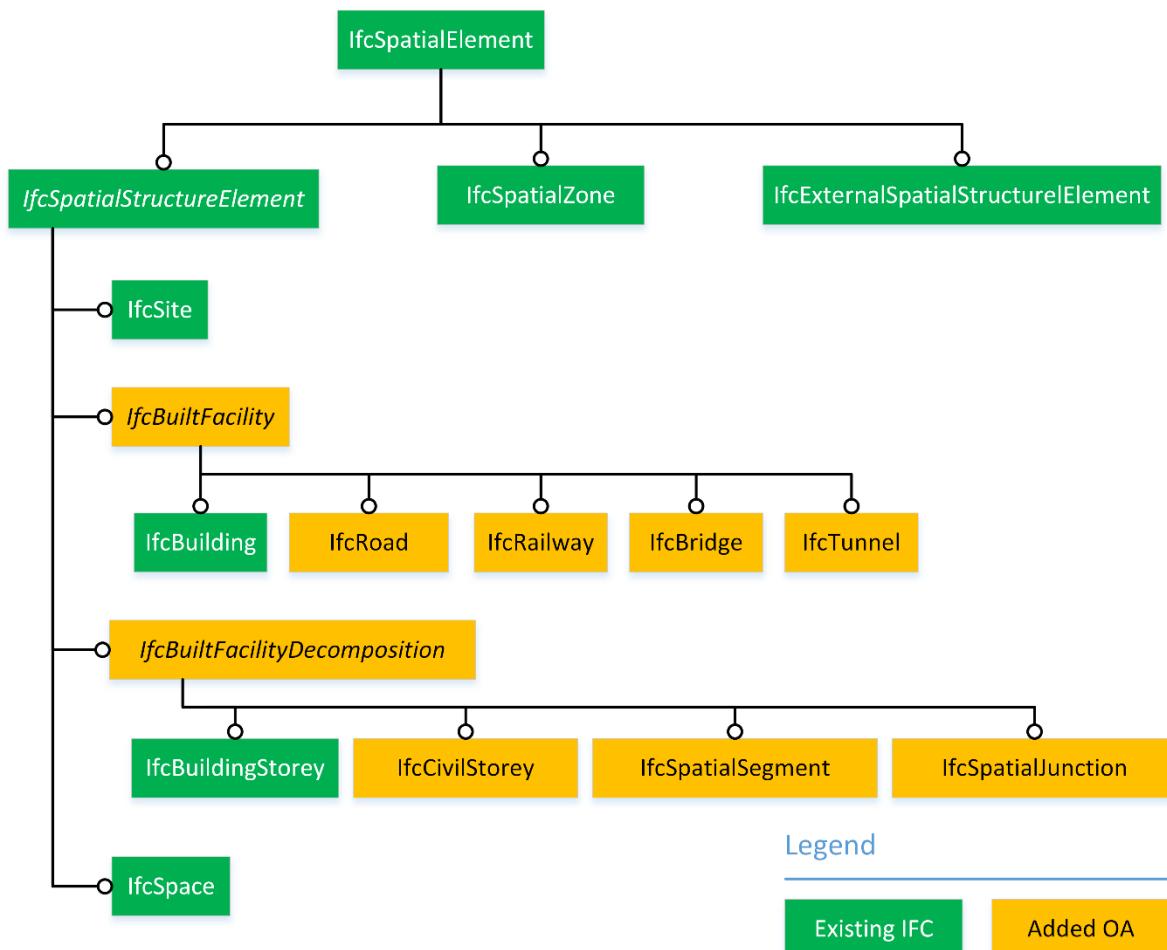


Figure 2: Proposed changes to the Spatial Structure to support Infrastructure

The spatial structure hierarchy proposed here to support infrastructure follows the same principles as applied in previous releases of IFC, ensuring that a project defined using these concepts is fully

backward compatible. The introduction of new abstract superclasses (*ifcBuiltFacility* and *ifcBuiltFacilityDecomposition*) maintains the four levels of decomposition, while permitting the introduction of new classes of decomposition at those two levels to accommodate infrastructure entities.

Consistent with current IFC specification, the top of a spatial hierarchy is always *ifcProject*, and the only “required” spatial entity is at least one instance of any subclass of the abstract superclass *ifcBuiltFacility*. This conforms with current practice where a minimal spatial hierarchy would consist of an *ifcProject*, consisting at least one instance of *ifcBuilding*, with no *ifcSite* entity and no further decomposition.

The spatial hierarchy for any project is defined using the objectified relationship, *ifcRelAggregates*. This relationship has two mandatory attributes, the first specifying the *RelatingObject* (the parent being decomposed) and the second being an unordered list of *RelatedObjects* (being the set of children that decompose the aggregated parent).

This proposed spatial organization is explained in further detail in Annex 1: Spatial Structure – Explanatory Text. This annex outlines the principles around embedding this spatial hierarchy within the context of the existing IFC data model and provides several examples of how it may be applied to specific instances of roads, railways, bridges and tunnels.

The design of the spatial structure is not yet finished and requires further discussion and experimentation with real-world infrastructure projects. This is supposed to be performed in the upcoming IFC-Bridge, IFC-Rail and IFC-Road projects and that a final agreement on a spatial structure for infrastructure project is added to the task list of the upcoming “Common Definitions” project. In the meantime, the authors recommend to use only one IfcSite spatial object and avoid any spatial breakdown structure in infrastructure IFC instance files.

3. Positioning and Geometry Representation

Linear infrastructure facilities can be described by one or multiple of the following geometry representations:

- String Lines Representation (Section 3.4);
- Cross Sections Representation (Section 3.5);
- Surface Representation (Section 3.6);
- Solid Representation (Section 3.7):
 - explicit boundary representation;
 - cross-section sweep representation.

The geometry representations are described in more detail in the indicated Sections.

The flexibility in geometric representation allows support for a multitude of uses cases, including (but not limited to):

- Quantity Take-off;

- Clash Detection;
- Design-Intent transfer;
- Code Compliance Checking;
- GIS-BIM data transfer.

Figure 3 illustrates the relationship between the use cases, identified during and within the scope of this project, and their required geometry representation. As a general rule a singular use case will require a specific geometry representation.

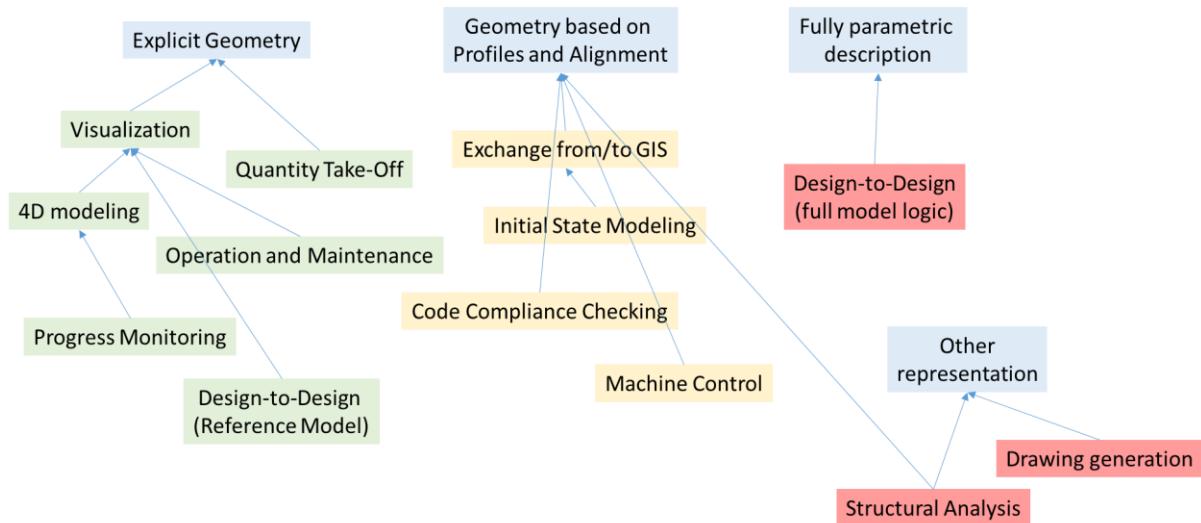


Figure 3: Relationship between use cases and required geometry representation

Placement of objects can be achieved by either:

- combining geometry representations with linear referencing related to the underlying alignment (Section 3.3),
- or by applying a Cartesian coordinate system (local or global).

In both cases, the geodesic reference system needs to be specified (Section 3.1). It is the basis of the underlying (local) engineering coordinate system of the infrastructure works.

Each infrastructure element (a road shoulder, a bridge deck, etc.) can be described by several geometric representations, such as an explicit geometry and an axis plus cross section. If multiple geometry representations of one object are provided in one IFC instance file, the writing application must ensure their consistency.

3.1 Geodetic reference systems

In large infrastructure projects the specification and consistent use of the geodetic reference system (GRS) is of utmost importance. To this end, the IFC standard provides the entity *IfcCoordinateReferenceSystem*, which allows for the following information to be defined:

- Name: Name by which the coordinate reference system is identified. The name shall be taken from the list recognized by the European Petroleum Survey Group (EPSG).
- GeodeticDatum: Name by which this datum is identified. The geodetic datum is associated with: firstly, the coordinate reference system and indicates the shape and size of the rotation ellipsoid; and secondly, this ellipsoid's connection and orientation to the actual globe and/or earth. Examples of a geodetic datum are: ED50, EUREF89, WSG84
- VerticalDatum: Name by which the vertical datum is identified. The vertical datum is associated with the height axis of the coordinate reference system and indicates the reference plane and fundamental point defining the origin of a height system (often mean sea level). Examples: AHD, ODN.

While *IfcCoordinateReferenceSystem* is an abstract class, its subclass *IfcProjectedCRS* is used to define a concrete projected coordinate reference system.

- MapProjection: Name by which the map projection is identified. Examples: UTM, Gauss-Krueger
- MapZone: Name by which the map zone is identified, which also relates to the MapProjection. Based on the above-mentioned MapProjection examples, the following MapZones can be identified: first, for UTM the zone number 32S is represented as UTM32 South; second, for Gauss-Krueger the zones of longitudinal width are represented as 3'.
- MapUnit: Unit of the coordinate axes composing the map coordinate system.

The projected coordinate reference system is assumed to be a 2D or 3D right-handed Cartesian coordinate system, the optional MapUnit attribute can be used to determine the length unit used by the map.

A typical instance in STEP P21 format could look like this:

```
#17=IFCPROJECTEDCRS ('EPSG:31467', 'EPSG:31467 - DHDN / 3-Degree Gauss-Krueger Zone 3', 'EPSG:31467', $, 'Gauss-Krueger', '3', #18);
```

The IFC standard does not provide the calculations needed to correctly map to and from a Cartesian coordinate system into a geodesic coordinate system. The writing and/or reading application is responsible for this mapping, including the transformation of lengths.

In infrastructure design and surveying, mostly map grid coordinates (easting, northing) are used for digital representations, such as (3461404.34 , 5483498.73). In GML-based data formats (such as CityGML or InfraGML) these full coordinates are stored directly into the GML-file. BIM tools available at this date might encounter problems when handling these large coordinates. In order to prevent this issues, the IFC standard allows for referencing of the locally defined coordinate system to the global coordinate system. The advantage of using a local coordinate system is that it allows the use of shorter specified coordinates.

The use of a local coordinate system and applying a transformation is not mandatory, if original map grid coordinates are preferred, they can also be represented and exchanged using the IFC data schema. Attention has to be paid to the fact that the units for the map coordinates for x and y

(Eastings and Northings) are still Geodetic Coordinates and may involve a non-zero combined (height) scale factor which can vary from point to point.

The entity *IfcMapConversion* is used for defining a local coordinate system with respect to the global coordinate system. The map conversion allows for: firstly, the origin of the local engineering coordinate system to be related to its place within a map (easting, northing, and orthogonal height); and secondly, rotation of the x-axis of the local engineering coordinate system within the horizontal plane of the map (easting and/or westing).

Furthermore, the entity *IfcMapConversion* provides the attributes *SourceCRS* and *TargetCRS*. By convention, a coordinate operation is given between the *SourceCRS* and the *TargetCRS*, in which the *SourceCRS* is the more local, or child coordinate reference system, and the *TargetCRS* is the more remote, or parent coordinate reference system. Typically, an *IfcGeometricRepresentationContext* entity will act as the *SourceCRS*. The *IfcProject* entity of the IFC model will then refer to this *IfcGeometricRepresentationContext* by means of its *RepresentationContexts* attribute.

The definition of a geodetic reference system is optional, but highly recommend in the infrastructure context. Though it is possible to define more than one geodesic reference system within an IFC model, it is not recommended to do so as the definition of multiple GRS in one model has not been implemented or tested thus far and is expected to be handled incorrectly by writing and/or reading applications.

3.2 Terrain

An important aspect of infrastructure modeling is the description of terrain, including pre- and post-construction surfaces. To capture and describe the terrain surface, a new entity *IfcTriangulatedIrregularNetwork* has been introduced as a subclass of *IfcTriangulatedFaceSet*.

How the triangles in a TIN are generated varies between software products so recording the final triangles is the only way of guaranteeing that the TIN can be exactly transferred between software products. *IfcTriangulatedIrregularNetwork* efficiently carries sufficient information to reconstruct equivalent surfaces in various software products, including every 3D point and triangular face. Such surfaces will return the same elevations and ground profiles to the functions that depend on them. It differs from the OGC approach, reflecting the difference in use cases. Whereas GML3.3 has to consider exchanges where no triangulation has been done, IFC covers exchange cases where triangulated surfaces have already been generated. Nevertheless, it is possible to attach multiple shape representations to the terrain (*IfcGeometricElement* of type **TERRAIN**) where missing data (*TINElement* of type "hole" in GML3.3) can be represented by polygons instead of fully triangulated surfaces. Breaklines are generally not supposed to be instantiated since they are captured in the *IfcTriangulatedIrregularNetwork*. Another notable difference is that in IFC triangles are represented as an indexed set, where the indices refer to existing points, compared to each triangle storing all the coordinates in the GML3.3 schema. Consequently, the point instances can be referenced by various other entities.

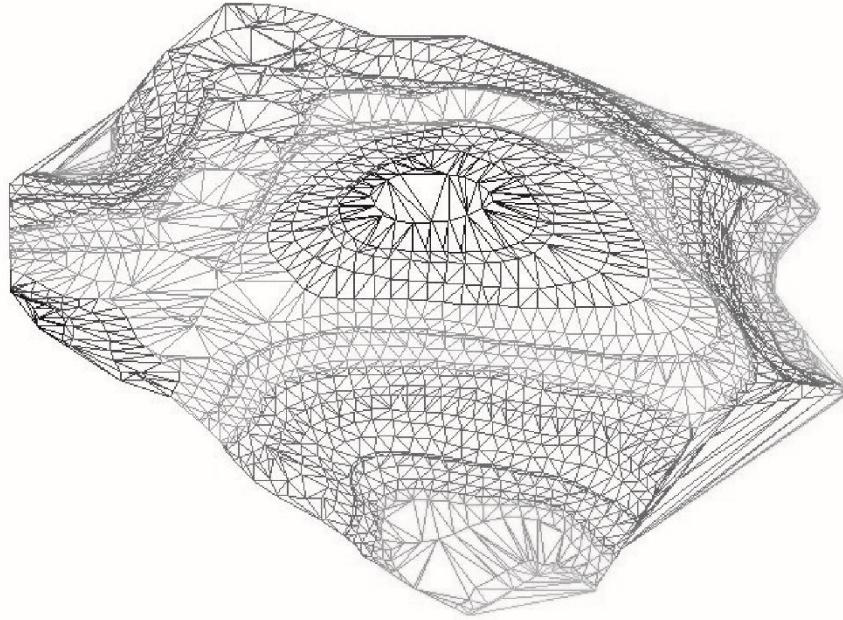


Figure 4: Digital Terrain Model represented by *IfcTriangulatedIrregularNetwork*

IfcTriangulatedIrregularNetwork derives from *IfcTriangulatedFaceSet*. In addition to a list of 3D points and an indexed set of triangular faces (coming from the aforementioned *IfcTriangulatedFaceSet*), *IfcTriangulatedIrregularNetwork* also carries for each face an integer (from a list of predefined values) representing a break line and visibility information. For visible faces, the break line data simplifies assignment of materials and contour smoothing. The following values are featured:

- 0 – no break lines,
- 1 – break line at edge 1,
- 2 – break line at edge 2,
- 3 – break line at edges 1 and 2,
- 4 – break line at edge 3,
- 5 – break line at edges 1 and 3,
- 6 – break line at edges 2 and 3,
- 7 – break line at edges 1, 2 and 3.

OGC's "void" and "hole" concepts for invisible faces are also supported in IFC. The corresponding integer values assigned to these values are:

- -2 – invisible void,
- -1 – invisible hole.

"Void" may be assigned to faces within the outline of a building, for instance. These faces will never return elevations or show contours. "Hole" denotes missing data, which may be supplemented if this surface is merged onto another larger one. Supplying "hole" faces in *IfcTriangularIrregularNetwork* is

optional; the import and verification process must check whether any additional padding “hole” faces are needed by the receiving software.

Additional rules for *IfcTriangularIrregularNetwork* that need to be verified:

- All the 3D points must be unique in the XY plane. [Should there be a minimum resolution, e.g. 0.5mm ?]
- All faces must have the same direction: clockwise or counterclockwise. A shared edge between two faces is indicated when each has the indexes of the same two points but in the opposite order.
- No face may overlap another in the XY plane. Boundaries of contiguous faces may touch at one or more common points, but they may not cross.

3.3 Alignment & Positioning

As previously described, the IFC-Overall Architecture and IFC Alignment 1.1 project have collaborated on the Alignment and Positioning. Positioning in infrastructure projects typically features linear referencing. Based upon the ISO standard 19148, the IFC Schema has been extended to incorporate linear referencing. Several new classes have been introduced into the existing *IfcAlignment* 1.0, though a more limited set of subtypes would typically be applicable for most uses cases.

The design representation with a horizontal (*IfcAlignment2DHorizontal*) and a vertical (*IfcAlignment2DVertical*) 2D curve defining a 3D curve geometry has been encapsulated in the entity *IfcAlignmentCurve* (see Figure 5). By definition, it is an *IfcCurve* and can therefore be reused for creating 3D geometry of elements by sweeping a cross-section along the corresponding *IfcAlignmentCurve* (see Section 3.5).

For describing spans that are subsets of an alignment curve (such as for a bridge along a road), and/or offset laterally and vertically from an alignment curve (such as for bridge girders or guardrails), the new entity *IfcOffsetCurveByDistances* has been defined. The existing *IfcPolyline* curve may be used for describing existing conditions where design information is not available (such as extracted from a geospatial database).

IfcReferent, a new entity deriving from *IfcPositioningElement*, was introduced for defining known positions on the alignment. It can represent anything from a kilo point, mile point, to a station.

Positioning along a linear positioning element can be achieved by *IfcLinearPlacement* (see Figure 5 entities added in IFC4x1 are marked in red). Deriving from *IfcObjectPlacement*, it allows for *IfcProduct* instances to be placed along an *IfcAlignment*. This is accomplished by defining the location, through setting the distance along and offset values from the alignment (*Distance* attribute), and orienting the element (*Orientation* attribute).

Distance and offsets are handled by *IfcDistanceExpression* that features the values for the distance along the alignment (*DistanceAlong*) and the three offsets from the alignment (*OffsetLongitudinal*, *OffsetLateral* and *OffsetVertical*). *OffsetLateral* is defined as the direction to the left of the alignment

and *OffsetVertical* as the normal direction from the alignment (in the plane and direction of global Z). *OffsetLongitudinal* is required in cases where the alignment curve has points of discontinuity. At these points the local coordinate system is defined by the curve that is terminated with the discontinuity.

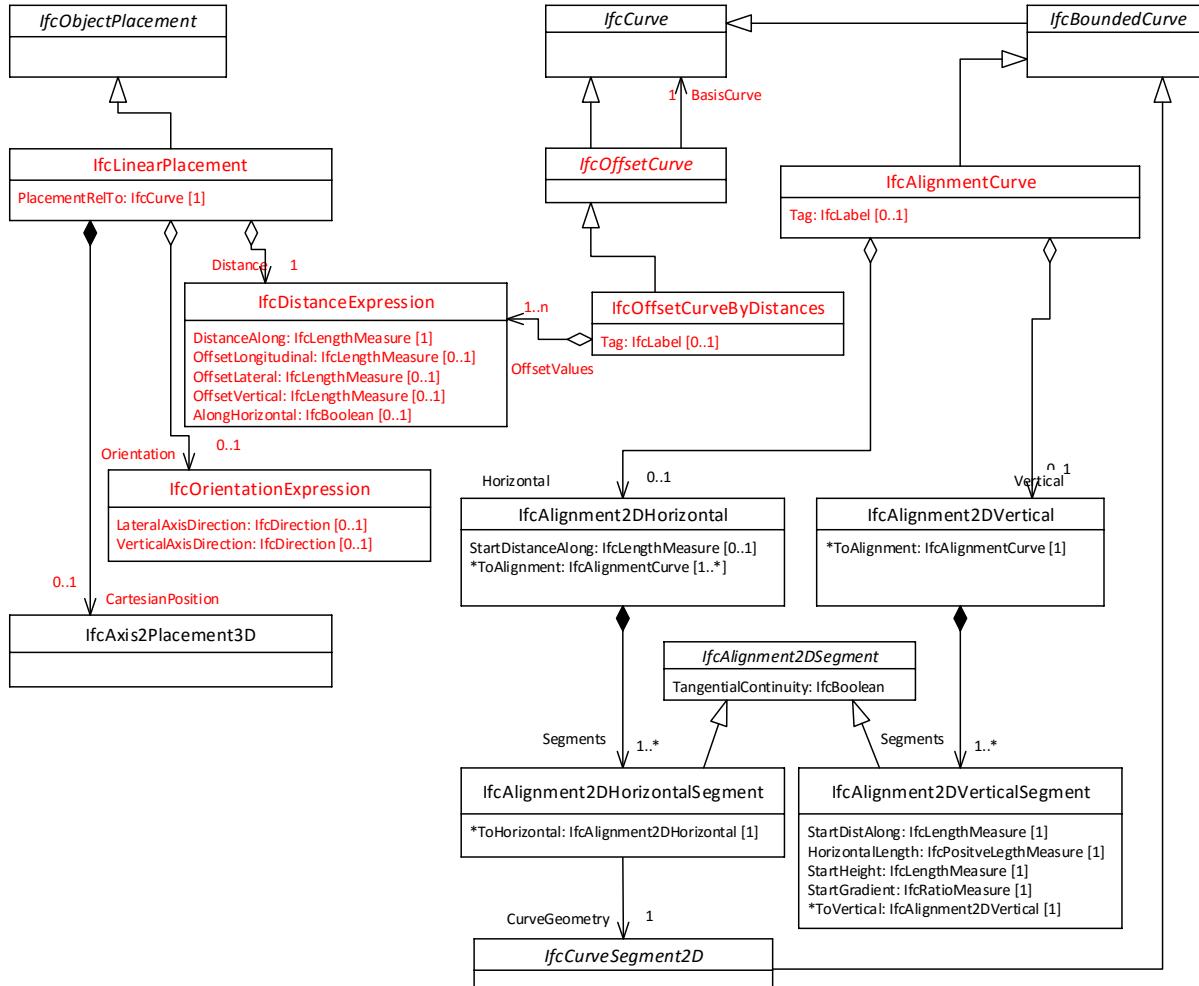


Figure 5: Class diagram depicting the extensions required for describing alignment and linear placement (re-colored classes are new)

The example in Figure 6 and Figure 7 shows a simple straight line geometry of the *IfcAlignmentCurve* where the horizontal alignment starts at the coordinates (0, 1000) and ends at (0, 1100) resulting in a segment which is 100 units long. The vertical segment length coordinate corresponds to the horizontal alignment (1000 and 1100 respectively) and the vertical coordinates start at 50 and end at 49 making for a 1.0% drop on the 100 units long segment. This *IfcAlignmentCurve* decomposition can be observed in Figure 6.

With the *IfcAlignmentCurve* geometry from Figure 6, an *IfcAlignment* is defined and can be seen in Figure 7. For demonstration purposes two *IfcReferent* entities are defined at the start and at the end of the *IfcAlignment*. Both have their own *IfcLinearPlacement* with an *IfcDistanceExpression*. Since *IfcReferent* cannot be oriented, the optional *OrientationExpression* is not populated. Additionally,

the *IfcReferent* is defined as a known location on the *IfcAlignment*, therefore all the offset values of *IfcDistanceExpression* are zero.

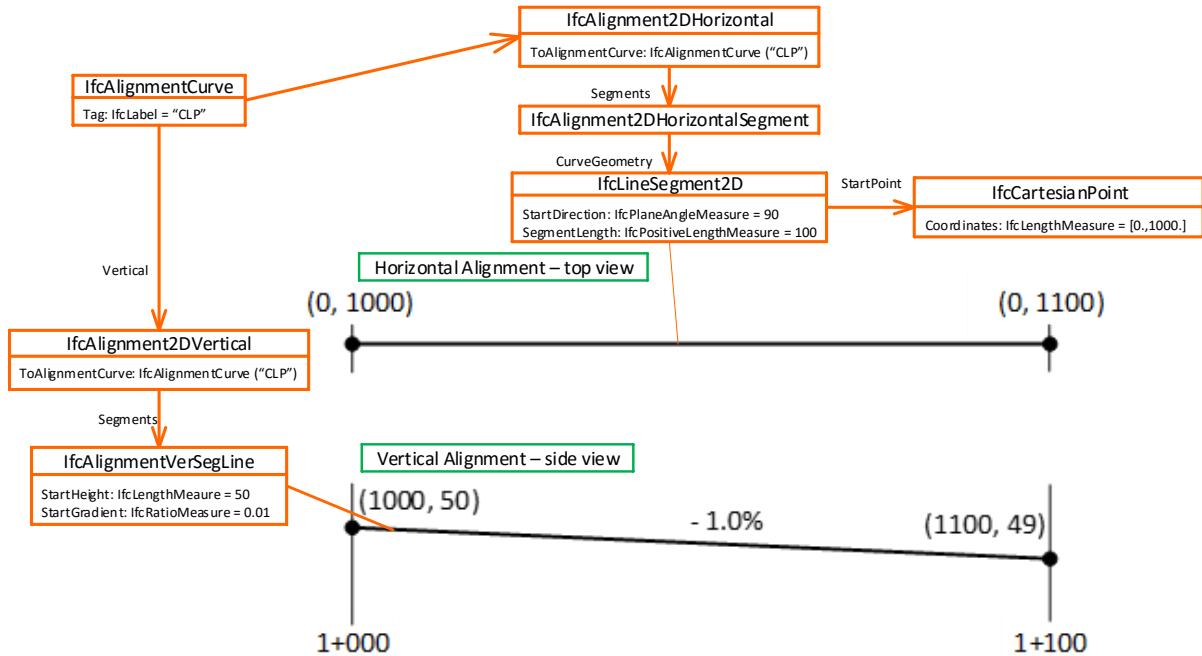


Figure 6: Decomposition of the *IfcAlignmentCurve* into horizontal (*IfcAlignment2DHorizontal*) and vertical (*IfcAlignment2DVertical*) alignment (based on OGC figure)

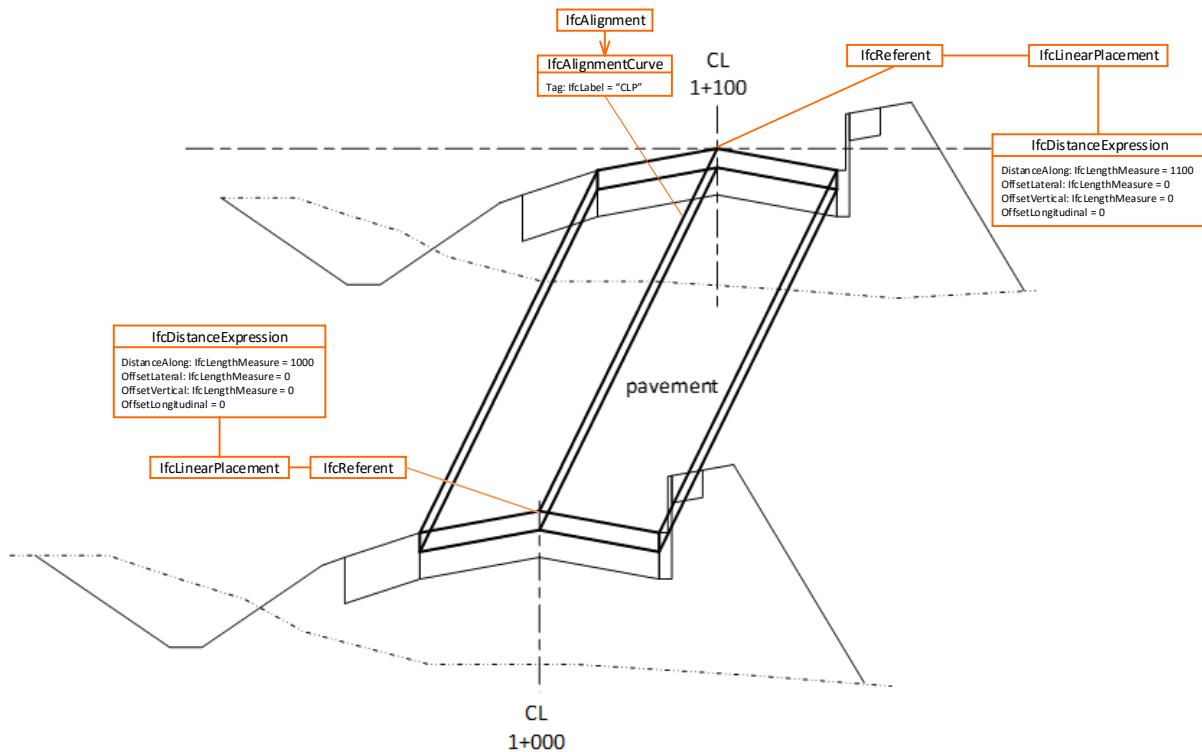


Figure 7: Use of *IfcReferent* and *IfcDistanceExpression* on the *IfcAlignmentCurve* from Figure 7 (based on OGC figure¹).

¹ All figures marked with OGC in this document originate from:
OGC 16-104 InfraGML Part 4 - LandInfra Roads- Encoding Standard (proposed)

3.4 String Lines Representation

String Line representation allows for infrastructure facilities to be represented by (multiple) 3D lines. This allows for unambiguous modeling of edges of different elements or layers, in particular for more complex cases such as turn-offs and junctions. These lines can be composed by any of the available 3D curves provided in the current IFC standard (e.g. *IfcPolyline*), or described by combining separated vertical and horizontal curves (*IfcAlignmentCurve*), or derivative curves according to span and offsets (*IfcOffsetCurveByDistances*), which provides a list of offset coordinates relative to another curve. Such offset curves may be used to position and size objects at lateral offsets such as guardrails, and may be constant or vary linearly along the referenced curve.

Figure 8 provides a simple example where three *IfcPolyline* objects are used to describe the geometry of a road segment: center line (CL); left-edge of pavement (LEP); and right-edge of pavement (REP). To assign a code or classification to a curve, the classification approach described in Section can be applied.

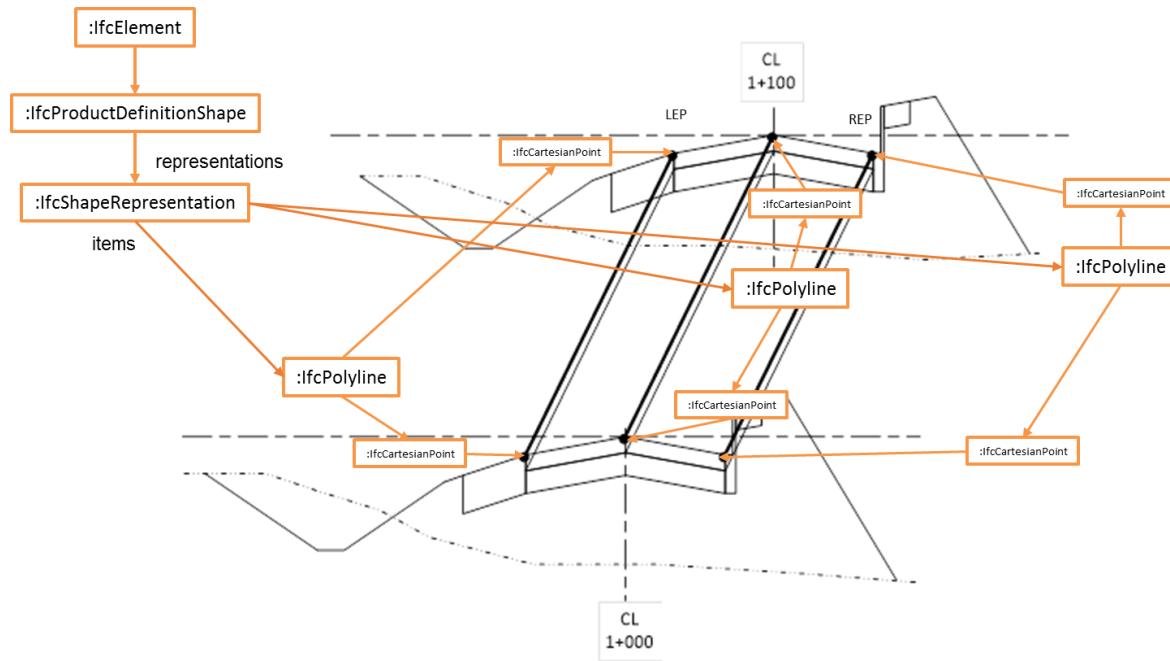


Figure 8: Instance diagram depicting the String Line representation of a road segment (based on OGC figure)

Figure 9 depicts the same example, but now with using offset curves to describe left-edge pavement and right-edge pavement. The offsets are defined by a coordinate list: the first coordinate describes the distance along the alignment; the second coordinate describes the offset in vertical direction; and the third coordinate describes the offset in vertical direction.

Each string line may optionally be named and/or classified using *IfcShapeAspect* – now expanded to support geometric representations, see Section 5.1 – to enable association with points of cross-sections to be positioned relative to one or more string lines.

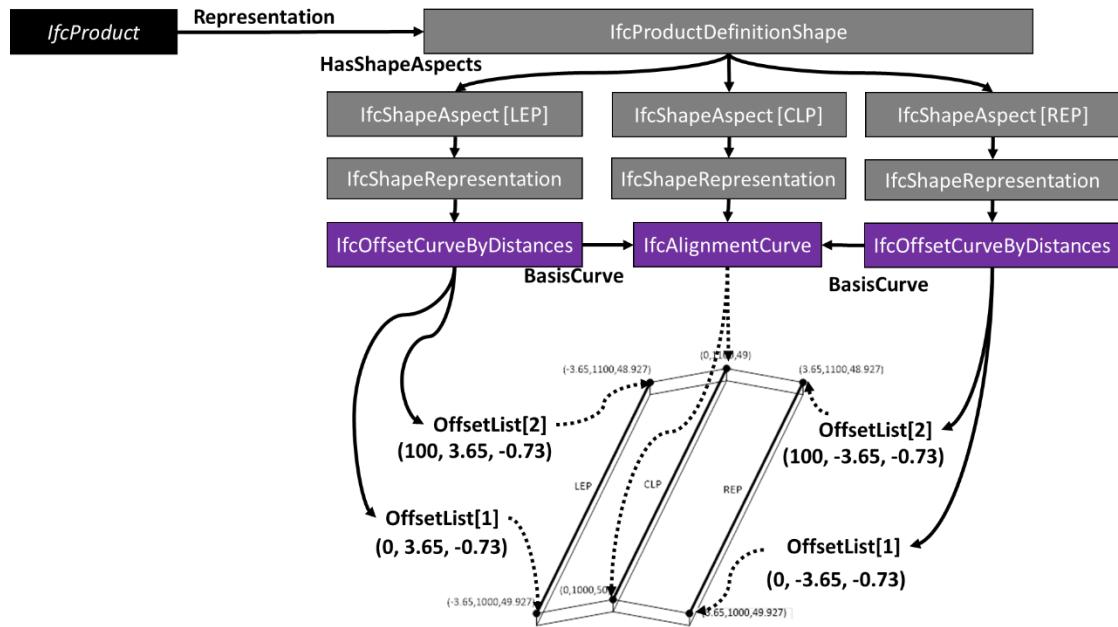


Figure 9: Use offset curves to describe the stringline geometry of a road segment (based on OGC figure).

3.5 Cross Section Representation

Infrastructure facilities can also be described by a number of cross-sections. These cross-section may form the basis for creating solid geometry by sweeping along the alignment (see Section 3.7).

Cross sections may be represented by existing definitions, including:

- closed areas (*IfcArbitraryClosedProfileDef*) for constructing solids;
- open curves (*IfcArbitraryOpenProfileDef*) for constructing surfaces;
- common parameterized forms (e.g. *IfcShapeProfileDef*, *IfcRectangleProfileDef*);
- and rotated or transformed derivatives (e.g. *IfcDerivedProfileDef*, *IfcMirroredProfileDef*).

Individual points of profiles may be uniquely identified using *IfcIndexedPolycurve* and *IfcCartesianPointList2DLabelled*. Doing so allows for profile vertices to be correlated with the corresponding string lines.

Figure 10, Figure 11 and Figure 12 illustrate cross sections of a road at two positions, where each shaded region may be described using *IfcArbitraryClosedProfileDef*. The overall cross-section may be defined at a specific position along an alignment curve (*IfcReferent*) using the *IfcRelAssociatesMaterial* relationship linking to *IfcMaterialProfileSet*. This *IfcMaterialProfileSet* contains an *IfcMaterialProfile* instance for each defined area of the cross-section, which may link to *IfcArbitraryClosedProfileDef* for closed regions.

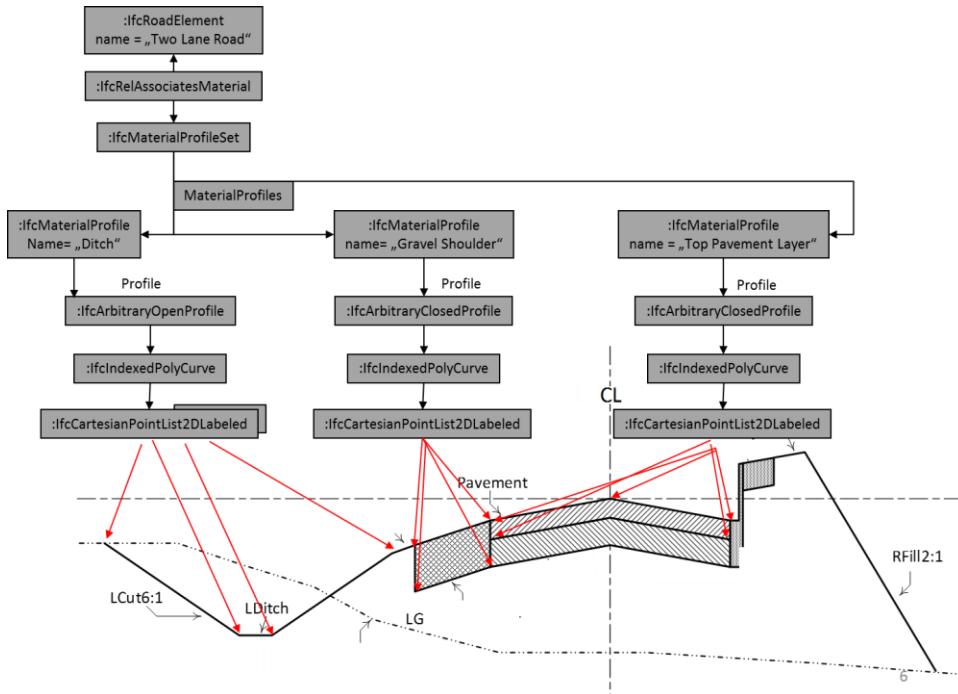


Figure 10: Cross-section of a road, including labels of individual edges and points,
Version 1: One *IfcElement* with multiple *IfcMaterialProfiles* (based on OGC figure)

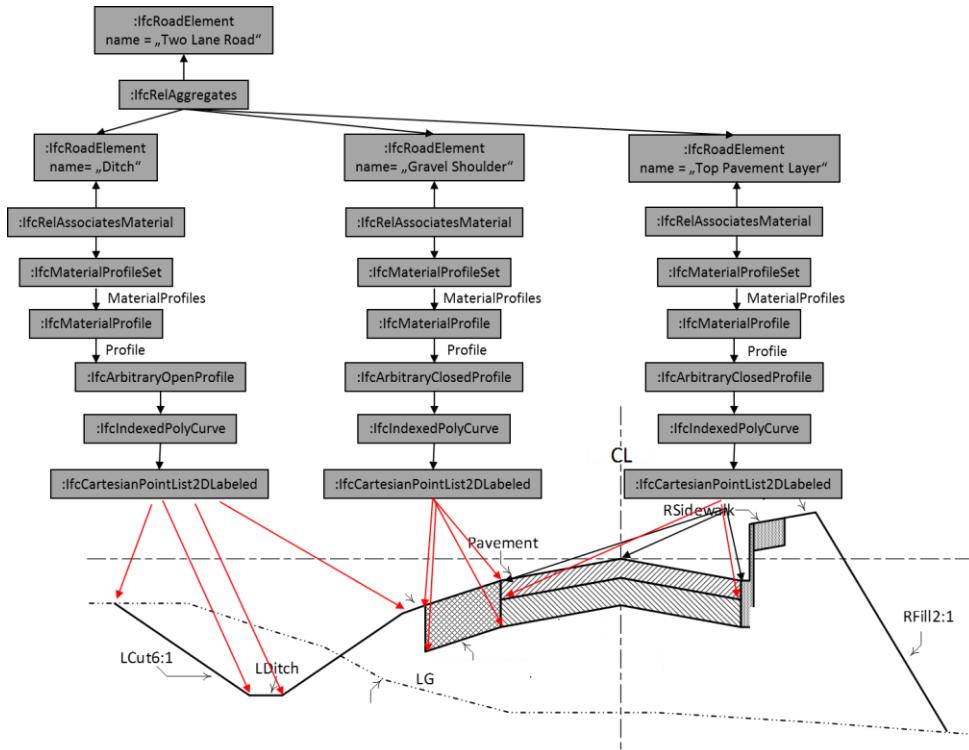


Figure 11: Cross-section of a road, including labels of individual edges and points,
Version 2: Multiple *IfcElements*, each having one *IfcMaterialProfile* (based on OGC figure)

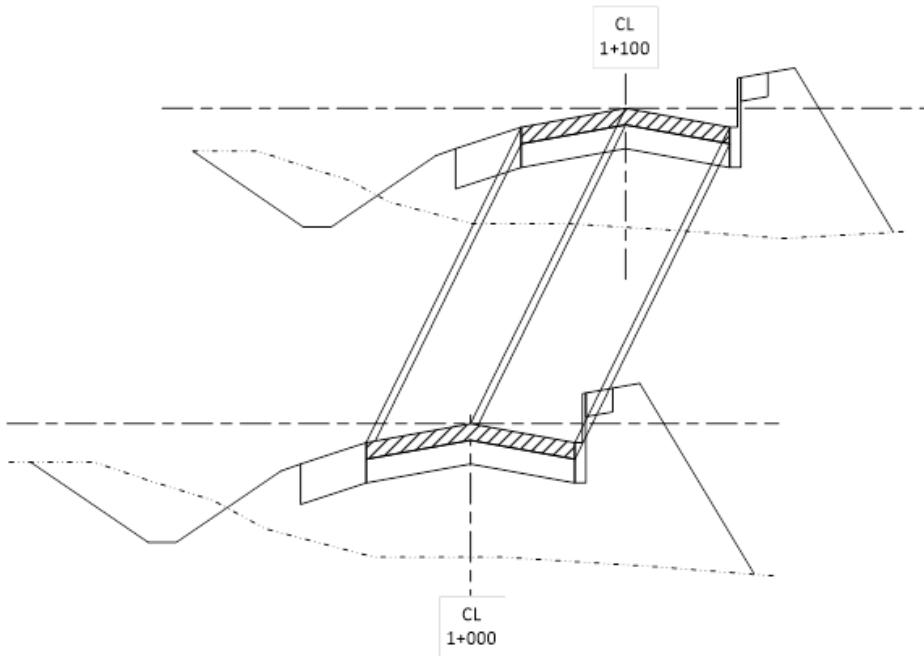


Figure 12: Cross section representation of a road segment (Courtesy: OGC)

Figure 13 illustrates design plans for a representative bridge having:

- a deck where the cross section (*IfcArbitraryClosedProfileDef*) is defined by lines and arcs (*IfcIndexedPolyCurve*);
- named vertices (*IfcCartesianPointList3DLabelled*), which may be correlated with string lines (*IfcOffsetCurveByDistances*) to form resulting geometry.

In this particular case, cross sections may be defined in such a way for preliminary design, or as standard templates provided by transportation agencies to be used on multiple projects.

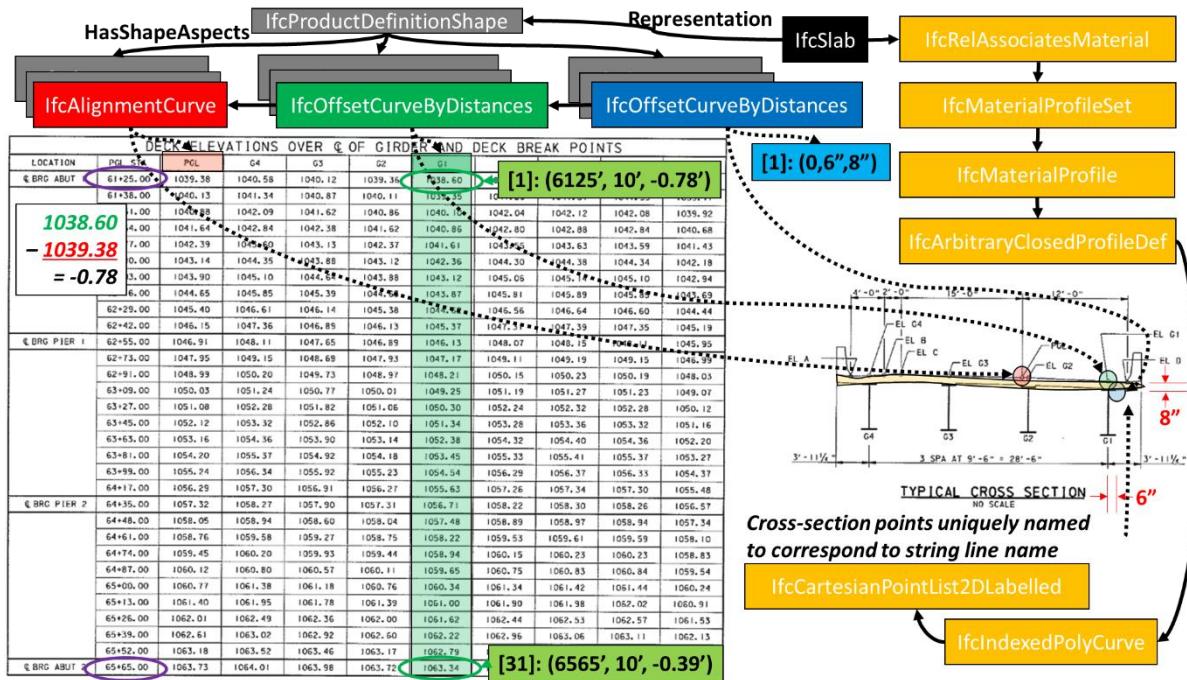


Figure 13: Cross section correlated with string lines by using identical names / labels (PGL, G1, G2, G3, G4 etc.)²

² Courtesy U.S. Federal Highway Administration, Pennsylvania Department of Transportation

3.6 Surface Representation

For specific use cases, such as visualization, it may be sufficient to describe the top surface of each model element. This is a very common approach in Geographic Information Systems (GIS) and related standards. Existing IFC entities can be used for surface representation. These existing entities include *IfcSurface* and the inherent subclasses.

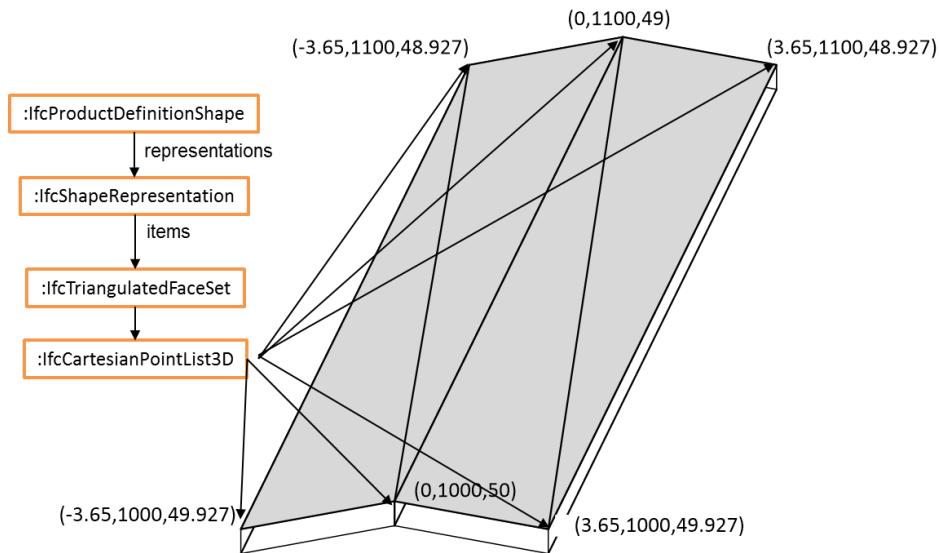


Figure 14: Using a triangulated face set for representing the surface of a road.

In most cases of surface representation the entity *IfcTriangulatedFaceSet* will be used. Figure 14 depicts a simple example illustrating the application of *IfcTriangulatedFaceSet*. This existing entity, introduced in the development of IFC4, can be used as-is. As depicted in Figure 15, in those cases where an infrastructure facility is composed out of multiple layers only the top surface of each layer is represented using the entity *IfcTriangulatedFaceSet*.

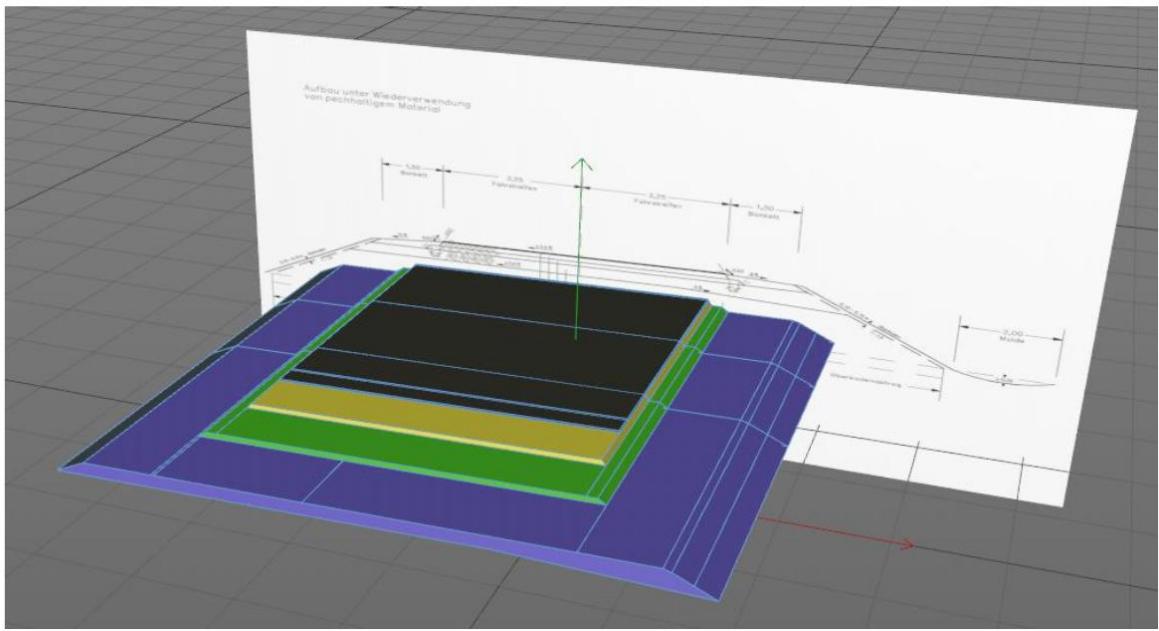
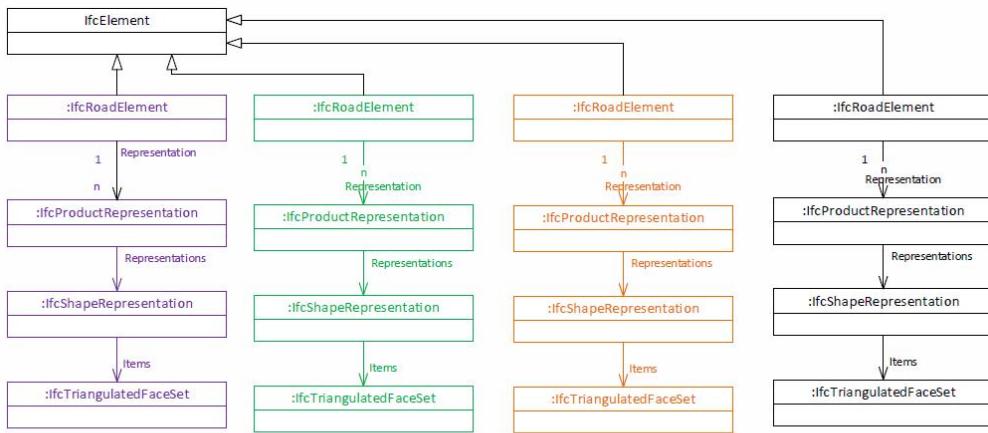


Figure 15: Surface representation: Each layer is represented by its top surface. Please note that the entity *IfcRoadElement* is not defined yet and only used as a placeholder here.

3.7 Solid Geometry

For representing solid geometry, the existing subclasses of IfcSolidModel can be applied. This includes explicit Boundary Representation (*IfcFacetedBRep*), CSG Geometry (*IfcCsgSolid*) or Sweeping-based Geometry (*IfcSweptAreaSolid*, *IfcSweptDiskSolid*).

To support geometry for roadways and bridge decks based on cross sections that conform to vertical/horizontal alignment curves and/or derivative string lines at varying super-elevation, a new geometry definition has been introduced called *IfcSectionedSolidHorizontal*. Conceptually similar to the existing *IfcSectionedSpine*, it allows multiple cross-sections to be defined at arbitrary intervals along a reference curve, with points of each cross section defined as lateral and vertical offsets relative to the reference curve.

IfcSectionedSolidHorizontal overcomes the limitations of *IfcFixedReferenceSweptAreaSolid* which only works with a single cross-section and rotates the cross-section in order to make it always being perpendicular to the *Directrix*. In the case of *IfcSectionedSolidHorizontal*, it is possible to set the attribute *FixedAxisVertical* to TRUE, meaning that only rotations in the x-y plane are applied, whereas the cross-section's up-vector remains equal to the z-axis of the containing coordinate system.

While *IfcSectionedSpine* is defined based on a Cartesian coordinate system, *IfcSectionedSolidHorizontal* is defined based on a coordinate system that is relative to the directrix where all points along the curve can be projected onto the horizontal plane. Vertical offsets may be defined as vertically perpendicular to the reference curve (such as for paved construction), or as vertically upwards opposite gravity (such as for poured construction). Common uses of this data structure include geometry with variable cross-sections for road pavement and bridge decks, and geometry with fixed cross-sections for girders and guardrails.

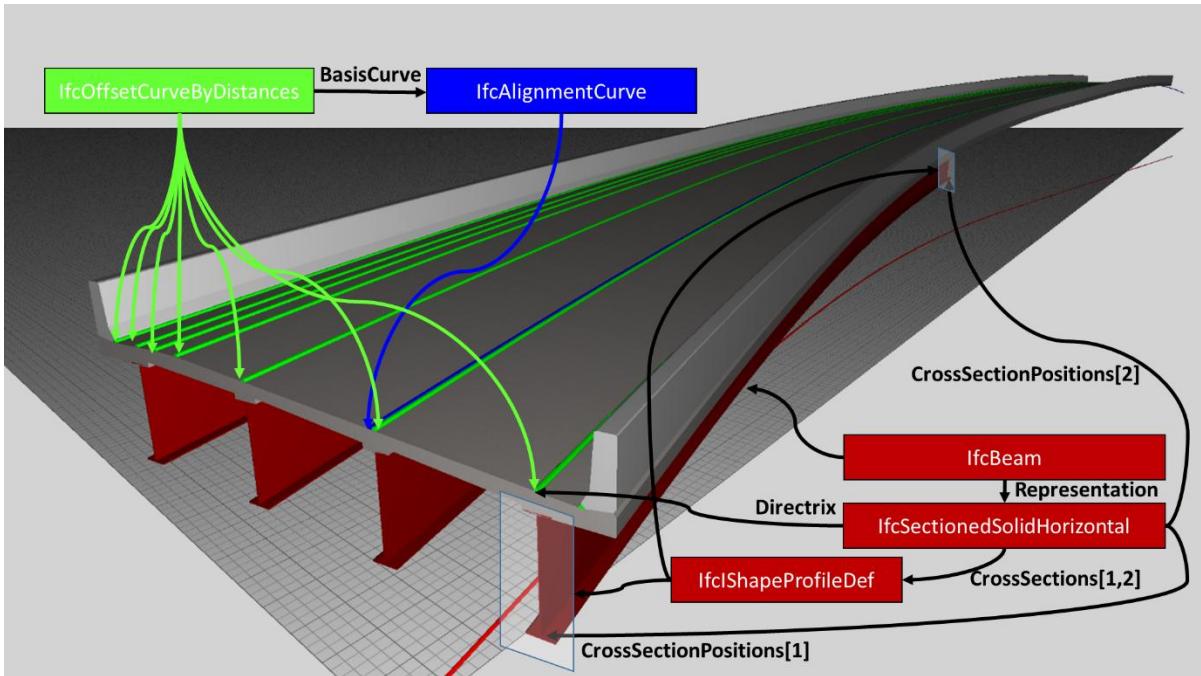


Figure 16: Sectioned solid with constant cross section

Figure 16, on the previous page, illustrates a girder segment (*IfcBeam*) having swept geometry (*IfcSectionedSolidHorizontal*) with a constant I-shape cross-section (same *IfcShapeProfileDef* at start and end) spanning between two positions (*IfcDistanceExpression*) along a string line (*IfcOffsetCurveByDistances*) defined relative to an alignment curve (*IfcAlignmentCurve*) where lateral and vertical offsets are constant.

As depicted below, Figure 17 illustrates a bridge deck (*IfcSlab*) having swept geometry (*IfcSectionedSolidHorizontal*) with a variable cross-section (varying *IfcArbitraryClosedProfileDef* instances) spanning multiple positions (*IfcDistanceExpression*) based on superelevation transitions along the main alignment curve (*IfcAlignmentCurve*) where lateral offsets are constant and vertical offsets vary along the alignment.

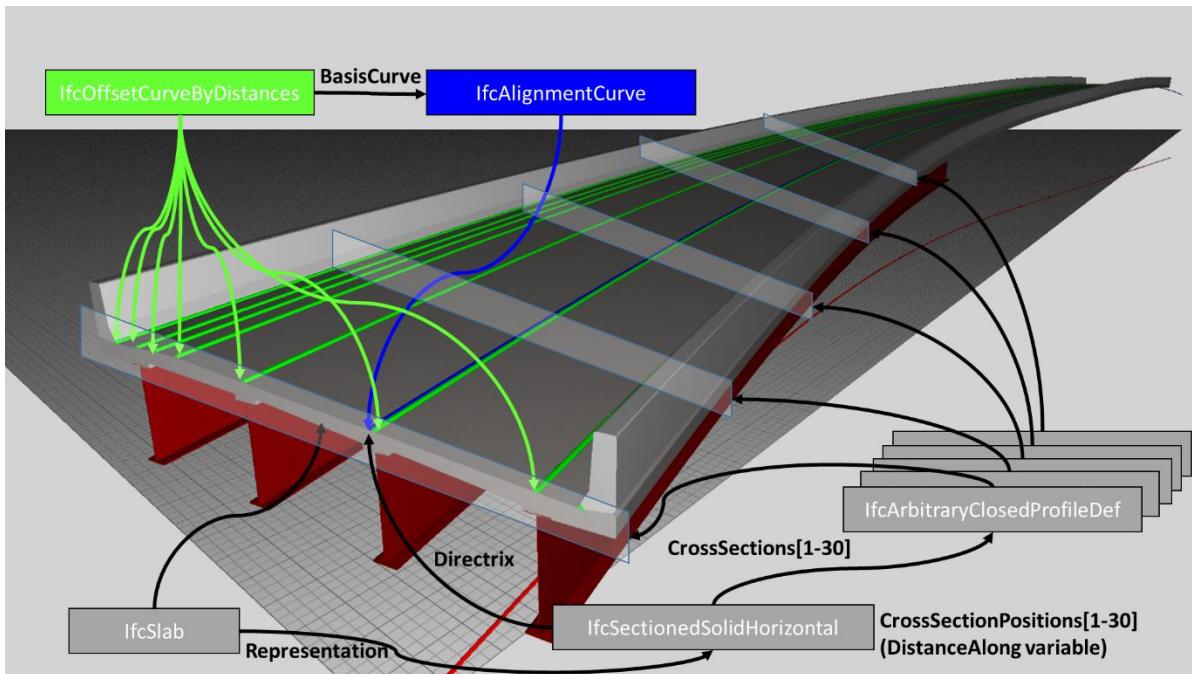


Figure 17: Sectioned solid with variable cross section

It is foreseeable that *IfcSectionedSolidHorizontal* or a derivative may be extended in the future to support skew at each end to address scenarios where a road intersects with another at an angle, or a bridge crosses another road at an angle where available clearance below requires piers or abutments to be skewed, and therefore girders and decks must follow suit at joints. It is anticipated that the IFC-Bridge follow-on project will propose such extension upon modeling applicable use cases in detail.

It is also foreseeable that a new definition may be introduced in the future to represent an extruded solid that is defined according to starting and ending positions along an alignment curve, as is typically the case for precast concrete girders. While the same geometry can be described with *IfcExtrudedAreaSolid*, a definition relating alignment positions would reflect parameters consistent with how such dimensions are commonly represented on design plans.

3.8 Railway cant

For representing railways it is important to explicitly define the cant and associate it with the alignment. In this section the Overall Architecture project provides one example of modeling railway cant using the IFC data model. This is not a proposal for an IFC cant information model, such proposal shall be developed in IFC-Railway project. Similarly, this section does not include an example of road super-elevation modeling, which would be a far more complex issue and should be addressed in a future IFC-Road project. To summarize, this section is limited to one example of railway cant for IFC.

Cant information, also known as superelevation information, associated to a railway alignment, which is typically the track center line, should be specified by applying the same basic logic as in the LandInfra Conceptual Model (Draft 2016-07-01) and in the LandXML schema (v1.2):

The difference in elevation (applied cant) of the two rails of a track is specified at the beginning and at the end of the alignment, and at every point in between where the cant development rate is changing. Specification of all cant events for the alignment are gathered in a collection that is related to (OGC LandInfra) or contained in (LandXML) the alignment instance.

The LandInfra classes for representing cant information are CantSpecification and CantEvent, whereas LandXML uses elements Cant and CantStation. A comparison analysis of these concepts in both LandInfra and LandXML has been performed and the outcome can be found in Annex 2: Comparison OGC LandInfra and LandXML.

The main difference in LandInfra and LandXML is that LandInfra always assumes linear change of applied cant from one specified value to another, as is usual when Clothoid is the type of transition curve. In contrast, LandXML allows the transition type to be specified, as it is usually matched with the rate of change in horizontal curvature. Another difference is that LandXML relies on stationing for locating cant events, while LandInfra allows distance expressions using various types of absolute or relative linear referencing methods.

One way of capturing cant information in the IFC model could be to define a property set (Pset_CantEvent) with:

- an applied cant value for both left and right rail, as in some cases both rails can be different from 0 at the same location;
- a rate of change, when the applied cant value is different from the previous;
- the rate of change could be implied according to the corresponding transition curve type, as in LandXML, unless an explicit analytical definition can be given.

The location of each cant event along an alignment curve would be given as a linear placement, defined by an instance of *IfcReferent* (where Pset_CantEvent would be assigned by *IfcRelDefinesByProperties*). All the referents representing cant events for a single track, would be nested under the main alignment of that track by one instance of *IfcRelNests*.

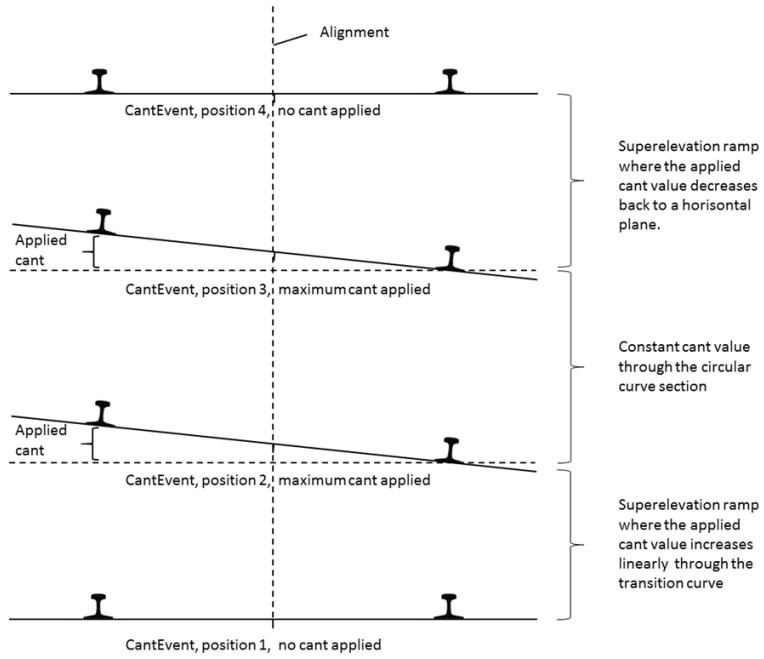


Figure 18: Example of four cant events at positions 1-4 along railway track alignment (Courtesy: OGC).

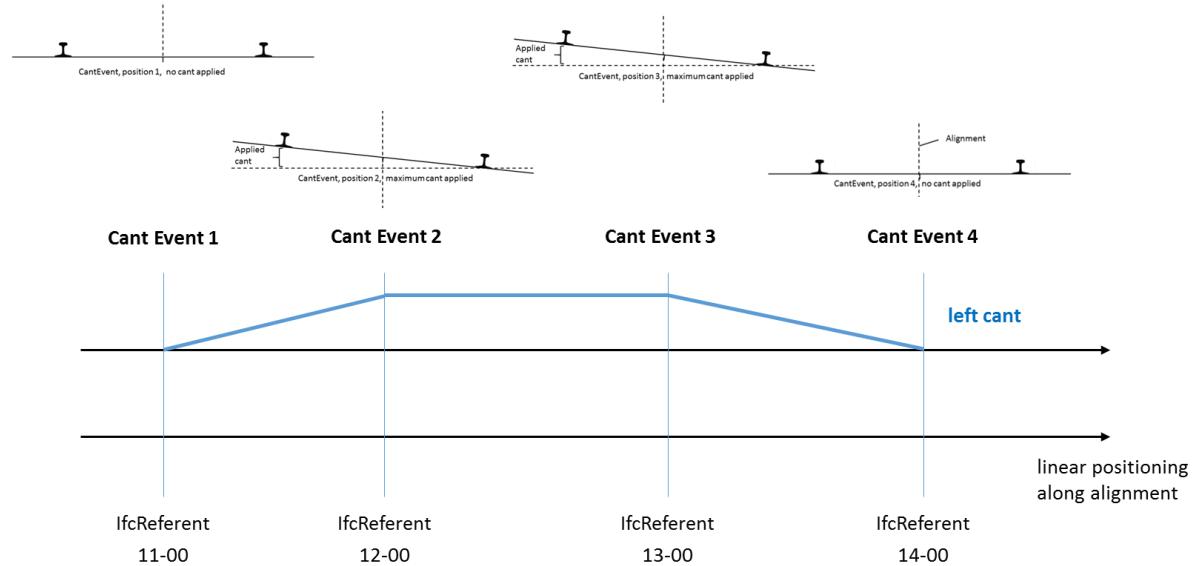


Figure 19: Illustration of Cant Events located along the alignment using IfcReferent. The cant is interpolated linearly between two subsequent Cant Events

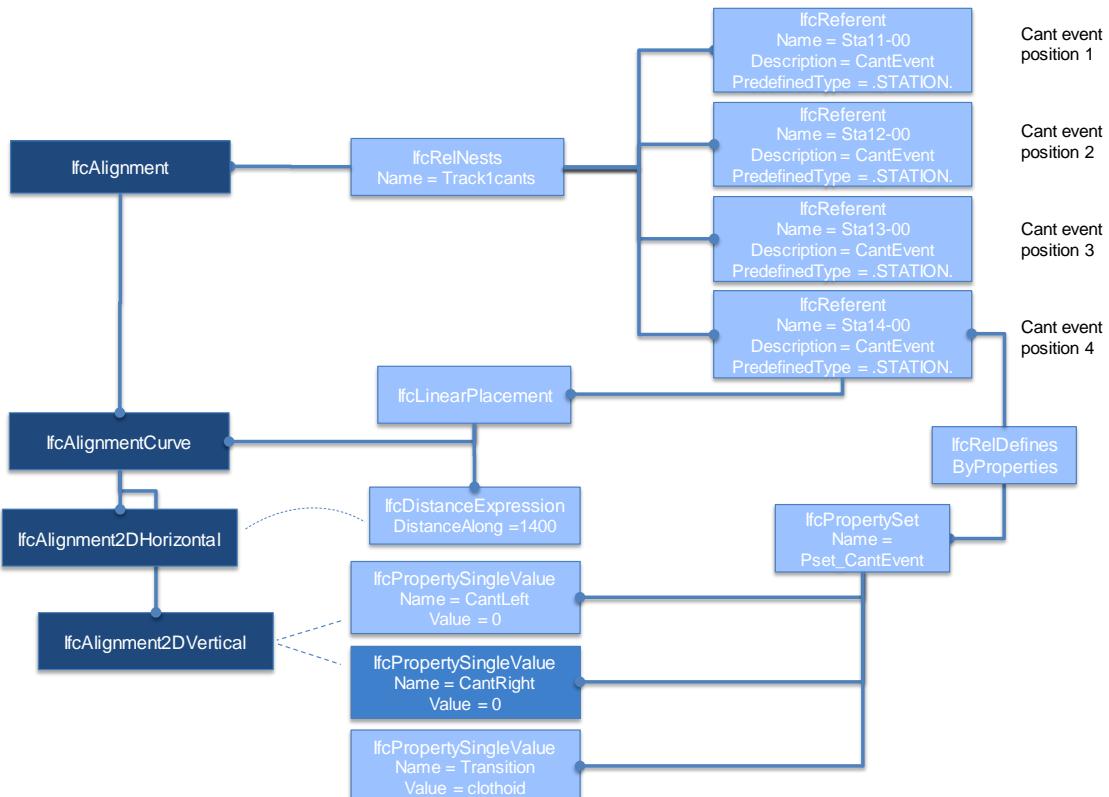


Figure 20: Railway cant: a possible solution using property sets to capture applied cants at locations defined by an **IfcReferent** instance

Figure 18, Figure 19 and Figure 20 show an example of four cant events in a situation where the track geometry is changing from a straight line (no cant before or at station 11-00) to a circular curve (between stations 12-00 and 13-00) through a clothoid (between stations 11-00 and 12-00), and again back to straight line through another clothoid (between stations 13-00 and 14-00).

Note: The authoring team has received feedback from the German Infra Expert Group formulating additional requirements for track and cant modeling, including the introduction of a dedicated *Track* entity, the decoupling of alignment elements and CantEvents as well as the use of value-pairs for coupling left and right cant. This will be taken forward as valuable input for the upcoming IFC-Rail project.

4. Physical element description

The guidelines described in this section should be followed when investigating whether to add new concepts to the IFC object model. In particular for those concepts meant to represent physical elements in civil structures.

N.B. The examples below are provided solely to explain the possibilities when introducing new concepts, they are not meant to propose a solution in any particular case.

For each new concept being considered, the closest existing concept (entity type) in the IFC class structure should be identified according to their function, not according to the domain where it may be used.

Example: Slab in bridge structure would be a type of *IfcSlab* (or may have common supertype *IfcBuildingElement* with *IfcSlab*), rather than be classified as a type of *IfcBridgeElement*, derived from *IfcCivilElement*. The name “*IfcBuildingElement*” exists for historic reasons, when IFC only covered building like structures. Elements being subtypes of *IfcBuildingElement* should not be interpreted as being only applicable to buildings, but according to its function – a vertical element to separate space or ground and to withstand loads should always be an *IfcWall* – either for walls in buildings, for sound protecting walls along roads, for wing walls in bridges, etc.

Note: in the current IFC4 Release *IfcCivilElement* “has been introduced as a stub for future extensions of this specification to include an object model for civil engineering works”. While many, if not all, of these objects can be represented as types of *IfcBuildingElement*, *IfcDistributionElement*, or *IfcGeographicElement*, it is for the future extension projects in the civil and infra domain to define proper usage of *IfcCivilElement* and propose appropriate subtypes (if any).

The IFC Taxonomy Flow Chart may be used as a decision tree in the process, as shown in Table 1:

- Starting from top left, go downwards and answer yes/no to each question: if yes, then go rightwards; if no, keep going downwards.
- Repeat the cycle (going rightwards or downwards within the next level), until at leaf node, which represents the final classification.
- While not perfect at distinguishing in every case, this illustrates a general guideline of how to distinguish object types by function based on the existing breakdown structure.
- The elements are ordered vertically by priority, such that a YES answer to a question takes priority higher in the tree than lower.

The particular questions indicated in the flow chart are not prescriptive, they are intended to be representative of how a breakdown could be distinguished, and are subject to modification and future refinement.

IfcObject	
IfcProduct	Is it defined in space?
IfcAnnotation	Is it a marking (rather than a real object)?
IfcSpatialElement	Does it describe an invisible volume of space for some functional purpose?
IfcSpatialZone	Does it describe a volume of space based on a partition within a system or network?
IfcSpatialStructureElement	Does it describe a volume of space based on how it is logically organized by people using it?
IfcSite	Does it describe an area of land?
IfcBuilding	Does it describe an enclosed structure as a whole, or a complex of multiple structures?
IfcBuildingStorey	Does it describe a volume of space that encompasses all connected spaces at the same vertical elevation?
IfcSpace	Does it describe any other spatial volume?
IfcExternalSpatialStructureElement	Does it describe a volume of space external to the logical organization of people using it?
IfcStructuralItem	Does it describe a structural (analysis) idealization of an object?
IfcFeatureElement	Does it describe a modification to the shape of an object?
IfcTransportElement	Does it transport people or freight?
IfcDistributionElement	Does any solid, liquid, gas, or electricity flow through it?
IfcDistributionControlElement	Is its sole purpose to sense or control flow conditions?
IfcUnitaryControlElement	Does it perform two or more of the functions listed below?
IfcProtectiveDeviceTrippingUnit	Does it interrupt power flow based on electrical power conditions?
IfcController	Does it perform control logic based on sensed physical conditions?
IfcFlowInstrument	Does it sense physical conditions between two points in space?
IfcSensor	Does it sense physical conditions at a single point in space?
IfcActuator	Does it control physical conditions?
IfcAlarm	Does it alert people based on abnormal conditions?
IfcDistributionFlowElement	Is it any other purpose?
IfcDistributionChamberElement	Is it a formed volume with primary purpose of insulating distribution elements?
IfcEnergyConversionDevice	Does it convert energy from one form to another?
IfcFlowStorageDevice	Does it store energy or a flow substance?
IfcFlowMovingDevice	Does it regulate the velocity of a flow substance?
IfcFlowTreatmentDevice	Does it regulate the composition of a flow substance?
IfcFlowFitting	Does it transition flow between different modes or cross sections?
IfcFlowSegment	Does it carry flow through a consistent mode or cross section?
IfcFlowTerminal	Does it terminate a network or serve any other purpose?
IfcFurnishingElement	Is it not permanently part of the structure (i.e. movable)?
IfcGeographicElement	Is it part of the natural environment (e.g. tree, water, ground cover)?
IfcElementAssembly	Is it a superstructure (not constructed within the ground) consisting of multiple assembled components?
IfcElementComponent	Is it used to fasten, reinforce, or insulate components?
IfcReinforcingElement	Is it embedded within another element with main purpose providing structural reinforcement?
IfcVibrationDulator	Is its main purpose to absorb mechanical vibration?
IfcMechanicalFastener	Is it used to connect two or more objects mechanically?
IfcFastener	Is it used to connect two or more objects chemically (i.e. via adhesive, mortar, weld)?
IfcAccessory	Does it have a fixed shape?
IfcBuildingElementPart	Does it have a variable shape intended to adapt to the location where it is placed?
IfcBuildingElement	Is it a permanent built part of the facility?
IfcCovering	Does it cover another object?
IfcShadingDevice	Is its primary purpose to block sunlight?
IfcChimney	Is its primary purpose to enclose the exhaust of burning substances?
IfcWindow	Is it placed within a planar element and intended to enclose a space while transmitting natural light?
IfcDoor	Is it placed within a planar element and intended to enclose a space while allowing entry/exit?
IfcStair	Does it bridge between vertical elevations with steps?
IfcRamp	Does it bridge between vertical elevations with an incline?
IfcRailing	Is it a space separation structure preventing injury by falling or providing physical support to occupants?
IfcPile	Is it linear, vertical, and intended to transmit axial loads into the ground?
IfcColumn	Is it linear, vertical, and intended to transmit axial loads onto another object?
IfcFooting	Is it linear, horizontal, and intended to transmit shear loads into the ground?
IfcBeam	Is it linear, horizontal, and intended to transmit shear loads onto other objects?
IfcMember	Is it linear with any other orientation or load configuration?
IfcCurtainWall	Is it planar, vertical, intended to separate space horizontally, and hung from a slab above (not carrying its own load)?
IfcWall	Is it planar, vertical, and intended to separate space horizontally?
IfcRoof	Is it intended to cover the top of a structure?
IfcSlab	Is it planar, horizontal, and intended to separate space vertically?
IfcPlate	Is it planar with any other purpose?
IfcCivilElement	Is it a substructure within the ground (of loose soil or rock)?
IfcProcess	Is it defined in time?

Table 1 IFC Taxonomy Flow Chart

When going through the identification process, note that some existing entity types may be renamed (if abstract) or their semantic definition re-written to be more generic or more appropriate for civil structures.

Example: Entity type *IfcBuildingElement* could be renamed *IfcBuiltElement*.

Example: Definition "*building element comprises all elements that are primarily part of the construction of a building, i.e., its structural and space separating system*" could be re-worded "*built element comprises all elements that are primarily part of the construction of a built facility, i.e., its structural and space separating system*".

A draft proposal for organizing the new concepts proposed by IFC-Bridge, IFC-Road and IFC-Rail projects (as of early 2016), according to the principles above, can be found in document Element_breakdown_20161123.pdf (see Annex 3: Element breakdown). This document is also available on Google Drive: <https://goo.gl/7eqtFQ>. In this document the new entity types have been detached from their proposed place IFC class hierarchy, as they might:

- not all necessarily be added as new entity types, if existing ones can be used or extended in other ways (property definitions), or
- relate more closely to some other existing entity type, when classified primarily by function.

This first attempt to group similar concepts and link the groups to potential existing entity types is proposed as the baseline for the Common schema project (also known as Common Definitions), and subsequently for the IFC-Bridge, IFC-Road and IFC-Rail projects to resolve the most appropriate IFC mode extension solution for each new concept.

Once the best match in existing IFC class hierarchy has been found for a new concept, it should be considered if the existing entity type can be used as such, either 1) with associated external classification (and property definitions) for further specialization, or 2) by extending its definition and adding to its available predefined types and corresponding property sets. Only when neither of these approaches are satisfactory, should a new subtype of an existing entity type be considered (or adding attributes to a leaf node type).

The IFC object model provides the possibility for extended definitions by project, company, national or regional agreements. This capability for extension is provided by Property Set and Object Typing concepts. Property Set and Object Type definitions may be part of IFC model specification or come from an external library (discussed in Section 5.3).

Many entities have an attribute PredefinedType with a specific enumeration (with an option for USERDEFINED to allow the use of alternatives/extended code lists). This provides an alternative to adding another level of inheritance. Predefined types can also be used to drive which property set templates or type objects may be applicable to instances at that entity type.

As a general rule, subtypes should not be added if predefined types (code lists) and property set or object type definitions can be used.

When adding another level of inheritance is considered, there should be a clear justification for doing so:

- are there any specific relationships that need to be supported in subtype level?
- are there any specific rules applicable to subtype?
- is software functionality typically different depending on subtype?
- is the subtype and are all its attributes globally applicable (not just locally, or in specific life cycle stage)?

The following circumstances are examples of when it may be particularly appropriate to consider using a data-driven property set (see Section 5.2), rather than defining a class with attributes within the IFC model:

- where a class has a lot of properties that resolve to simple data types,
- where attributes can be grouped in particular ways that express functional data requirements (for instance, to express different data requirements at various stages of the project lifecycle) and where the same property may need to be repeated in several property sets,
- where attributes or properties may be used in a particular view, discipline, lifecycle phase or other subset purpose of the object descriptions,
- where attributes or properties can be used in regional extensions in addition to the international IFC specifications.

5. Classification, Property Sets and linkage with object type libraries

5.1 Classification

Classification information can be associated to any subtype of *IfcObjectDefinition* (or *IfcPropertyDefiniton*) using the *IfcRelAssociatesClassification* relationship, referring to *IfcClassificationReference*. This existing capability in IFC4 can be used when infrastructure entities are classified, whether they are:

- spatial structure elements, such as *IfcRoad*, *IfcRailroad*, *IfcBridge*, or any of their parts;
- positioning elements, such as *IfcAlignment*;
- or any physical element that might be introduced as a subtype of *IfcElement* (or its subtypes).

When modeling infrastructure there often is a need to not only classify objects, but to also classify certain identifiable parts of their geometry. In IFC4 classification information can also be associated with a select list of resource level objects. These resource level objects should not be subtypes of *IfcObjectDefinition* or *IfcPropertyDefiniton*. The association is then created using *IfcExternalReferenceRelationship*, referring to *IfcClassificationReference*. One of the items in the *IfcResourceObjectSelect* list is *IfcProfileDef*, allowing cross section profiles to be classified; for other types of object geometry parts (such as surfaces or breaklines) associating classification information has not been possible. As a generic mechanism to identify any part of object geometry for naming and coding (or classification purposes), use of existing *IfcShapeAspect* is proposed, and it has been added to the select list.

The example in Figure 21 below illustrates a situation, where part of a road structure (all structural layers) is represented by an instance of *IfcBuildingElement*³, classified according to an external classification system named “Infra classification” as an item with code “2130” and title “Base Structure” (some data omitted for brevity, e.g. publisher and location/URI of the classification system). The shape representation of the object consists of two TIN surfaces (top and bottom surface of the object). The both TIN surfaces are also individually assigned to additional shape representation instances, identified as shape aspect and named as “Top” and “Bottom”, respectively. Both shape aspects are also assigned a classification from a system named “InfraBIM”, code “201000” and title “Top Surface of Structure”, and code “201100” and title “Bottom Surface of Structure”, respectively.

³ *IfcBuildingElement* is an abstract entity type and cannot be instantiated as such, but it is used in this example since the correct subtype is not yet known (pending IFC Road project development).

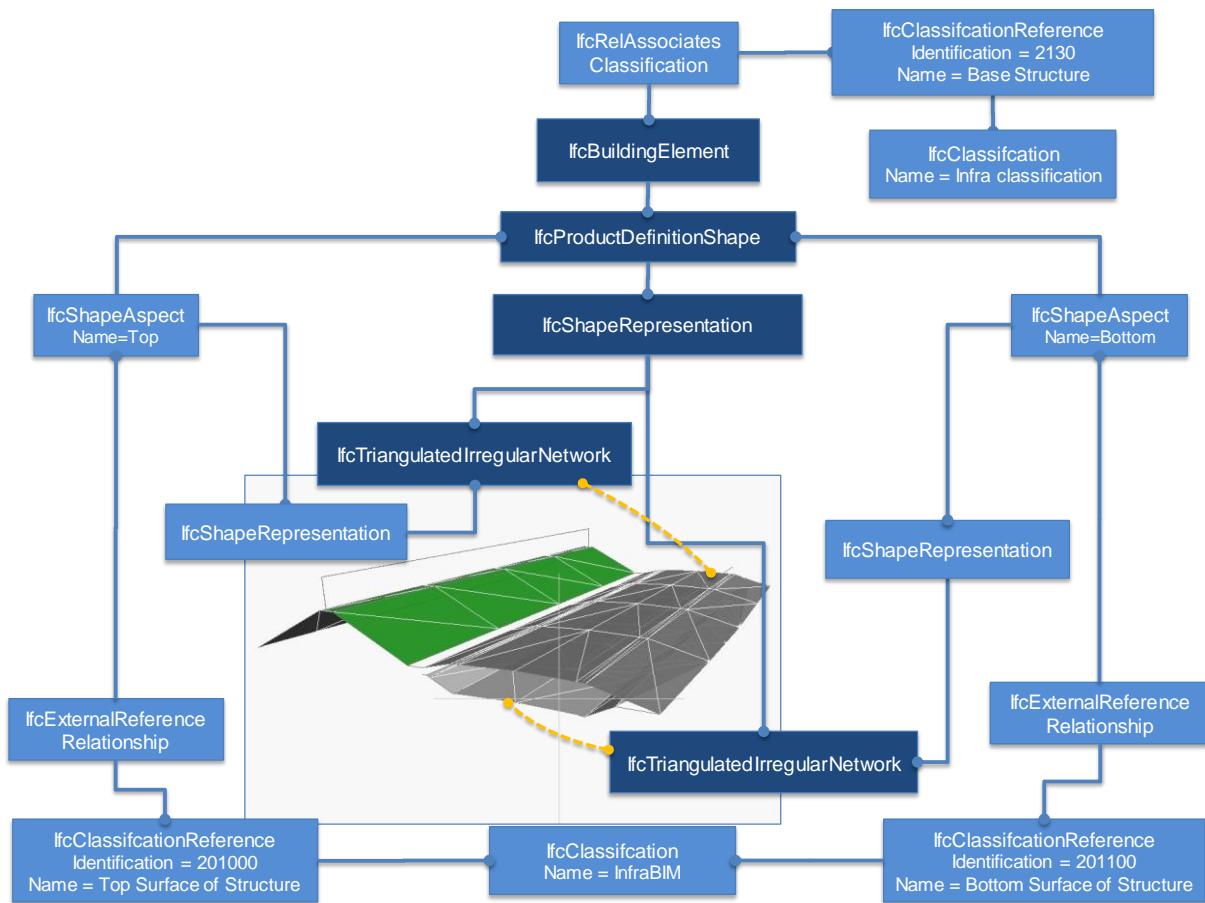


Figure 21. Classification example: road structure segment with its shape represented by top and bottom surface.

Another example in Figure 22 below illustrates a situation where the pavement of a road segment is represented by three String Lines (as shown in Figure 8). Here the pavement is modelled as **IfcElement**⁴ and classified as an item from “Infra classification” system with code “214110” and title “AB”. Its shape representation has three polylines: left edge, center and right edge of the top surface, respectively. This entire shape representation is identified also as a shape aspect (named “TopSP”) and classified with code “214111” and title “Top surface of pavement” from “InfraBIM” system. Individually, the polylines are also assigned to shape representation instances belonging to shape aspects, names from left to right as “LEP”, “CL” and “REP”, and classified according to “InfraBIM”: code “122” and title “Edge of pavement” for left and right edge, and code “127” and title “Breakline” for center.

⁴ **IfcElement** is an abstract entity type and cannot be instantiated as such, but it is used in this example since the correct subtype is not yet known (pending IFC Road project development).

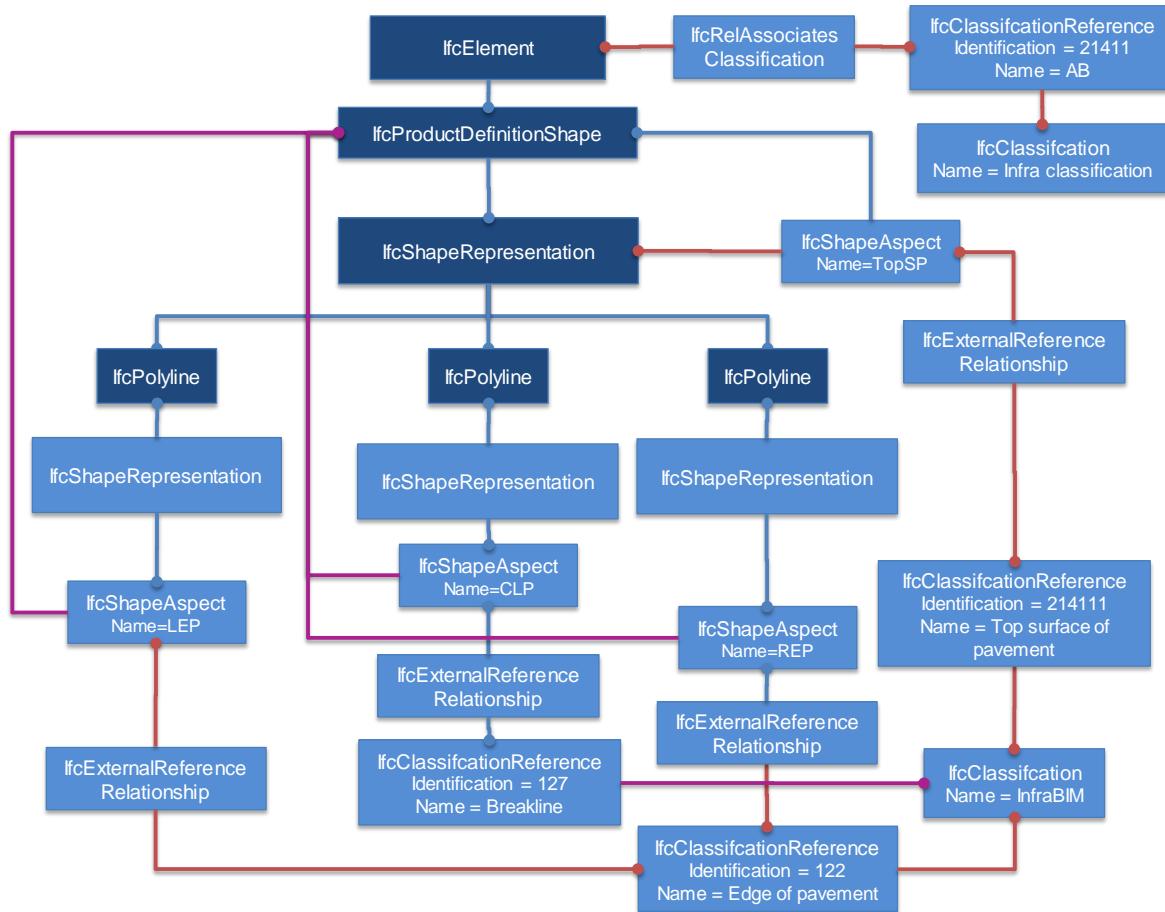


Figure 22. Classification example: road pavement segment with its shape represented by three polylines (top surface edges and center).

For classification purposes, it is also possible to relate individual items to concepts of the buildingSMART data dictionary (Section 5.3).

5.2 Property Sets

An internationally standardized data model such as the Industry Foundation Classes is supposed to provide common data structures which can be used in an identical manner in most parts of the world. In order to nevertheless cover also information exchange requirements of particular regions, countries or projects, the IFC provide a flexible extension mechanism called “property sets”.

Property sets are basically (typed) name-value pairs which are defined outside of the EXPRESS schema using external XML files which must conform to the “Property Set Definition” schema defined by bSI. The list of properties (the “names”) are usually managed by making use of the buildingSMART Data Dictionary (bsDD) for managing terminology and properties in an integrated manner. Of particular interest is the combination of national classification systems (see Section 5.1) with the properties required by national use cases.

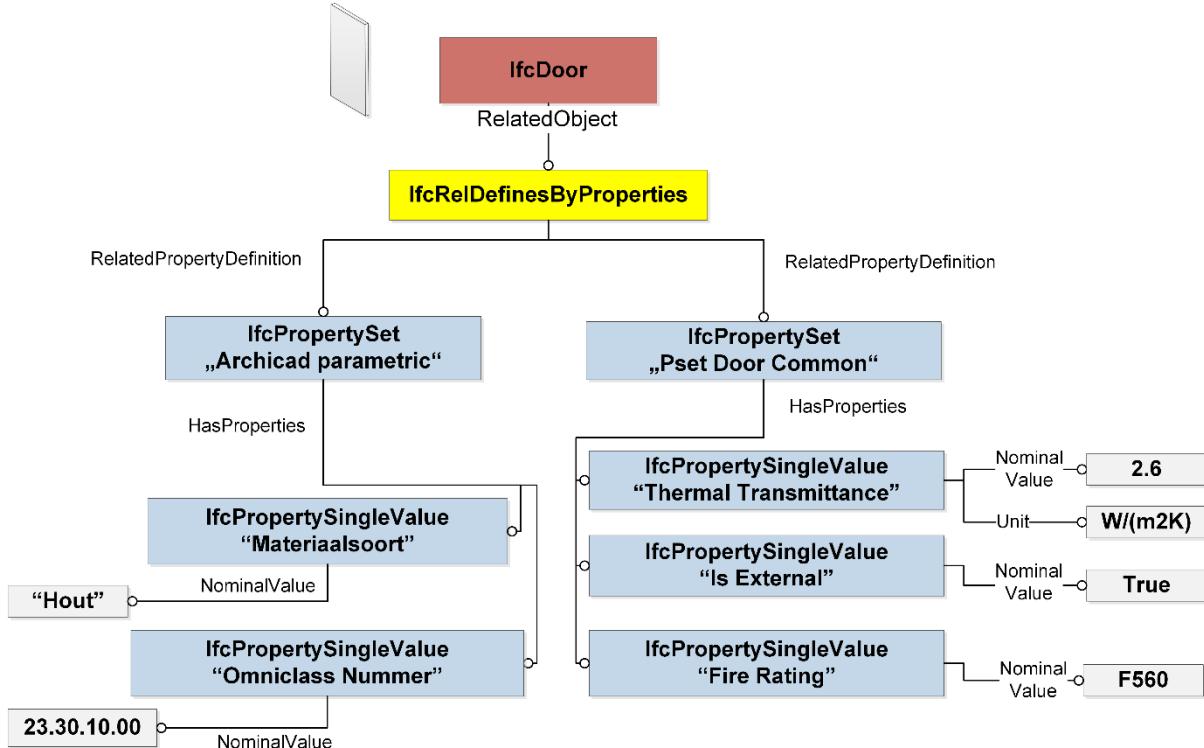


Figure 23: Using the Property Set mechanism for specifying, regional, national or local information requirements (Courtesy Jakob Beetz, TU Eindhoven)

Property sets may form part of a Model View Definition (MVD). This allows to provide the required flexibility and at the same time supports to check the model content of an IFC instance file for conformance with the exchange requirements defined for a particular exchange scenario.

For the future extension of IFC, a careful separation between the functionalities of the international data model and its local extensions is recommended. The applicability of property sets should also be taken into account when deciding upon the extension of the type system. In many cases new subtypes can be avoided by applying the property set mechanism.

For some use cases, the usage of **IfcProxy** objects for geometry description and the association of property sets for all classification and attribute data may be sufficient.

5.3 Linkage with the buildingSMART Data Dictionary and Object Type Libraries

The buildingSMART Data Dictionary (bSDD) and other Object Type Libraries (OTLs) such as the Dutch CB-NL provide construction-specific terminology in a structured computer-readable manner. The ontologies define concepts, their properties and relationships (Figure 24, Figure 25). They are typically modeled according to ISO 12006.

While bSDD has its focus on providing a multi-language vocabulary and managing (widely agreed) property sets, other OTLs such as CB-NL focus on national particularities.

For linking IFC entities with bsDD or other OTLs, there are two major approaches:

- Usage of the classification approach. The identification attribute can refer to the URL of the online resources of the corresponding OTL concept, e.g. ont.cbnl.org/cb/def/Brug
- Application of the Linked Data approach (see Section 5.4), where a linkage scheme is applied. This requires both the ontology and the IFC schema to be modelled using Semantic Web Technology such as OWL. For the latter, the bSI published standard ifcOWL exists (see bSI Linked Data Working Group).

While the former approach allows for a direct integration in IFC instance files, the latter allows for more flexibility. A detailed explanation of the pros and cons of both approaches was, however, out of scope of this project.

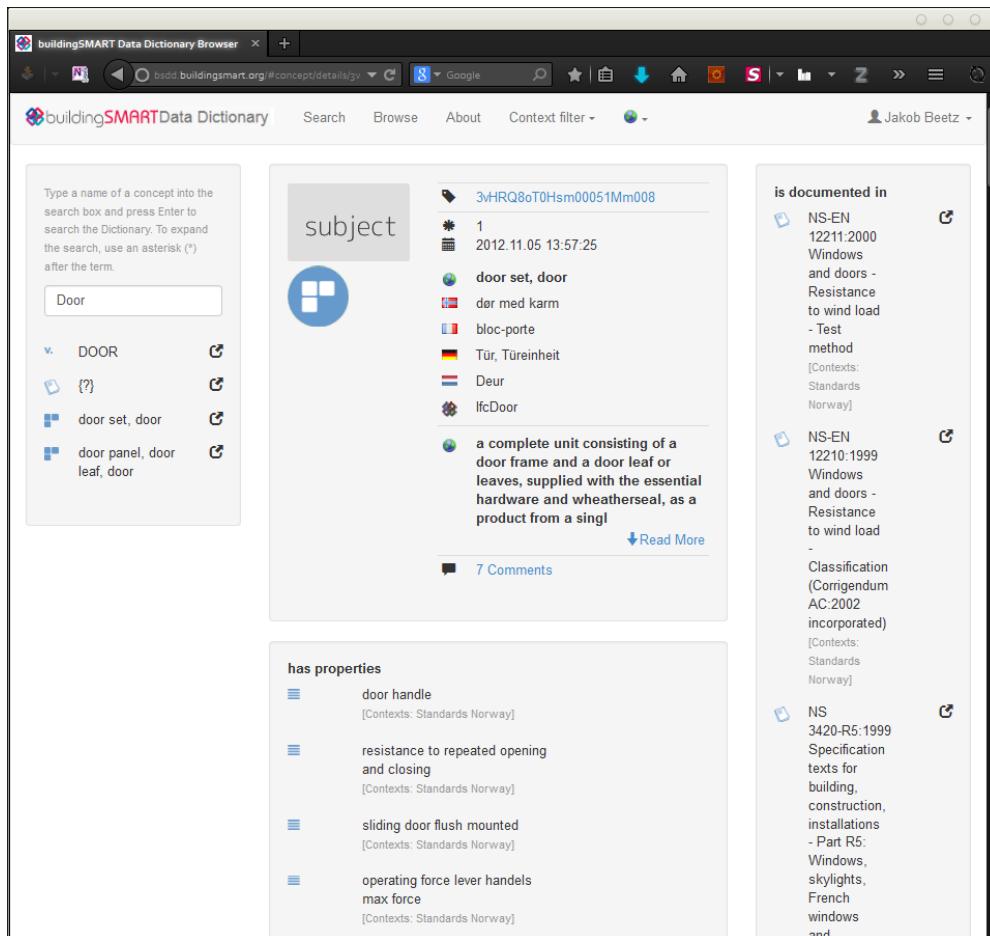
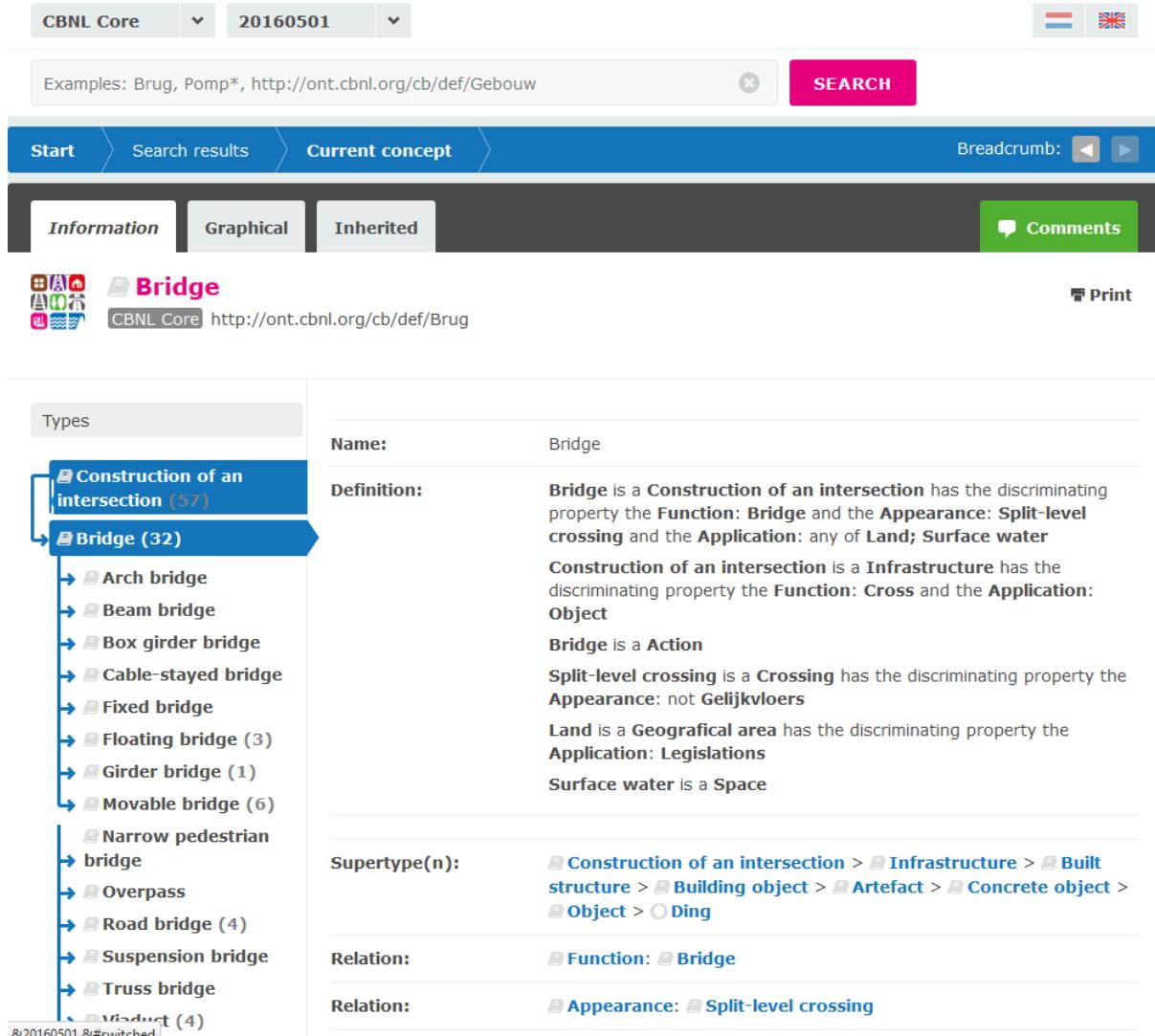


Figure 24: The BuildingSMART Data Dictionary, here showing the concept "Door"



The screenshot shows the buildingSMART CB-NL interface. At the top, there are dropdown menus for 'CBNL Core' and '20160501'. On the right, there are language selection buttons for English and Dutch. Below the header is a search bar with placeholder text 'Examples: Brug, Pomp*', a search button, and a close button. The breadcrumb navigation shows 'Start > Search results > Current concept'. The main content area has tabs for 'Information', 'Graphical', and 'Inherited', with 'Comments' on the right. The 'Information' tab is selected. The page title is 'Bridge' with a small icon. Below the title, it says 'CBNL Core http://ont.cbnl.org/cb/def/Brug'. On the right, there is a 'Print' button. The left sidebar is titled 'Types' and lists categories: 'Construction of an intersection (57)' (selected), 'Bridge (32)' (highlighted in blue), 'Arch bridge', 'Beam bridge', 'Box girder bridge', 'Cable-stayed bridge', 'Fixed bridge', 'Floating bridge (3)', 'Girder bridge (1)', 'Movable bridge (6)', 'Narrow pedestrian bridge', 'Overpass', 'Road bridge (4)', 'Suspension bridge', 'Truss bridge', and 'Windt (4)'. The main content area contains the following details for the 'Bridge' concept:

- Name:** Bridge
- Definition:** Bridge is a Construction of an intersection has the discriminating property the Function: Bridge and the Appearance: Split-level crossing and the Application: any of Land; Surface water
- Construction of an intersection** is a Infrastructure has the discriminating property the Function: Cross and the Application: Object
- Bridge** is a Action
- Split-level crossing** is a Crossing has the discriminating property the Appearance: not Gelijkvloers
- Land** is a Geographical area has the discriminating property the Application: Legislations
- Surface water** is a Space
- Supertype(n):** Construction of an intersection > Infrastructure > Built structure > Building object > Artefact > Concrete object > Object > Ding
- Relation:** Function: Bridge
- Relation:** Appearance: Split-level crossing

Figure 25: The Dutch Object Type Library CB-NL, here showing the concept “Bridge” its definition and relations

5.4 Linked Data approaches

As an alternative to integrating classification by using IFC entities as described in Section 5.1 above, it is possible to apply Linked Data approaches where two or more data sets modeled in different schemas are combined by associating the IDs of the corresponding data items (Figure 26).

For implementing Linked Data approaches, typically technology of the Semantic Web is applied, most importantly the web ontology language (OWL). For linking two data models, both need to be to be modelled using OWL. For representing the IFC data model in OWL, the bSI published standard ifcOWL exists.

Currently it is investigated how the German national standard OKSTRA can be linked to the IFC model using Linked Data Approaches. To this end, an okstraOWL representation is being developed.

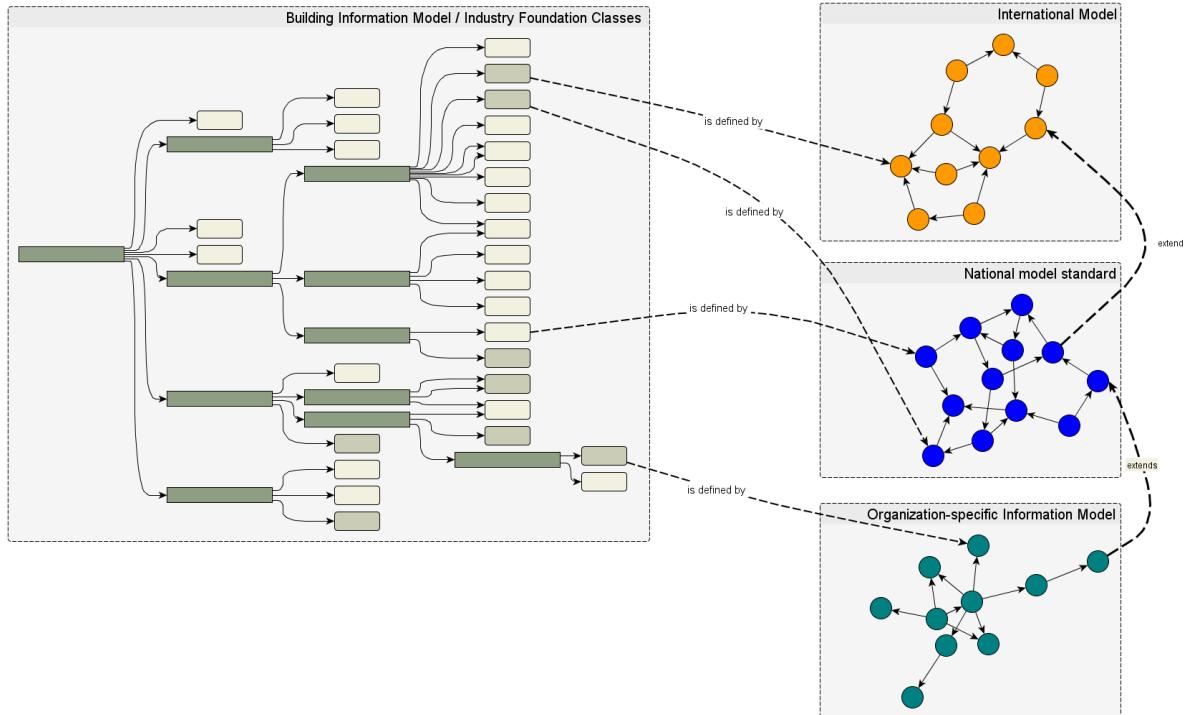


Figure 26: Using linked data approaches for combining IFC with regional, national and project-specific data models
(Courtesy: Jakob Beetz, TU Eindhoven)

6. Summary and Outlook

This document provides an overview of the existing and new data IFC structures for modeling infrastructure facilities as well as guidelines for the use of these data structures and for developing future extensions.

Among the main new features the IFC data model now offers for infrastructure modeling are the alignment, the linear referencing, and the horizontal sweeping. Together they provide mechanisms for representing and placing geometry as required by many use cases in infrastructure modeling and data exchange. At the same time existing features such as the possibility to define a global reference system and the application of classification are very much required for IFC-based infrastructure modeling.

It is of major importance for any IFC extension project to limit the introduction of new IFC types (entities) to a reasonable extent as handling new entities puts an extra burden to developers and may hinder the fast adoption of the developed IFC extensions by software vendors. Accordingly, this document has provided guidelines for the decision whether to create a new (sub-)type or not. It also explained how available mechanisms such as classification, property sets and linked data can be used to combine local, regional or project-specific information exchange requirement with the core IFC data model.

The authors believe that the now available data structures form a sound foundation for the work in the upcoming IFC-Bridge, IFC-Rail, and IFC-Road projects. These projects will define in more detail how the data structures are applied to model particular infrastructure objects and provide concrete examples. Also, the spatial data structure has to be discussed further and the final decision regarding its setup have to be taken. It is strongly recommended that all IFC-Infra projects implement a strong exchange of information, e.g. by the creation of joint “common definitions” work packages. This is of particular importance as some joint data structures such as subsoil and earthworks are still missing.

It is also strongly recommended that use cases to be implemented by the future IFC extensions are formally defined and prioritized in order to focus the development effort of both the bSI Infra Room members and the software vendors and harvest the “low hanging fruits” first.

Annex 1: Spatial Structure – Explanatory Text

The spatial hierarchy in IFC creates a framework for the spatial positioning of physical entities within a piece of infrastructure. For example, it allows a section of road or railway (or any physical feature, such as a drain or light pole) to be associated with a named section of that piece of infrastructure. This facilitates asset management and permits a meaningful spatial organisation of an IFC model.

Proposed Spatial Structure

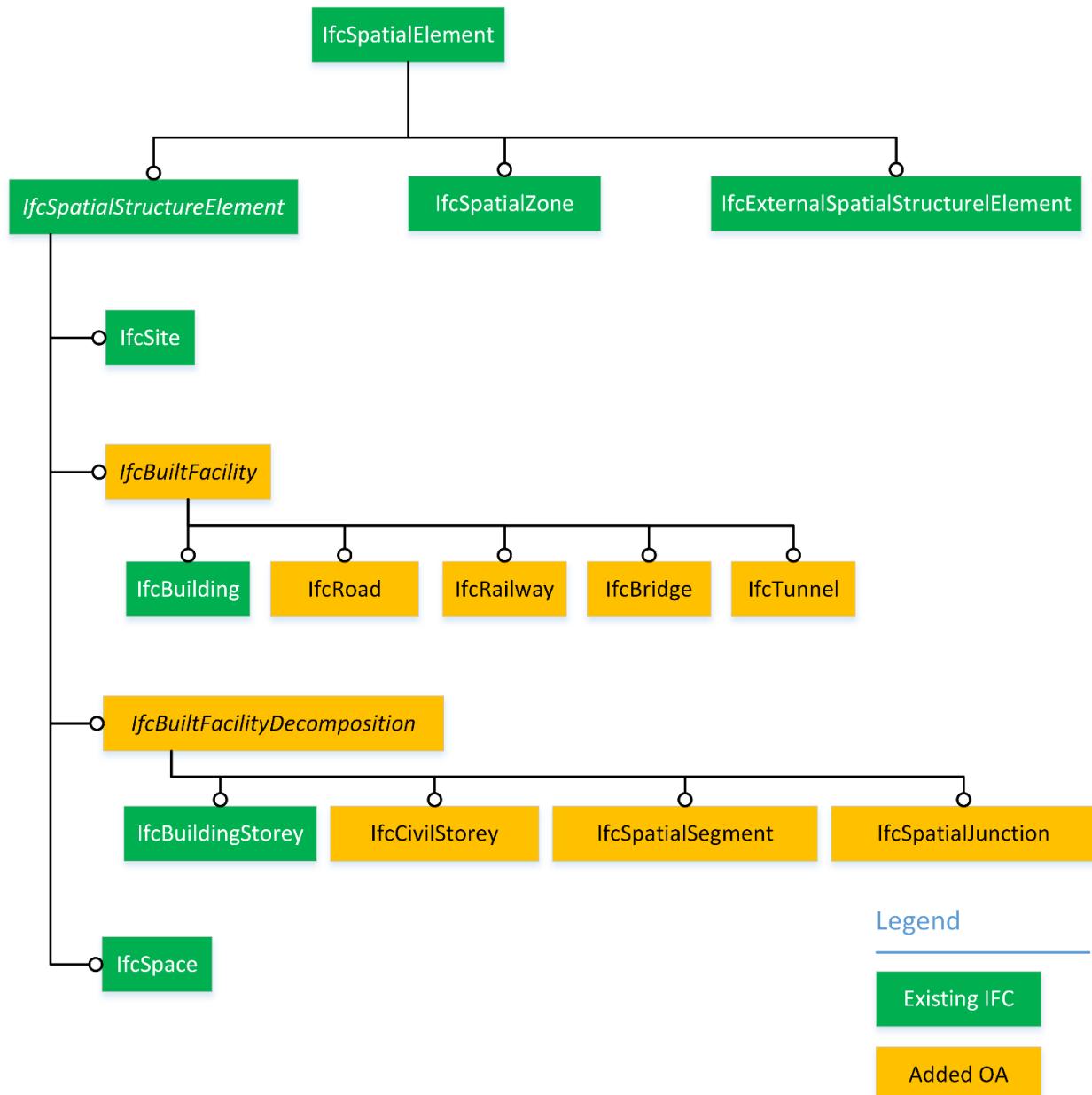


Figure 27 - Proposed Spatial Structure Concepts to support Infrastructure

The spatial structure hierarchy defined here to support infrastructure follows the same principles as applied in previous releases of IFC, ensuring that a project defined using these concepts is fully backwards compatible. The introduction of new abstract superclasses (*ifcBuiltFacility* and *ifcBuiltFacilityDecomposition*) maintains the four levels of decomposition, while permitting the introduction of new classes of decomposition at those two levels to accommodate infrastructure entities.

Consistent with current IFC specification, the top of a spatial hierarchy is always *ifcProject*, and the only “required” spatial entity is at least one instance of any sub-class of the abstract superclass, *ifcBuiltFacility*. This conforms with current practice where a minimal spatial hierarchy would consist of an *ifcProject*, consisting at least one instance of *ifcBuilding*, with no *ifcSite* entity and no further decomposition.

It is critical in this discussion to note the definition of a project in IFC, and the purpose of creating an IFC model. IFC4 defines a project in these terms:

“*ifcProject* indicates the undertaking of some design, engineering, construction, or maintenance activities leading towards a product. The project establishes the context for information to be exchanged or shared, and it may represent a construction project but does not have to. The IfcProject's main purpose in an exchange structure is to provide the root instance and the context for all other information items included.”

The spatial hierarchy for any project is defined using the objectified relationship, *ifcRelAggregates*. This relationship has two mandatory attributes, the first specifying the *RelatingObject* (the parent being decomposed) and the second being an unordered list of *RelatedObjects* (being the set of children that decompose the aggregated parent). For example, Figure 2 (from the IFC4 documentation) shows an *ifcProject* consisting an *ifcSite* and an *ifcBuilding*. Notice that the spatial elements can have a local placement and provide containment for physical entities.

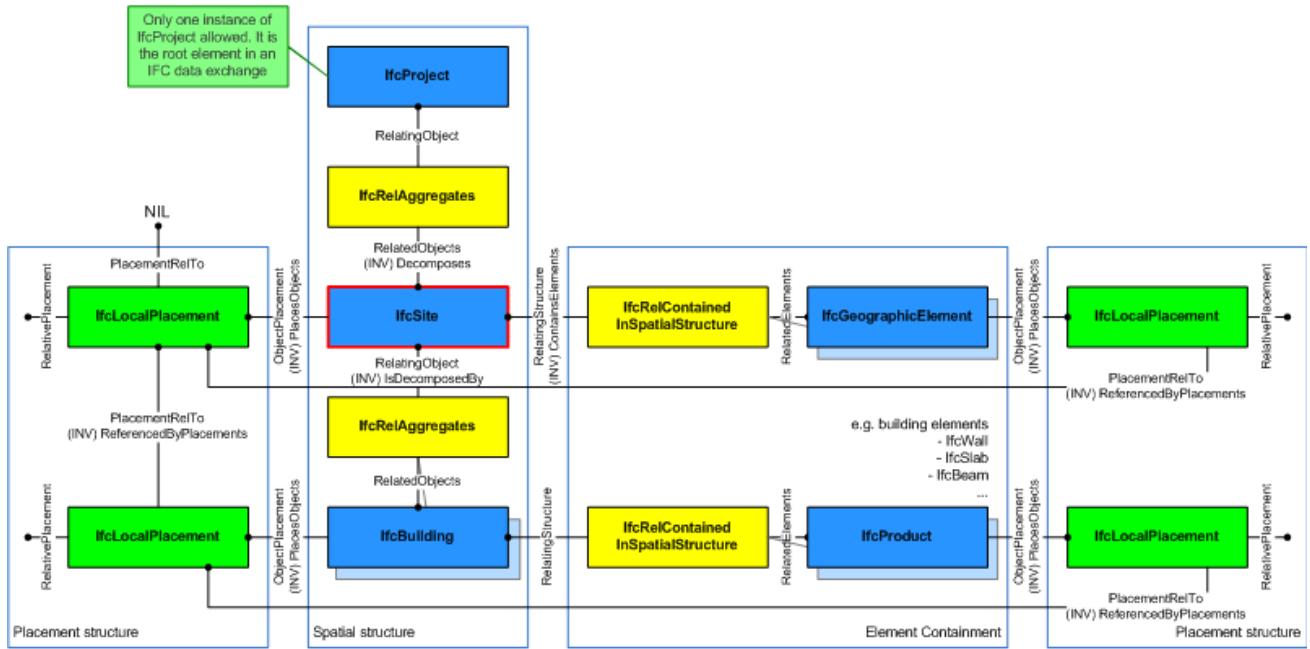


Figure 28 - Simple Spatial Hierarchy in IFC4 (source IFC4 On-Line Documentation)

Due to their scale and complexity, infrastructure projects generally consist of more than one category of *ifcBuiltFacility*, which in turn may be composed of more than one *ifcBuiltFacilityDecomposition* element. For example, a major railway project will consist of railway lines as well as bridges, tunnels and railway stations (*ifcBuilding* entities), while any one of those bridges (for example) may be composed of two *ifcCivilStoreys*, each carrying one or more road or railway segments (*ifcSpatialSegments*).

For that reason, we need to define a few principles to be followed (again, conforming to current practice to ensure backwards compatibility) that address these complexities. These are discussed in the following sub-sections.

Strict Acyclic Decomposition Hierarchy

The current spatial structure in IFC4 forms a strict acyclic decomposition hierarchy, always commencing with *ifcProject* and ranging in order from high to low level (site, building, building storey and space). We propose maintaining that strict hierarchy for infrastructure: *ifcSite*, *ifcBuiltFacility* (any combination of its sub-classes), *ifcBuiltFacilityDecomposition* (again, any combination of its sub-classes) and *ifcSpace*, where any sub-class of *ifcBuiltFacility* is the only “required” level (i.e. an *ifcProject* must consist of at least one *ifcBuiltFacility*).

At any level in that hierarchy, a spatial entity may be decomposed into partial entities of the same type. This is managed using the *CompositionType* attribute of the abstract superclass *ifcSpatialStructureElement*, taking on one of three enumerated values: COMPLEX, ELEMENT or PARTIAL. Unless specified otherwise, a spatial entity is assumed to be a simple (undivided) ELEMENT

that can only be decomposed by entities below it in the spatial hierarchy. A COMPLEX spatial entity can be decomposed (using *ifcRelAggregates*) into a set of spatial entities of type PARTIAL.⁵ It is proposed that the *CompositionType* attribute only be used to decompose entities of the same specific type. For example, roads can only be decomposed into other roads, etc.

In general, to enforce the strict spatial hierarchy, a spatial structure entity may only be part of one (and only one) other entity, either at the same level or next level up in the hierarchy. However, the following two exceptions are specifically allowed in IFC4:

- If there is no *ifcSite* defined, then an *ifcBuilding* can be associated directly with *ifcProject*;
- An *ifcSpace* can be associated directly with an *ifcSite* (to accommodate external spaces).

For backwards compatibility, those exceptions are maintained in this proposal and extended as follows:

- If no *ifcSite* is defined, then any sub-class of *ifcBuiltFacility* can be associated directly with *ifcProject*, but there must be at least one such sub-class of *ifcBuiltFacility* in any IFC model;
- An *ifcSpace* entity can directly form part of the decomposition of another *ifcSpace*, an *ifcSite* entity, any sub-class of *ifcBuiltFacilityDecomposition*.

Shared Spatial Entities

There are several cases in infrastructure modelling where spatial entities may form part of two separate branches in the spatial hierarchy. This occurs because at two levels in the defined hierarchy, there are several sub-classes. For example, a road segment crossing a bridge (or passing through a tunnel) would be part of the spatial breakdown of both the bridge and the road. A railway station would be modelled as a building, but would include spatial segments associated with the railway. More simply, a road intersection (*ifcSpatialJunction*) would be associated with two roads, or a pedestrian crossing is shared space associated with a traffic lane. All these cases are reasonable representations of the real world, but require those shared spatial entities to have two parents, thus breaking the strict hierarchy.

There are several solutions to this dilemma, ranging from simply allowing the spatial hierarchy to be broken (and effectively become a network, creating a potentially confusing structure) through to keeping all the “pure” spatial hierarchies intact and duplicating the shared entities with a corresponding equivalence relationship to assert their singular identity (this could be done in a single project database, or be managed in separate federated databases with external links to assert identity).

⁵ Source: documentation on *IfcElementCompositionEnum*, but this seems to be contradicted by the documentation for *ifcSite* (as well as other IFC4 spatial entities) where examples suggest a 3-level hierarchy, where a COMPLEX spatial entity can be decomposed into a set of spatial entities of type ELEMENT, which in turn can be decomposed into PARTIAL spatial entities. Proposed approach: adopt the simpler one-level decomposition, where the attribute value has a less ambiguous meaning.

The proposed approach lies between those extremes. In reality, it is useful to treat any shared spatial entity as having a primary association (perhaps related to maintenance responsibility for the asset) as well as optional secondary associations (or parents). This is analogous to a curtain wall (that spans several storeys) in a building model having defined containment in one storey, but able to be referenced to several other stories. For shared spatial entities, this is achieved by introducing a new spatial relationship, *ifcRelReferencesSpatially*, with attributes: one *RelatingSpatialElement* and one (one more) *RelatedSpatialElements*, both of type *ifcSpatialStructureElement*.

Physical Infrastructure Elements

Physical entities in an IFC model or database are associated with a spatial element using the relationship *ifcRelContainedInSpatialStructure*, with the constraint that an entity can only be contained within a single spatial structure element. However, where an element (such as a guard rail adjacent to a road or railway) is naturally associated with more than one spatial segment, the reference relationship *ifcRelReferencedInSpatialStructure* may be used to reference several spatial structure elements, though there must always be one containment relationship to locate the element within the spatial hierarchy.

Application of the Proposed Spatial Structure for Linear Infrastructure

This section describes the intended application of this spatial structure to each of the major linear infrastructure domains.

Roads

In general, the concept of a road encompasses any constructed entity that supports the free movement of people, traffic or animals. Within that definition, a forest track for bushwalking may be considered as a road, or a shared pedestrian/cycleway within a parkland setting. More commonly, any formed road to support vehicular traffic comes under this definition, as does a road designed to carry multiple forms of traffic in different lanes. The scope of the term road is intended to be very broad, and may even be used to represent a waterway. All roads are typically identified by a common name or a highway code and may have a description. These are recorded as attributes of the *ifcRoad* entity.

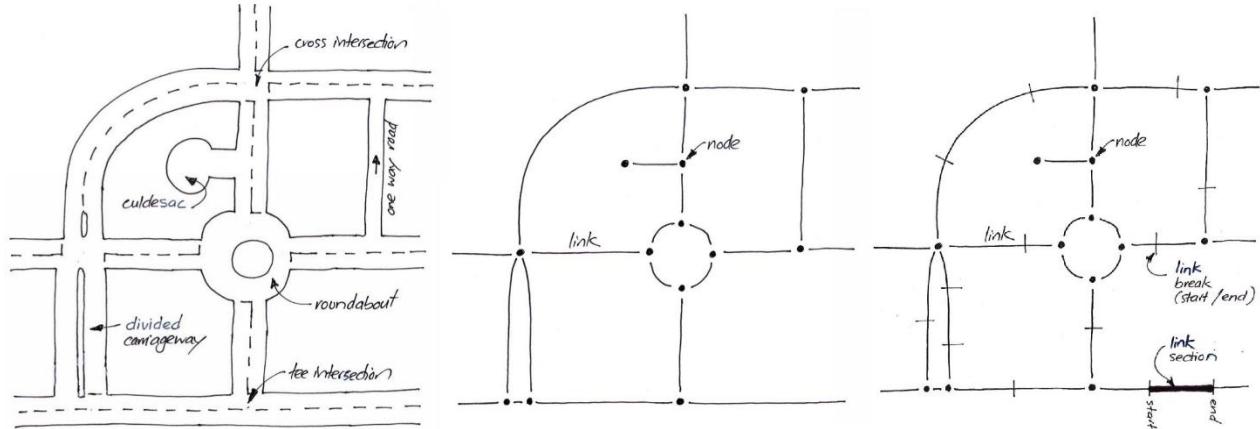


Figure 29 - Typical local road configuration, with corresponding linked node representation and extended with link breaks to depict link sections, such a change in road surface (Source: Austroads 2016)

As illustrated in Figure 3Fout! **Verwijzingsbron niet gevonden.**⁶, roads (*ifcRoad* entities) are commonly decomposed into sequences of linear segments (*ifcSpatialSegment*), typically defined between intersections or road junctions (*ifcSpatialJunction*) that themselves are part of the road spatial structure. In some cases, linear segments may abut end to end without an *ifcSpatialJunction*, such as at a defined threshold where the road passes onto a bridge or into a tunnel, or if there is any other discontinuity in paving type, function or layout of the road.

Where a road segment (*ifcSpatialSegment*) traverses a bridge (or tunnel), it must be defined to have a primary association (using *ifcRelAggregates*) with only ONE of those parent entities (*ifcRoad* or *ifcBridge*), and then referenced to the other (using *ifcRelReferencesSpatially*). The use-case for the project model would suggest the most appropriate parent element.

A road typically occupies the full width of the legally-defined road reserve, whether in an urban context (between the front boundaries of adjacent land parcels) or a rural context (where the road reserve may cut a swathe through the landscape, sometimes breaking up public or private land parcels). In all cases, a road is typically broken down in lanes or zones (*ifcSpace*) to serve different functions, e.g. traffic lanes, footpaths, cycleways, stopping zones, safety zones, nature strips, etc.

Geometrically, the spaces associated with a road are commonly linear structures that follow the same alignment (*ifcAlignment*) as the road itself, or may be defined by their own alignment if they follow a path within the same road reserve but are not parallel to the road. Similarly, a space that forms part of a road may take on any shape definition, e.g. an emergency stopping point on highway, defined relative to the main road alignment. These cases are illustrated in Figure 4.

⁶ Source: Austroads (2016): Data Standard for Road Management and Investment in Australia and New Zealand: Version 1, Technical Report AP-T315-16, Austroads, November 2016

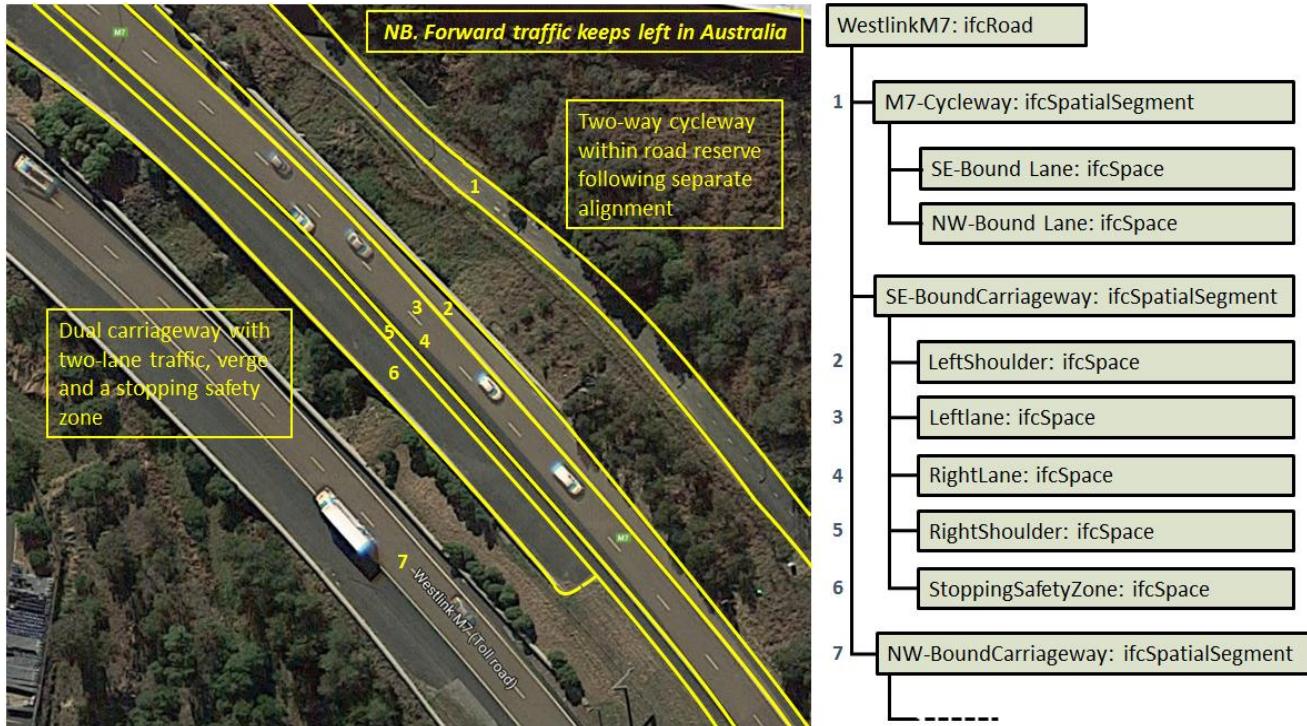


Figure 30 - Spatial breakdown of a typical major road with a dual carriageway and associated cycleway.

It is common for roads to be chained together to form a major route or highway, often assigned a specific name or route number. For example, Route 66 is a famous road that traverses the US, but is made up of several local road entities with their own identities (particularly where it passes through towns or cities). This case would be managed using the *CompositionType* attribute of the abstract superclass *ifcSpatialStructureElement*, where the major road is identified as a COMPLEX entity and each of its constituent parts are identified as PARTIAL entities.

Intersections that form part of a road network are handled spatially as *ifcSpatialJunction* entities and typically encompass the entire road reserve (and therefore include street corners used by pedestrians, etc.). In the same way as a linear segment of a road can be broken down into *ifcSpace* entities, so too, complex road intersections can be broken down into space entities to define the function of each part of the intersection. For example, a typical 4-way urban street intersection may be divided into 7 space entities as illustrated in Figure 5. This proposed breakdown considers asset management need, identifying spaces for maintenance purposes. If the use-case for the project model were more focussed on managing traffic movement, then “TrafficWay” space (labelled 6 in Figure 5) could be further broken down into appropriate turning spaces and the through-traffic zone.

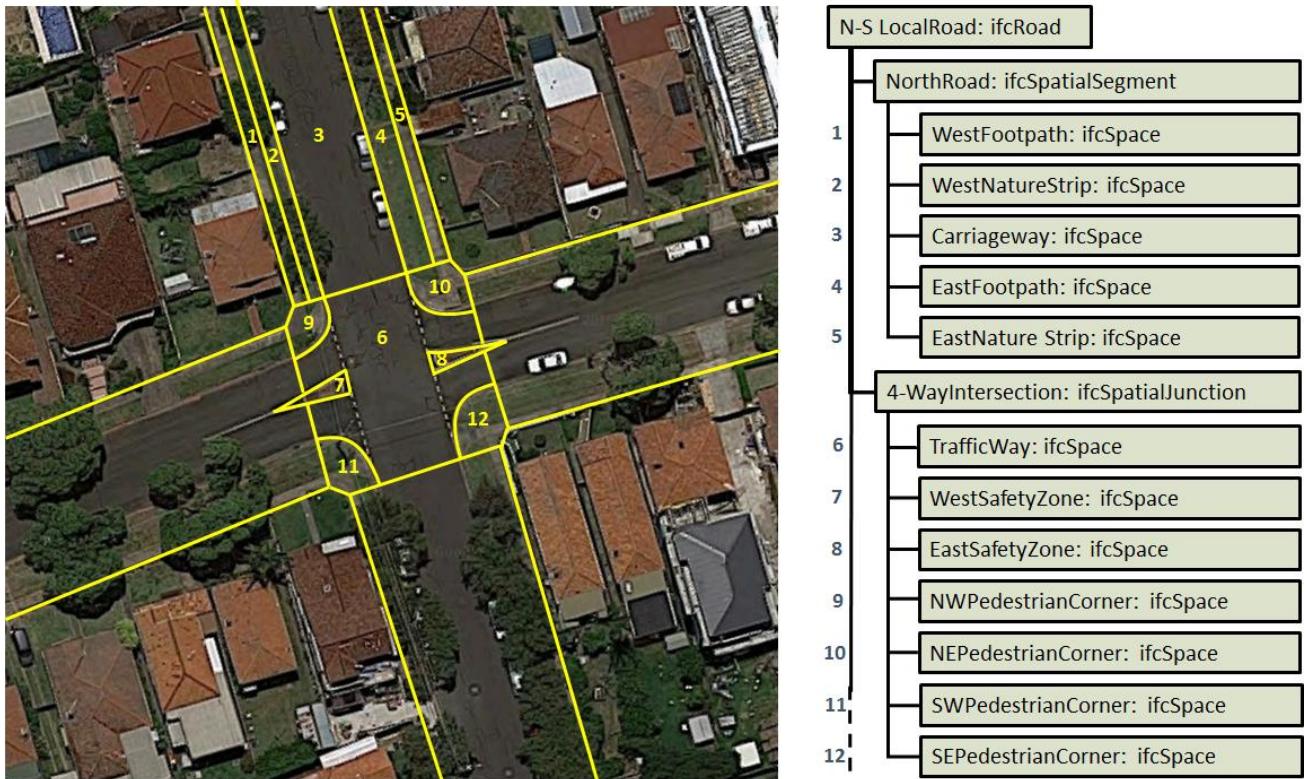


Figure 31 - Typical 4-way local road junction showing a spatial decomposition of one road segment and the intersection

Figure 6 shows a junction between two major motorways that have complex flyovers and on/off ramps. The area of land that is impacted by the motorway junction (the land reserve) is represented as an *ifcSite* entity, so that it can be assigned a unique referenced identity. Within the project model, that site would be part of the land reserve for each motorway, so its *CompositionType* attribute must be set to PARTIAL. Since it is part of two motorways, it becomes another example of a shared spatial element. In this case, its primary association is with the M5 motorway (that runs south and west), so it is part of the COMPLEX *ifcSite* entity that represents the land reserve for the M5 (that association being defined using *ifcRelAggregates*). At the same time, it would be referenced (using *ifcRelReferencesSpatially*) to the COMPLEX *ifcSite* entity that represents the land reserve for the M7 motorway (that runs north). The motorway junction is decomposed into all the pieces of infrastructure that make up the junction: bridges (*ifcBridge*) and tunnels (*ifcTunnel*); through-road segments (*ifcSpatialSegment*, including *ifcSpace* entities as required, defined to the boundary of the *ifcSite* representing the land reserve); on/off ramps (*ifcSpatialSegment*, including *ifcSpace* entities as required); open space and laybys.

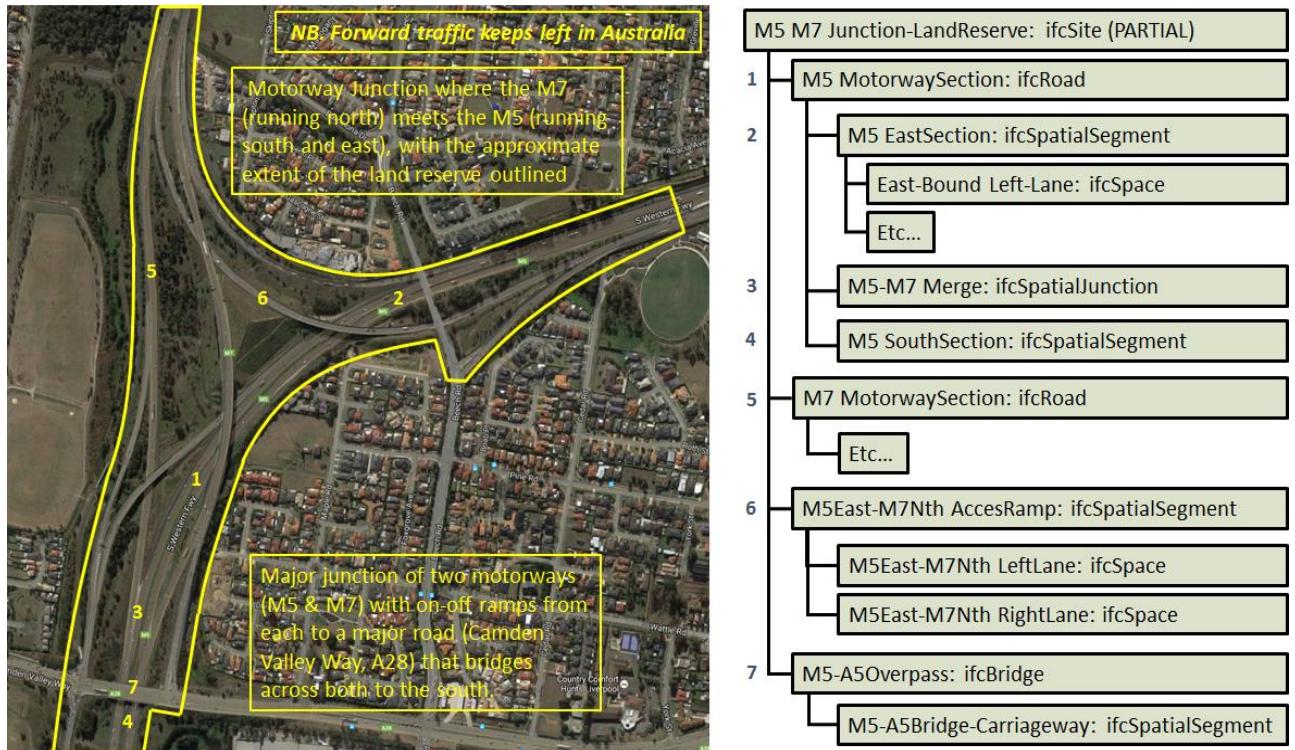


Figure 32 - Spatial structure for a typical complex motorway junction

Bridges and Tunnels

These are spatial entities that are constructed using physical components, but are commonly designed to carry linear infrastructure entities such as roads and railways, but can also include pedestrian paths and cycleways that are not actually associated with a road (e.g. a pedestrian or underpass). They may also include other spatial entities such as viewing platforms, safety zones, etc.

When decomposing a bridge or tunnel horizontally, then it is best to make use of the *CompositionType* attribute and treat bridge spans or tunnel segments as PARTIAL elements that make up a COMPLEX built facility. That permits the decomposition of each PARTIAL bridge or tunnel to address the vertical structure of the facility. In that way, a bridge may be decomposed into functional/structural levels (generally, *ifcCivilStorey* elements) such as foundation piers, one or more bridge decks (including associated sub-structure) and superstructure. Since the primary function of a bridge deck is to serve as a road or railway segment, it is natural to define it as an *ifcSpatialSegment*. Similarly, for tunnels: they can be decomposed vertically into an *ifcSpatialSegment* (to accommodate a road or railway) and an *ifcCivilStory* for services trunking.

Figure 7 (overpage) shows a typical railway tunnel cross-section (source??) with critical spaces shown. The accompanying instance diagram shows how they spaces can be accommodated by the proposed spatial hierarchy.

There is an issue of ownership/responsibility with respect to the way bridges and tunnels are viewed in terms of their function. For example, a road bridge may be fully owned and managed by a road

authority, or it may be owned and operated by another authority and simply accommodates the road that passes over it.

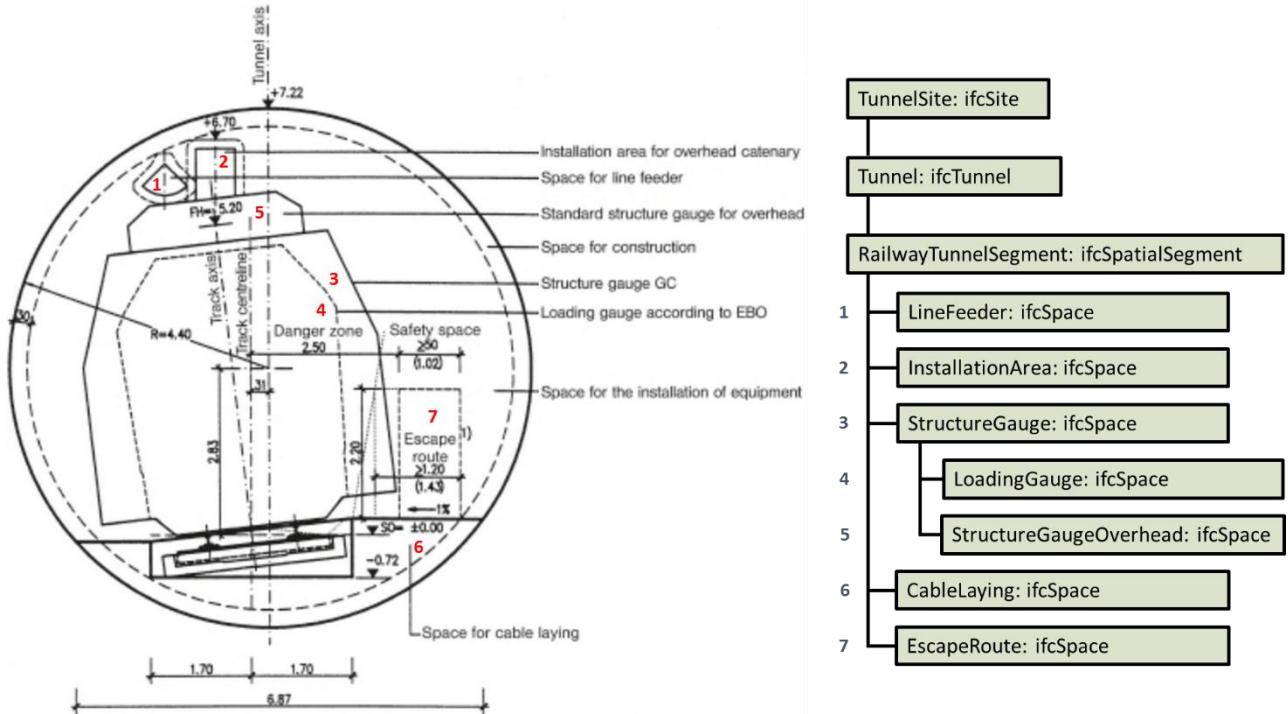


Figure 33 - Spatial decomposition of a railway tunnel

As stated previously, linear infrastructure entities that traverse a bridge or tunnel are best treated as separate spatial entities (typically, *ifcSpatialSegment* entities) that end at the point of access to the bridge or tunnel. This allows those spatial entities to be associated specifically with the bridge or tunnel, which may be important when considering issues of ownership or maintenance responsibility.

Figure 8 shows a case where the road section that crosses the bridge (bridge deck labelled 3 in Figure 8) may be defined as *ifcSpatialSegment* and included in the spatial hierarchy for both the bridge and the road, though only one of those associations can be treated as the primary in order to maintain the hierarchy.

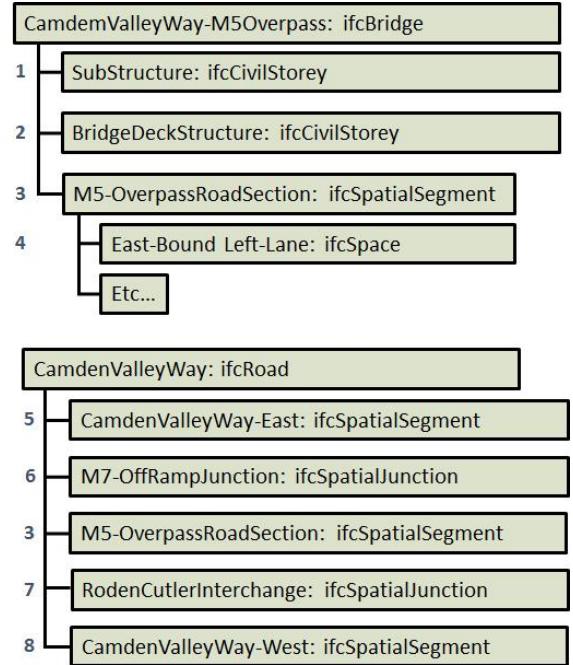
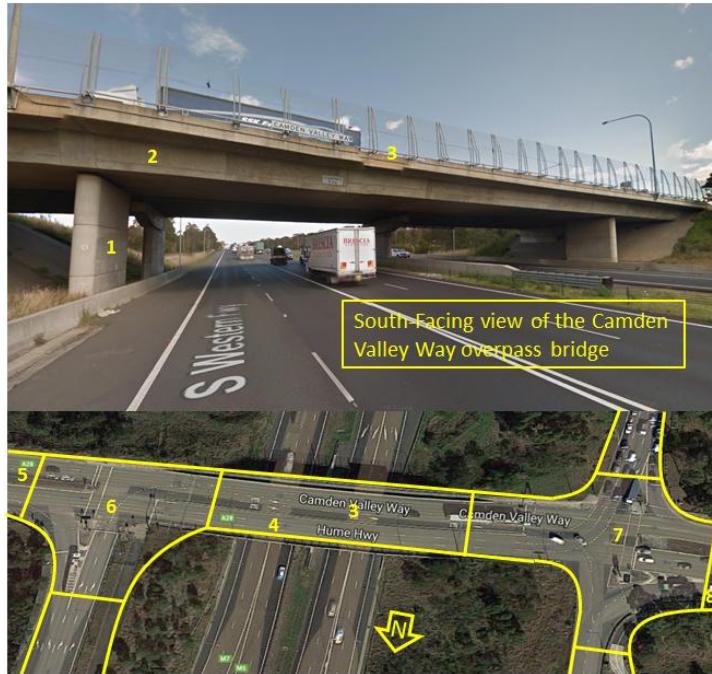


Figure 34 - Spatial decomposition of a typical road overpass bridge

Railways

Railways are distinct from roads because they are much more rigidly organised, with trains being constrained to follow tracks in a coordinated way to move between destinations according to a pre-planned timetable. For that reason, the primary element in the spatial hierarchy is a railway line (*ifcRailway*), having a defined start and end, and supporting train movements in both directions.

Therefore, a single railway line may be considered spatially as a sequence of links (*ifcSpatialSegment*) between stopping points (*ifcSpatialJunction*), typically with separate in-bound and out-bound tracks (*ifcSpace*) and sometimes additional through tracks in either direction to accommodate express services. The railway line has starting and ending stations (though they can form loops that start and end at the same station), and is typically identified in some way for operational purposes (e.g. the Piccadilly Line on London's Underground).

In most cases, these railway lines form part of a larger railway network. A railway station (or stopping point) may be part of more than one railway line and railway segments may be shared by more than one railway line. In a railway network, there may be several "Lines" (e.g.) of that configuration with defined start and end stations, sometimes coming together as parallel lines, merged as a "shared" line or even crossing each other. A station may lie on one or more Lines, providing the opportunity for passengers to change services to a different Line.

- In spatial terms, a single Line may be treated as an instance of *ifcRailway* and is made up of linear segments (*ifcSpatialSegment*) between stations or major branching points (*ifcSpatialJunction*).

- The separate tracks that make up a railway defined in that way may be treated as space entities (*ifcSpace*), so the in-bound and out-bound tracks that form a linear segment between stations would be defined as spaces. Similarly, lengths of track to accommodate express services in either direction may be associated with either an *ifcSpatialSegment* or *ifcSpatialJunction* (e.g. an express track may branch off the main track just before a station and then re-join the main track after the station so the express service can overtake a slower service).
- The spatial boundary of a track entity (*ifcSpace*) is defined by the required width and height clearances.
- Turn-ins and turn-outs are generally identified as space entities (*ifcSpace*) because they typically have switch equipment and signalling associated with them. Similarly, crossover points may be treated as spaces (*ifcSpace*) because they serve a specific function and have equipment associated with them.
- There are many instances of special entities that are associated with railways, such as safety zones and pedestrian access pathways. These are also modelled as spaces (*ifcSpace*) and associated with line segments (*ifcSpatialSegment*) or stations (*ifcSpatialJunctions*).
- The physical structures such as station platforms, signal boxes and plant/equipment occupy space that is associated with and forms part of line segments (*ifcSpatialSegment*) or stations (*ifcSpatialJunctions*).
- A railway station could be treated as COMPLEX *ifcSpatialJunction* or may be treated as an *ifcSpatialZone* entity, as it may encompass a large railway reserve that includes other entities such as train sheds, car parks and commercial/retail facilities.
- Light rail and tramway networks may be modelled in the same way as heavy rail.

Annex 2: Comparison OGC LandInfra and LandXML

OGC LandInfra		LandXML	
CantSpecification	collection of CantEvents along a single linear element for a single Railway	Cant	collection of CantStation elements under a railway Alignment
name	name of the CantSpecification	name	name of the collection of Cants
gauge	measure defining the spacing between the inner faces of the load bearing rails	gauge	measure defining the spacing between the inner faces of the load bearing rails
locatedAlong	the linear element along which the CantSpecification is made	-	Cant is owned by exactly one Alignment
cantEvent	two or more cantEvents which shall be ordered in the direction of the linear element	-	Cant owns a sequence of CantStation
-	cantSide at CantEvent	rotationPoint	left, center or right
CantEvent		CantStation	
eventAt	DistanceExpression defining the linearly referenced location for the CantEvent as a single point along the linear element	station	station value where the cant is defined
appliedCant	difference in elevation between the two rails at the eventAt location	appliedCant	difference in elevation between the two rails at the station location
cantSide	left or right rail (or both) to which the cant shall be applied. The side relates to the positive direction of the linear element	-	(curvature, rotationPoint)
-	(cantSide)	curvature	cw or ccw (around rotationPoint)
-	assumed linear cant change	transitionType	clothoid, bloss, schramm
-	no design parameters such as design speed	speed	design speed at the station location
-		...	+ 8 other attributes

Annex 3: Element breakdown

A pdf version of this figure is available on Google Drive for viewing purposes: <https://goo.gl/7eqtFQ>.

