DISCUSSION PAPER

IFC SUPPORT FOR MODEL-BASED SCHEDULING

VERSION 21.04.2009

AUTHORS

MATTHIAS WEISE, THOMAS LIEBICH (AEC3)
JAN TULKE (HOCHTIEF)
PETER BONSMA (TNO)

The work on this IFC Implementation Guide has been supported by the European Research and Development project InPro, *Open Information Environment for Collaborative Processes throughout the Lifecycle of a Building*. InPro is an Integrated Project, IP 026716-2, under the 6th framework program.

PREFACE

The aim of this paper is to discuss the potential of BIM-based scheduling and its implementation with IFC. Main input to this document had been results and papers related to this topic in context of IFC. Thus, it is also a snapshot of available research whereas trying to be as objective as possible and to do not miss developments in that area. However, there might be further noteworthy developments that are not mentioned here due to lack of knowledge of the authors. If you feel that something needs to be added or corrected, you are kindly asked to report that to one of the authors.

Whereas the state-of-the-art analysis is looking for reusable concepts, specifications and software developments the main purpose of this discussion paper is to implement ideas that have been discussed within the InPro project. These ideas are shortly introduced in the first chapter also providing links for further reading. It provides the basis of our gap analysis and suggested IFC extensions. Decisions are not only presented but also explained showing possible alternatives and discussing their pros and cons. Thus, we hope that it enables to step in the discussion and contribute to interesting developments in the area of BIM-based scheduling.

A full example is discussed in the end of the paper, which is made available as a test file in the IFC SPF format. It is not yet described in all details and does not deal with all possible combinations as it is neither an implementation guide nor a full test bed. Instead, the aim of this example is to provide a more practical basis for further discussions, which also points to open issues that have to be agreed for implementation.

For a next release it is planned to describe the prototypes that have been developed on the basis of this discussion. The software and (parts of) the source code will be made available free of charge and can be used for further developments.

DOCUMENT HISTORY

21.04.2009 First release

- Refinement of proposed extensions based on experiences from prototype developments
- Extensions to provided example
- New document layout

02.12.2008 Second draft

- More detailed discussion of extensions and first draft of provided example

30.08.2008 First draft

- first published version including the state-of-the-art and gap analysis and discussion of possible extensions

CONTENT

1 INTRODUCTION

Scheduling and cost management are design activities that can take noteworthy advantages from the BIM approach. Main improvements are to provide better results, e.g. reliable quantities for procurement and cost management, and to simulate and optimize construction schedules, e.g. with 4D technologies. However, it is quite obvious that there have been and still are serious barriers for adopting the new approach.

A major challenge is to use 3D models, which significantly increase the complexity of design data. Accordingly, complexity of design processes is rising too, but often cannot be justified by advantages taken from model-based working. Main critics for using 4D technologies are:

1) The high effort for creating 4D simulations, in particular to breakdown 3D elements as needed for the construction work as well as to link them with work tasks (Heesom & Mahdjoubi 2004).
2) Poor support in case of design iterations, in particular to identify design changes and to react on them without starting the complete process from scratch again.

A special focus of KP4 is to provide answers to these critics and thus to improve the cost-value ratio for using 4D technologies. It comprises two ideas; (1) reducing the effort for the overall set-up process by using semi-automated linking mechanisms and (2) intelligent data dependencies management that enables to react on design changes. The proposed approach (Tulke 2008) defines the background for this proposal.

Beside conceptual issues of BIM-based working a main drawback of today's 4D developments is the missing interoperability between available software tools. Porkka & Kähkönen (2007) argue that IFC is *"one of most promising candidates for open BIM standard"* and that *"IFC2x3 can be taken to a technological starting point; to be further equipped with 4D case definition. This case definition needs to present the new process taking advantage of the holistic IT infrastructure within construction process."*

1.1 Scope of the Neutral BIM (IFC)

For describing the scope of the IFC data structure, i.e. the mapping of our exchange requirements[1] to a Neutral BIM, a classification of design data is introduced. From a very general viewpoint of data management there are three types of design data that are of interest for design processes, in particular in context of the suggested rule-based linking approach and in case of iterations:

1) Master data that provides design constraints[2] for design processes and is the basis for refinements.
2) Derivation rules that capture design decisions that were made within processes.
3) Derived data that provides the result of derivation rules, e.g. as needed to describe the construction schedules, quantities and costs (typically the data that is handy for further use).

These data types are illustrated in Figure 1 and Figure 2 showing the principle relationships on a simple example and the consequences for our process.

[1] Mentioned exchange requirements have been defined in work package 2 of the InPro project and are embedded in key process 4 (model-based scheduling), which is one out of seven key processes. Main interests of that process are: definition of construction schedules and 4D simulations, refinement of the building structure and access to quantities and cost items.

[2] Master data comprises not only the data that is received from other actors, e.g. from the architect who provides the layout of the building. It is also created within a design process and, if decisions from subsequent design processes depend on this master data, becomes important in case of design iterations.

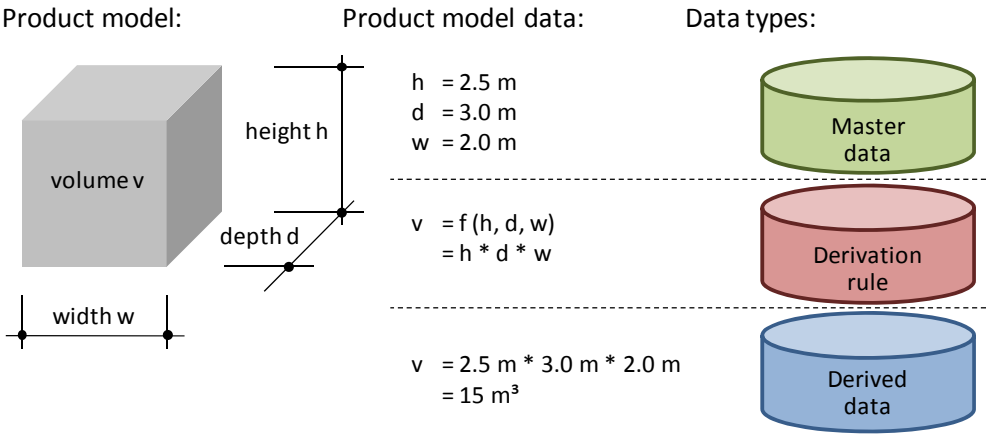


Figure 1: Differentiation of product model data into three data types: (1) the dimension of the cube, (2) the function for calculation of the cube volume and (3) the calculated cube volume. The mentioned data types apply to this example as shown on the right hand side. If the cube volume becomes more important than the cube dimension it can also be changed for instance to (1) master data: v, h, d (2) derivation rule: w = f(v,h,d) = v/(h*d) and (3) derived data: w.

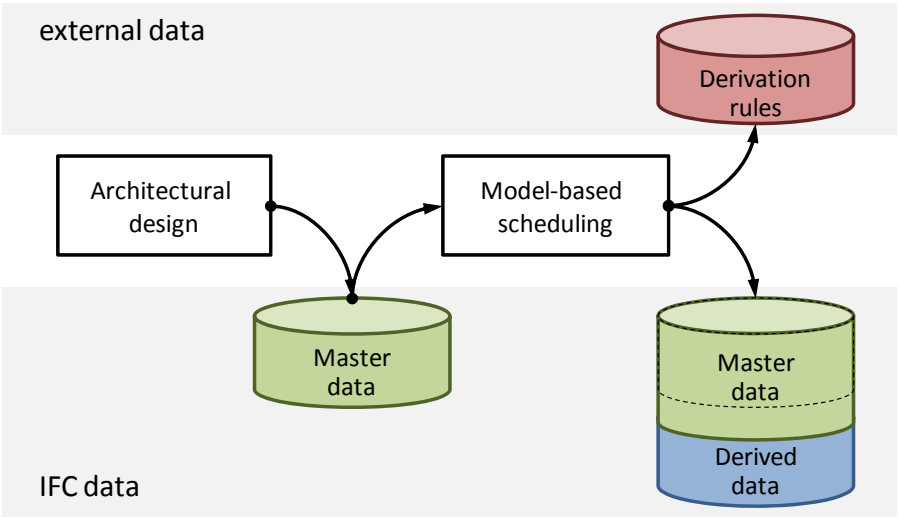


Figure 2: Initial architectural design provides the master data that is used for model-based scheduling. Results are threefold: (1) extension of master data, (2) set-up of derivation rules that in addition to extended master data describe the design decisions of that process and (3) results of derivation rules that are created for communication with other designers (derived data).

Based on these data types there are two options to capture results of design processes. It is possible to store master data either with derivation rules (1+2) or with derived data (1+3). In the first case we are talking about unevaluated models that have advantages in case of iterations. But unevaluated models typically come with noteworthy computational efforts since derivation rules have to be evaluated for getting required design data. It is not (yet) the way of working in the AEC industries and in particular it is rarely used for the purpose of data exchange (Koch & Firmenich 2006). Instead of storing design decisions, IFC is focused on design results that might hold a link to master data but without further details about used derivation rules[3].

[3] The IFC schema wants to avoid redundant design data, which occasionally leads to the use of “derived attributes”. However, such functional dependencies are fixed in the IFC schema and thus are not flexible enough to be used for the suggested approach. As shown in the example the same attribute (cube volume) can be regarded in one project as master data and in another project as derived data.

This principle works fine for storing a particular design solution but provides no support if design changes have to be revised and matched with previous design decisions. Consequently, the IFC data structure is primarily focused on the data types 1) and 3), whereas design decisions have to be handled by responsible actors themselves.

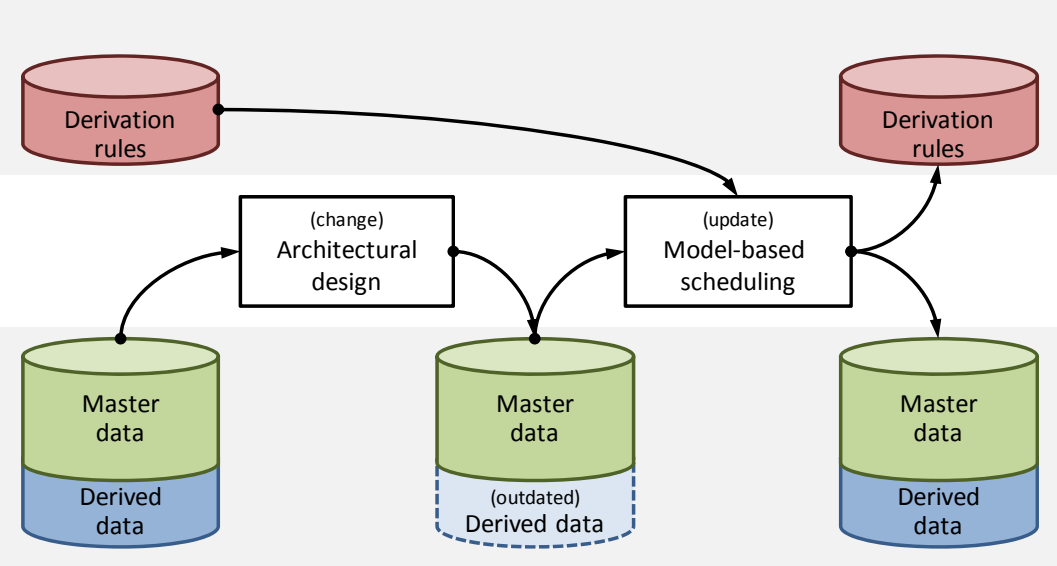


Figure 3: Design iterations are leading to outdated data (inconsistent data) that has to be updated using the derivation rules. Depending on performed data changes it might be necessary to update derivation rules and to adjust other master data. Thus, it is not expected to have a fully-automated update procedure as changes have to be checked for compliance with decisions of former design steps.

According to the approach from Tulke (2008), design decisions should be captured by a 'linking language'[4], which is, at least for the moment, out of scope for IFC extensions. Nevertheless, the aim of supporting model-based scheduling with a Neutral BIM is to capture the link between derived data and externally stored derivation rules.

For implementation of the discussed approach with IFC the following questions have to be answered in the document:

- how to capture required data, in particular what extensions are necessary
- how to differentiate between master data and derived data
- how to link master data and derived data
- how to set references to externally stored derivation rules

Before starting to go into further details it is important to point out that an IFC model always captures a single design solution at a specific point in time of the design process. It is a virtual representation of a specific building design. There is no versioning data, i.e. it is neither possible to store the design history (old design solutions)[5] nor to represent alternative solutions[6] (e.g. to compare prefabricated with cast-in place columns).

[4] The linking language comprises a query language on the basis of attributes and the spatial structure as well as the object splitting functionality.

[5] IFC enables to define the 'owner history' of objects, which manages meta-information of IFC objects but does not store old object versions.

Such additional data management dimensions, if needed and if supported by appropriate client tools, must be provided by the data management environment and are not in scope of the Neutral BIM.

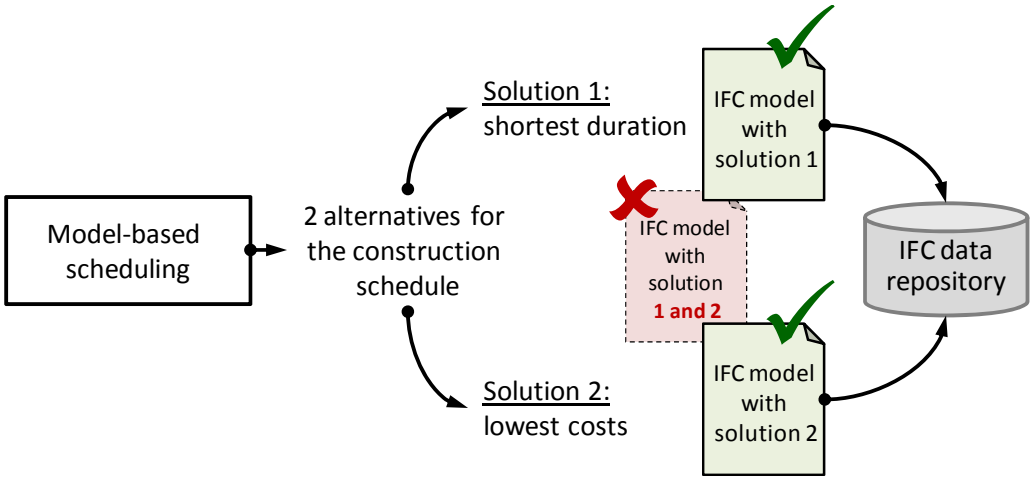


Figure 4: IFC models follow a single version approach so that each alternative requires a separate IFC model

1.2 Supported domains

According to the overview that is provided by Tulke in Houttu (2009) there are three domains that are of main interest for the scheduling processes:

- Architectural design
- Model-based scheduling
- Cost management & Quantity take-off

Beside these domains the interrelationships between them are of vital interest and thus have to be discussed too.

It is worth mentioning that these domains do not imply ownership information or access right settings, i.e. we do not imply that for instance elements of the architectural design domain are used (created, modified, deleted) by architects only. Instead, the suggested differentiation is based on element types. For instance a wall (*IfcWall*) is typically regarded as an element of the architectural design, even if being 'created'[7] by the construction scheduler as needed for 4D simulations and calculation of quantities. A more detailed definition of these domains can be found in the summary of exchange requirements.

[6] Management of alternative solutions requires a mechanism (1) to identify alternatives and (2) to select a proper product configuration, e.g. to see that "wall 1.1" and "wall 1.2" are alternatives and to decide whether "wall 1.1" or "wall 1.2" (or may be both of them) can be use together with "column 1.3". Such mechanism is not available in IFC. However, there might be workarounds such as the decomposition strategy suggested by Tulke et al. (2008), which for instance adjusts the object granularity in such a way that they can be used by alternative construction processes. Accordingly, the IFC model can be linked with different construction processes. Nevertheless, as such workarounds require external linking mechanisms they cannot be seen as an IFC-based management of design alternatives.

[7] 'Derived' might be the better word here. But independent how we call this functionality we are talking about the population of the IFC model that for instance requires to create new wall objects (instances of *IfcWall*) in order to support the splitting functionality. That functionality is particularly used by the construction scheduler to describe building element parts that can be assigned to work tasks of the construction process.

For each domain the following topics are discussed in the subsequent sections:

- Summary of main exchange requirements
- State of the art and gap analysis
- Proposed extensions:
    - using implementation agreements
    - using dynamic extension mechanisms (property sets and proxy elements)
    - using schema extensions
- Implementation guideline(s) to clarify suggested use of IFC and proposed extensions
- IFC examples

The survey and suggested extensions have been made on basis of the new IFC 2x4, which is available as alpha release since June 2008.

1.3 Summary of Exchange Requirements

The following tables summarize requirements of mentioned domains and their interrelationships. It is based on results of the process review in WP2 of the InPro project or were found in related literature. Instead of going into details it identifies main requirements by a concept name, which is shown on the left hand side of the table. On the right hand side of the table there is a short explanation to each concept.

Architectural design – requirement summary	
Concept	**Description**
Building structure	Spatial decomposition of a building (hierarchical structure) that typically consist of: • site • building • building storey • space • building elements
Building elements	All physical elements that are relevant for the construction and quantity take-off
Element splitting	Decomposition of building elements according to construction schedules (leading to a hierarchy of building elements)
Element representation	Location and geometry of building elements
Element properties	Any kind of properties and quantities that describes the building element, e.g. material properties, element dimensions, volumes, areas, etc.
Grids	A set of grid axes that can be used for positioning of elements and the definition of construction zones
Construction zones	Geometrical space within a building that is used for definition of construction sequences (typically the basis for splitting of building elements, i.e. the definition of rules)

Cost management & Quantity take-off – requirement summary	
Concept	**Description**
Cost value	Amount of money that is defined according to a cost type.
Cost quantity	Quantity value that is used for cost calculations. It can either be a number of elements or a measurable value such as length, area, volume or weight.
Cost impact factor	Cost factor that results from construction details or external impacts.
Cost group	Grouping of similar elements (or cost items) that define a cost group.
Cost group properties	Cost related properties that belong to cost groups, e.g. total quantities, total costs, unit costs and impact factors.
Cost item	An item that is relevant for cost and/or quantity calculations.
Cost item properties	Cost related properties that belong to cost items, e.g. quantities, total costs, unit costs and impact factors.
Cost classification	Further classification of costs according to a well-defined classification system.
Cost item hierarchy	Hierarchy of cost items that enables to define a breakdown structure of cost items with arbitrary depth.
Cost schedule	Grouping of cost items for various purposes, e.g. bill of quantities, estimation of construction costs etc.

Model-based scheduling – requirement summary	
Concept	**Description**
Work task	Definition of activities that have to be carried out for building construction.
Construction schedule	Logical collection of work tasks that are needed to fulfill a specific job.
Time constraints	Time-related information about processes, e.g. start, finish, duration, etc.
Logical dependency of work tasks	Logical dependencies define the workflow that describes or constraints the construction process (predecessor/successor relationship of processes).
Hierarchical refinement of work tasks	Detailing of work tasks into sub tasks enabling refinement of the construction schedule in any depth.
4D visualization parameter	The following basic parameters are needed:

	• flag indicating if a work task shall be visualized or not • process type (construct, temporary, demolish, start_period, middle_period, end_period[8]) • element color • element transparency Additionally, there are more sophisticated parameters such as: • 'scaling' (time-dependent growing/completion of elements) • transportation path of elements • surrounding properties (GIS data, background images, etc.)
Visualization templates	Templates that can hold 4D visualization parameters and can be assigned to processes.
Responsible actor and required resources	Information that enables calculation of task duration. Main aspects are element quantities but also used machinery and human resources (see also link to cost management/QTO).

Cross-model links – requirement summary	
Concept	**Description**
Link between work tasks and building elements	It is needed for 4D simulation of the construction process and the calculation of quantities.
References to splitting rules/queries	Splitting rules are a way to speed up the definition of construction schedules and 4D simulations. Thus, they capture the design intend of the scheduler. In case of design changes they enable to redo the element splitting.
Link between work tasks and quantity/cost information	Quantity and cost information as calculated by the cost manager are needed to estimate task durations.
Link between grids (axis) and construction zones	Capture the definition of construction zones, which might be based on grid axes.
Containment of building elements in construction zones	Building elements that are (completely) located within a construction zone.
Link between building elements and quantity/cost information	Quantity and cost information are based on building elements as planned by the architect. Quantities for cost calculation and building element quantities depend on each other but are often not identical.

[8] Further information about suggested process types can be found in Tulke et al. 2008.

2 STATE-OF-THE-ART AND GAP ANALYSIS OF DISCUSSED DOMAINS

The subsequent analysis discusses available work, refers to main IFC classes that are used to represent required domain information and finally presents identified gaps. Main sources of the analysis are the official IFC documentation from the IAI including the schema definition (IFC 2x3, IFC 2x4), implementation guidelines (Liebich 2004), model view definitions (Hietanen 2006, MVD), implementation agreements (ISG), reported software certification results (Kiviniemi et al. 2008) and available information delivery manuals (Wix 2006, IDM). It also takes into account research results that are reported in various publications. The analysis provides an overview of available work and gives references for further reading.

Before going into details of research projects and specifications a short comment to the used notations is necessary. Since highly specialized applications are not able to completely implement a domain-spanning BIM the software companies very soon realized that they have to agree on the scope of BIM interfaces. They also realized that in context of specific data exchange scenarios the BIM specification needs to be clarified to be processed more efficiently and to avoid misunderstandings. This is the general background for above mentioned IFC-related specifications. However, different notations have been used to describe the scope of software implementations (view). The following notations were found in the state-of-the-art analysis:

- Excel sheets, which is a straight forward documentation of agreements for the Coordination View that points back to first IFC implementations.
- Various proprietary notations such as the VTT mapping definition that were used for describing project results and decisions.
- Aspect Cards (Karstila & Serén 2005), which is a notation that was developed in the ProIT project for supporting BIM implementations.
- Information Delivery Manual (Wix 2006), which is a BIM development methodology that puts special focus on processes for data exchange specifications.
- Model View Definition format (Hietanen 2006), which is a notation for supporting implementation of IFC data exchange scenarios. It comprises the definition of IFC subsets and further implementation agreements.

2.1 Architectural design

IFC status: The architectural design domain refers to an IFC subset for which most commercial CAD solutions and practical experiences are available. The implementation is internationally agreed by the so called *Coordination View* that identifies all relevant parts of IFC and comprises further implementation agreements and hundreds of test cases. The Coordination View is the basis for certification of IFC-enabled applications that need to read or write architectural design data. More detailed information can be found at the website of the Implementation Support Group of the IAI (ISG[9]), where the IFC subset (IFC2x3 Extended Coordination View[10]) and all implementation agreements[11] are documented and the test cases[12] for software certification can be downloaded.

[9] http://www.iai.hm.edu

[10] http://www.iai-tech.org/products/ifc_specification/ifc-view-definition/coordination-view/summary

[11] http://www.iai.hm.edu/how-to-implement-ifc/Implementer-Agreements

IFC subset:

The requirements from model-based scheduling regarding the architectural design are limited to coordination purposes and thus can be fulfilled by "shared" IFC functionalities that are mainly defined in the *IfcProductExtension* schema and the *IfcSharedBldgElements* schema. Required concepts are represented by the following IFC entities:

Architectural design – concept mapping		
Concept	**IFC2x3**	**IFC2x4 alpha**
Building structure	*IfcProject,* *IfcSite,* *IfcBuilding,* *IfcBuildingStorey,* *IfcSpace,* *IfcBuildingElement* (subtypes) *IfcRelAggregates* (definition of the spatial structure), *IfcRelContainedInSpatialStructure* (containment of building elements in the spatial structure)	similar to IFC2x3
Building elements	*IfcBuildingElement* (subtypes), *IfcBuildingElementProxy*	some new entity definition for "standard" elements, e.g. *IfcBeamStandardCase*
Element splitting	*IfcRelNests*	similar to IFC2x3
Element representation	*IfcProduct.ObjectPlacement* and *IfcProduct.Representation*	similar to IFC2x3
Element properties	*IfcElementQuantity* and *IfcPropertySet* either directly connected to building elements through *IfcRelDefinesBy-Properties* or indirectly connect via building element types (*IfcTypeObject*/ *IfcType-Product*), which are associated to building elements through *IfcRelDefinesByType*; *IfcMaterial, IfcRelAssociatesMaterial*	similar to IFC2x3
Grids	*IfcGrid, IfcGridAxes* (definition of 2D grids), *IfcRelContainedInSpatialStructure* (containment of the grid in the spatial structure)	similar to IFC2x3
Construction zones	-	*IfcSpatialZone*

[12] http://www.iai.hm.edu/how-to-implement-ifc/certification/test-cases

Gap analysis and limitations:

It is worth mentioning that architectural design data exported by IFC-enabled CAAD applications provide the basis for model-based scheduling. Accordingly, available CAAD applications are seen as major sender and receiver of BIM data. In that context it must be differentiated between the capabilities of (1) IFC and (2) the Coordination View that is implemented by software vendors agreeing on a subset of IFC. These implementation agreements are interesting for data exchange scenarios using available CAAD applications.

There are only minor remarks for the architectural design domain regarding IFC capabilities in context of model-based scheduling and the suggested approach. Even if there are minor gaps (see table below) it can be concluded that the IFC2x4 schema provides all main features for model-based scheduling.

Architectural design – IFC schema gaps	
Concept	**Identified gaps of the IFC schema**
Element splitting	The IFC nesting functionality enforces a hierarchical refinement structure. Hierarchical refinement as needed for scheduluing might collide with other refinement purposes.
Grids	Grids are currently limited to 2D grids only, but can be defined for different levels within a building and can be contained in a building storey or any other spatial structure element. Furthermore, *IfcGrid* is a subtype of *IfcObject* and thus is capable to use dynamic extension mechanisms, i.e. property sets and element quantities. For visualization purposes the style of grid axis (line type, color and width) can be defined by using *IfcAnnotationCurveOccurrence*. Further details about definition of grids, property sets and element quantities can also be found in related MVD concept descriptions, in particular “Grid”, “Single Value Property Definition” and “Simple Quantity” (see Appendix)
Construction zones	Up to IFC2x3 there is no equivalent IFC entity for representation of construction zones. In IFC2x4 the entity *IfcSpatialZone* is introduced, which is *“…a non-hierarchical and potentially overlapping decomposition of the project under some functional consideration.”* Even if it is not listed in the predefined type enumeration (*IfcSpatialZone.PredefinedType -> IfcSpatialZoneTypeEnum*) the definition fulfills requirements of construction zones. It is a subtype of *IfcSpatialElement* and thus inherits a number of interesting functionalities, such as: • unique identification, setting of user defined name and description • zone hierarchy through nesting relationships • placement and geometry definition • use of property sets and element quantities • reference to building elements

For communication with IFC-enabled CAAD applications IFC data roundtrip[13] becomes interesting, in particular in case of design iterations. If there is no data sharing environment that is capable to manage an IFC data repository as show in Figure 5, the issues that are shown in the table below might result in data exchange errors (most likely wrong interpretation of design data) or even data loss. The reasons for these issues are the implementation agreements of the Coordination View, which restrict the IFC schema definition.

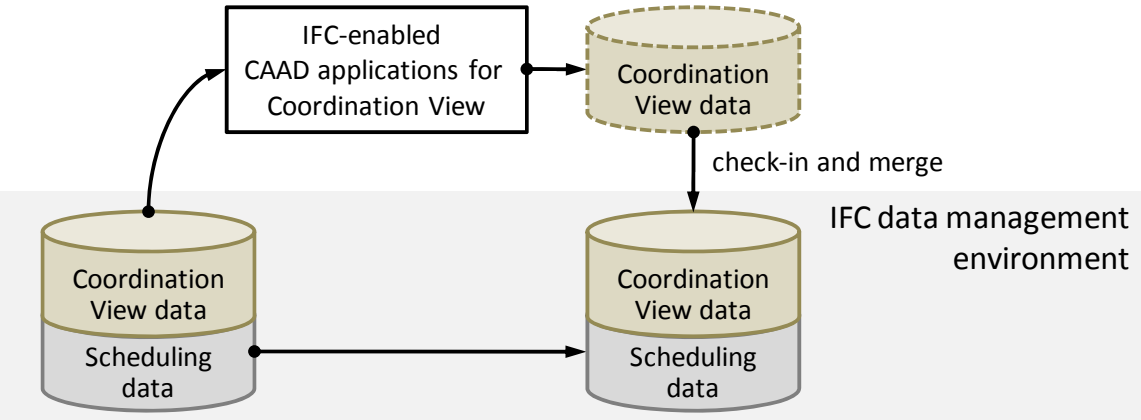


Figure 5: CAAD applications that have implemented the IFC Coordination View are for instance not able to handle scheduling data. Consequently, construction schedules will be lost after IFC-export of CAAD applications. Therefore, it is necessary that an IFC data management environment compensates the data loss through a merge operation.

Architectural design – remarks to IFC2x3 implementations (Coordination View)	
Concept	**Remarks to the Coordintation View**
Element splitting	• Visualization of building elements; the implementation agreement #CV-2x3-119 is saying: *"geometry for decomposed elements shall either be given at the element container or at the element part level"* • Decomposition depth of building elements; the implementation agreement #CV-2x3-121 is saying: *"decomposed elements shall have a maximum of 1 level decomposition depth"*
Element properties	Dynamic extension mechanisms are limited to predefined properties, e.g. *Pset_XxxCommon* and *BaseQuantities*. Other properties and quantities are most likely lost after IFC data roundtrip.
Grids	Grids are rarely used in IFC-based data exchange so that there is only limited experience.
Construction zones	Not yet supported
Unique identification	Unique identification of relationship objects (subtypes of *IfcRelationship*) is not guaranteed

[13] Data roundtrip includes import and later export of design data assuming that there is no data loss due to the interface implementation.

Architectural design – gap analysis summary			
Concept	**IFC2x3**	**IFC2x4 alpha**	**Coordination View (2x3)**
Building structure	++	++	++
Building elements	++	++	++
Element splitting	+	+	o
Element representation	++	++	+
Element properties	++	++	-
Grids	+	+	+
Construction zones	--	+	--
Unique identification	+	+	o

2.2 Model-based scheduling

IFC status: The model-based scheduling domain is frequently discussed for implementation and a couple of prototype developments can be found. The mapping of construction schedules to IFC is discussed in Porkka & Kähkönen (2007), Serén & Karstila (2001) and Froese & Yu (1999), to mention main contributions in the field of IFC-based construction schedules. They also provide additional references to further research. In the Aspect Card Library (Karstila & Serén 2005) scheduling is discussed in the concept ProIT-140.

IFC subset:

Requirements from model-based scheduling are mainly covered by the *IfcProcessExtension* schema, which belongs to the core layer of IFC.

Model-based scheduling – concept mapping	
Concept	**IFC2x3**
Work task	*IfcTask*
Construction schedule	*IfcWorkSchedule*
Time constraints	*IfcScheduleTimeControl* assigned to work tasks through *IfcRelAssignsTasks*
Logical dependency of work tasks	*IfcRelSequence*
Hierarchical refinement of work tasks	*IfcRelNests* also for hierarchical refinement of construction schedules
4D visualization parameter	*IfcRelAssignsToProcess* (link between processes and building elements) Based on the link between processes and building elements there are additional information available that might be interesting for 4D simulations: • element representation (subtypes of *IfcPresentationStyle*, assigned to *IfcGeometricRepresentationItem's* via the *IfcPresentationStyleAssignment* through an intermediate *IfcStyledItem* or one of its subtypes)

	• *IfcScheduleTimeControl* (planned and actual times of processes) • *IfcRelSequence* (workflow) • *IfcPropertySet* and *IfcElementQuantity* (individual extensions to work tasks, construction schedules and elements)
Visualization templates	Possible solution depends on way of describing 4D parameter.
Responsible actor, required resources	*IfcResource* and link to cost management/QTO domain

Gap analysis and limitations:

There are no major gaps for the definition of construction schedules. There are issues about proper display of work tasks[14], e.g. to know about the work task order if shown in the list view of MS Project, and about mapping between different ID concepts, e.g. to preserve proprietary IDs of objects. Furthermore, there are no agreements how to use available IFC functionality for definition of 4D visualization parameters.

There are no commercial applications that have implemented the process extension schema of IFC[15]. Testing and further use might be possible with the prototype implementation Visual Product Chronology (VPC) that is described in Kähkönen K. & Leinonen J. (2003). However, it was available for this study so that we do not know its current status.

Model-based scheduling – gap analysis summary	
Concept	**IFC2x3**
Work task	++
Construction schedule	++
Time constraints	++
Logical dependency of work tasks	++
Hierarchical refinement of work tasks	++
4D visualization parameter and templates	O
Display and update mechanism (interesting for user interactions)	O

2.3 Cost management & Quantity take-off

IFC status: The cost management & quantity take-off domain is further described in the Cost Modeling IDM, which is divided into 5 stages. Whereas the *Process Map* and associated *Exchange Requirements* have to be merged with KP4 and KP7 all related *Functional Parts* (e.g. *fp_model_cost_item, fp_associate_cost* and

[14] Such issues are not mentioned in exchange requirements, because they are not needed to fulfill the tasks.

[15] IFC-based data exchange capabilities of commercial packages are limited to geometry data.

fp_model_cost_schedule) provide a reusable mapping specification to IFC 2x2. As the cost modeling domain has not been changed since IFC2x2 the IDM specifications can be used without changes. In the Aspect Card Library (Karstila & Serén 2005) it is discussed in the concepts ProIT-101, 102 and 103.

IFC subset:

The requirements from model-based scheduling regarding the cost management & quantity take-off are mainly defined in the *IfcSharedMgmtElements* schema. Required concepts are represented by following IFC entities:

Cost management & Quantity take-of – concept mapping	
Concept	**IFC2x3 (IFC2x2)**
Cost value	*IfcCostValue* assignment to cost items (and objects/groups) via *IfcRelAssociateAppliedValue* dependencies between cost values are defined via *IfcAppliedValueRelationship*
Cost quantity	*IfcElementQuantity* assignment to *IfcCostItem, IfcGroup* or building elements (sub-types of *IfcElement*) via *IfcRelDefinesByProperties*
Cost impact factor	*IfcPropertySet* assignment to cost items, cost groups, elements and cost schedules via *IfcRelDefinesByProperties*
Cost group	*IfcGroup* grouping of (building) elements via *IfcRelAssignsToGroup;* assignment of cost items via *IfcRelAssignsToControl*
Cost group properties	*IfcRelDefinesByProperties* (assignment of *IfcElementQuantity* and *IfcPropertySet* with a cost group)
Cost item	*IfcCostItem*
Cost item properties	*IfcRelDefinesByProperties* (assignment of *IfcElementQuantity* and *IfcPropertySet* with a cost item)
Cost classification	*IfcClassification,* *IfcClassificationReference,* *IfcRelAssociatesClassification*
Cost item hierarchy	*IfcRelNests* also used for nesting of cost schedules
Cost schedule	*IfcCostSchedule* assignment of cost item to a cost schedule via *IfcRelSchedulesCostItem*

Gap analysis and limitations:

There are no gaps from the perspective of construction schedules. But to our knowledge there are no commercial or prototype applications that are using IFC for cost management & cost related quantity take-off. However, there are CAAD-applications that are able to export so called “base quantities” of building elements. Such quantities provide a basis for cost-related quantities, which might be grouped to cost elements, multiplied with various impact factors or otherwise processed according to local cost management rules.

Cost management & Quantity take-off – gap analysis summary	
Concept	**IFC2x3 (IFC2x2)**
Cost value	++
Cost quantity	++
Cost impact factor	++
Cost group	++
Cost group properties	++
Cost item	++
Cost item properties	++
Cost classification	++
Cost item hierarchy	++
Cost schedule	++

2.4 References between different domains

IFC status: As there are no commercial IFC-enabled applications for cost management & quantity take-off and construction scheduling, the “cross-model links” are not yet of main interest. Discussions about “cross-model links” are limited to relationships between processes (tasks) and products (building elements), which are needed for 4D-simulations and calculation of quantities.

IFC subset:

The requirements from model-based scheduling regarding the links to architectural design and cost management & quantity take-off are mainly based on general functionalities that are defined in the *IfcKernel* schema. Required concepts are represented by following IFC entities:

Cross-model links – concept mapping		
Concept	**IFC2x3**	**IFC2x4 alpha**
Link between work tasks and building elements	*IfcRelAssignsToProcess (IfcRelAssignsToProcess.RelatingProcess = IfcTask; IfcRelAssignsToProcess.RelatedObjects = IfcBuildingElement)*	similar to IFC2x3
References to splitting rules/queries	-	-

Link between work tasks and quantity/cost information	*IfcRelAssignsToProcess* and/or *IfcRelAssignsToControl*	similar to IFC2x3
Link between grids (axis) and construction zones	Grids are used for the placement of building elements, including spatial structure elements. It is defined by using intersection of two grid axes. Further use cases are not known.	Described functionality of IFC2x3 can be used for construction zones too.
Containment of building elements in construction zones	*IfcRelContainedInSpatialStructure*	*IfcRelContainedInSpatialStructure* can only be used for site, building, building storey and space (not for spatial zone as it is not subtype of *IfcSpatialStructureElement*) *IfcRelReferencedInSpatialStructure* must be used for construction zones too.
Link between building elements and quantity/cost information	*IfcRelAssociatesAppliedValue (IfcRelAssociatesAppliedValue.RelatingAppliedValue = IfcCostValue; IfcRelAssociatesAppliedValue.RelatedObjects = IfcObjectDefinition or IfcPropertyDefinition)* *IfcRelAssignsToControl (IfcRelAssignsToControl.RelatingControl = IfcCostItem or IfcCostSchedule; IfcRelAssignsToControl.RelatedObjects = IfcObjectDefinition)*	similar to IFC2x3

Gap analysis and limitations:

There are a lot of general IFC functions that can be used to define “cross-model links”. However, since there are several options to use these general IFC functions it is mainly a lack of implementation agreements leading to open questions about definition of “cross-model links”.

Cross-model links – Missing implementation agreements or IFC schema gaps	
Concept	**Identified gaps**
References to splitting rules/queries	Splitting of objects is defined through *IfcRelNests*, but is limited by implementation agreements of the Coordination View (see architectural design – element splitting) and does not define additional dependency management information.
Link between work tasks and quantity/ cost information	There are two relationships that can be used for linking work tasks to quantity/cost information. Thus, an implementation agreement should clarify how to define this link.

Link between grids (axis) and construction zones	Except for definition of local placement there are no agreements about further use of grid axes.
Link between building elements and quantity/cost information	There are two relationships that can be used for linking building elements to quantity/cost information. Thus, an implementation agreement should clarify how to define this link.

Cross-model links – gap analysis summary		
Concept	**IFC2x3**	**IFC2x4 alpha**
Link between work tasks and building elements	++	++
References to splitting rules/queries	-	-
Link between work tasks and quantity/cost information	o	o
Link between grids (axis) and construction zones	-	-
Containment of building elements in construction zones	-	o
Link between building elements and quantity/cost information	o	o

3 PROPOSED EXTENSIONS

From the viewpoint of our use case scenario IFC fulfills almost all requirements for architectural design and cost management & quantity take-off. Only minor extensions are needed to support construction scheduling, in particular 4D simulation. Most questions arise for implementation of “cross-domain links”, which is an import concept of the suggest approach.

3.1 Model-based scheduling

Main extension issue is the visualization of construction schedules. As shown in the gap analysis there is a lot of workflow information that can be used for 4D simulations, even if not intentionally defined for visualization purposes. Missing information and proposed extensions are discussed in the following table.

Model-based scheduling – proposed extensions	
Concept	**Possible solution**
4D visualization parameter	It is suggested to use the concept of property sets (*IfcPropertySet*) for the following agreements: PropertySet name = **4DVisualizationParameter** The property set shall be attached to work tasks (*IfcTask*). The following properties can be defined: • **IsVisible**: *IfcPropertySingleValue/IfcBoolean* • **ProcessType**: *IfcPropertyEnumeratedValue* possible values: *CONSTRUCT* *TEMPORARY* *DEMOLISH* *START_PERIOD* *MIDDLE_PERIOD* *END_PERIOD* *PERIOD* • **RGB_Red**[16]: *IfcPropertySingleValue/IfcInteger* with: 0 <= red_value < 256 • **RGB_Green**: *IfcPropertySingleValue/IfcInteger* with: 0 <= green_value < 256 • **RGB_Blue**: *IfcPropertySingleValue/IfcInteger* with: 0 <= blue_value < 256 • **Transparency**: *IfcPropertySingleValue/IfcInteger* with: 0 <= transparency_value <= 100 Further details can be found in the instantiation diagram show in Figure 7.

[16] IFC provides an entity for definition of RGB colours (*IfcColourRGB*), but it cannot be used for 4D visualization.

Display and update mechanism for specific tools, e.g. MS Project	It is suggested to use the concept of property sets (*IfcPropertySet*) for the following agreements: PropertySet name = **MSProjectSupport** The property set shall be attached to work tasks (*IfcTask*). The following properties can be defined: • **uniqueNumber**: *IfcPropertySingleValue/IfcInteger* (must be unique, otherwise no ordering possible) • **PSP_code**: *IfcPropertySingleValue/IfcLabel* Further details can be found in Figure 6.

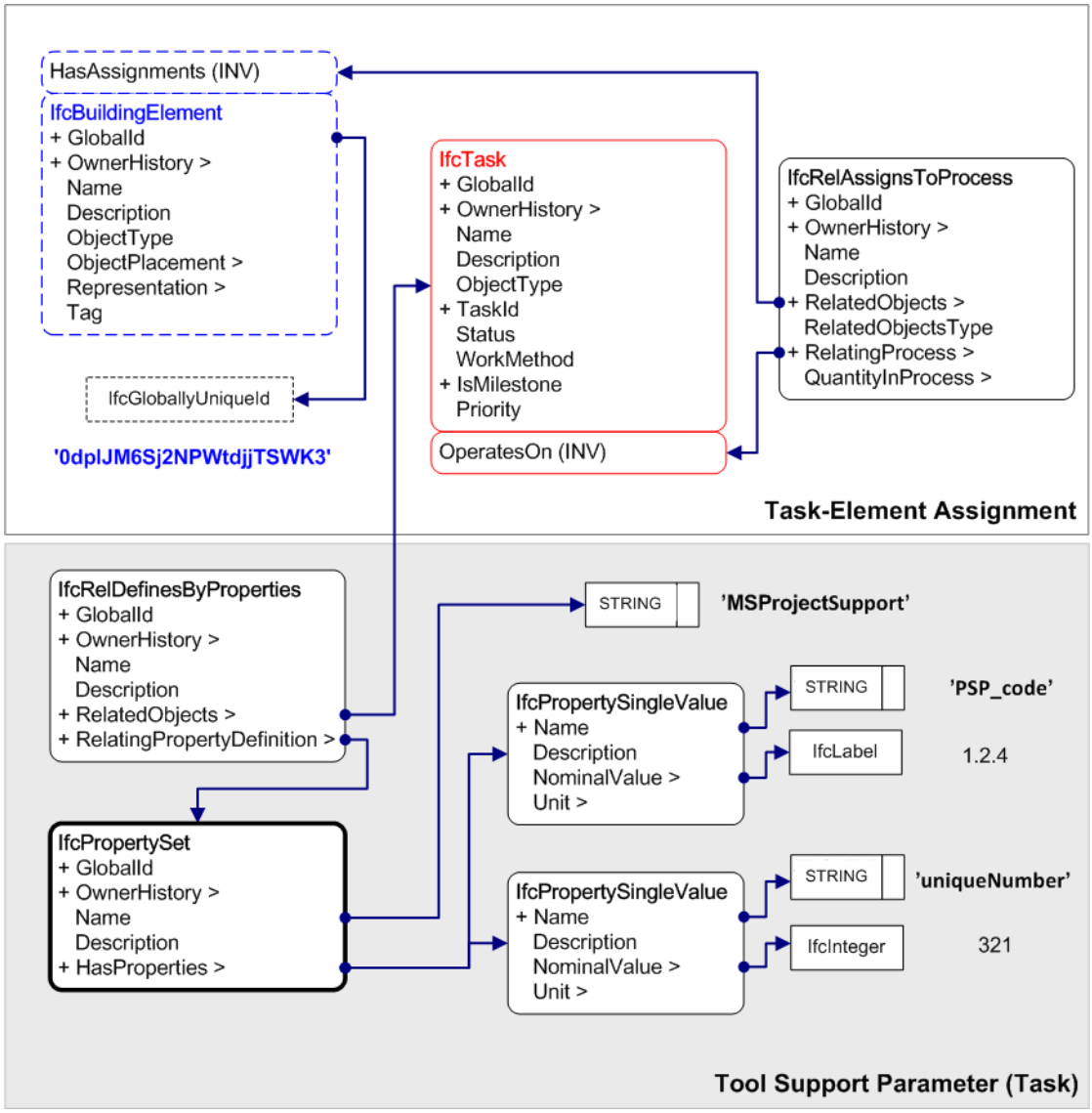


Figure 6: Instantiation diagram of suggested Tool Support Parameter for tasks, e.g. as needed by MS Project to support import and export of IFC model data.

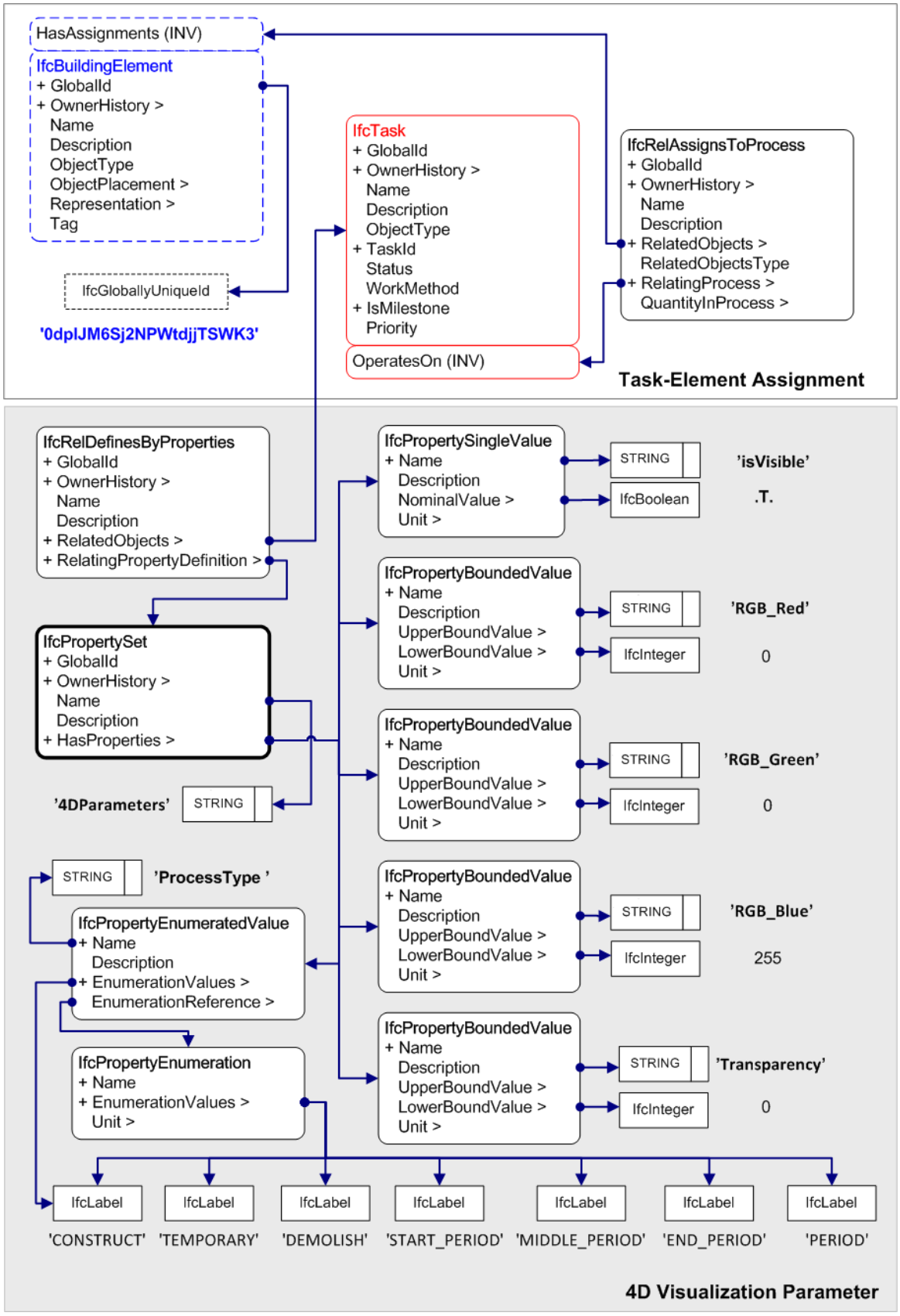


Figure 7: Instantiation diagram of suggested 4D Visualization Parameter.

3.2 References between domains

IFC is an object-oriented model making heavy use of the inheritance concept in order to be maintainable but also flexible. This advantage comes with the drawback that more general entity definitions are flexible enough to deal with new requirements but sometimes offer several options for representing the data. However, possible solutions may come with drawbacks or would require minor changes to the IFC schema. Thus, if there are alternative solutions they are presented with their pros and cons. On the basis of further discussions we finally vote for one solution.

Cross-model links – proposed extensions	
Concept	**Possible solutions**
References to splitting rules/queries	There are several options that require further discussion: 1.) Use references to external documents (*IfcDocumentReference, IfcRelAssociatesDocument*) 2.) If element splitting is defined by construction zones it is possible to implicitly describe the splitting rule through assignment of building elements to construction zones via *IfcRelAggregates, IfcRelReferencedInSpatialStructure* or *IfcRelContainedInSpatialStructure*. 3.) Use of constraints, which are defined by *IfcConstraint* and can be linked to IFC objects via *IfcRelAssociatesConstraint.*

1.) Reference to external documents

It is not required that derivation rules can be stored in IFC, but it is necessary to store a link to the rule or at least the rule name. If a derivation rule is defined in a separate document a rule reference can be stored using the “document reference concept” of IFC. Such document reference can be associated with building elements (parent and child elements) and other objects that are involved in the definition of splitting rules, e.g. construction zones or grids. But not all IFC object types can be associated with such reference. For example, due to a consistency rule (WR 21) it is not possible to use relationship objects (e.g. *IfcRelNests*) and resource objects (e.g. *IfcGridAxis*). If such objects are involved in the definition of derivation rules they have to be identified by their identifier (e.g. *IfcRelNests.GlobalId*) or name (e.g. *IfcGridAxis.AxisTag*). The suggested use of external document references is illustrated in Figure 8.

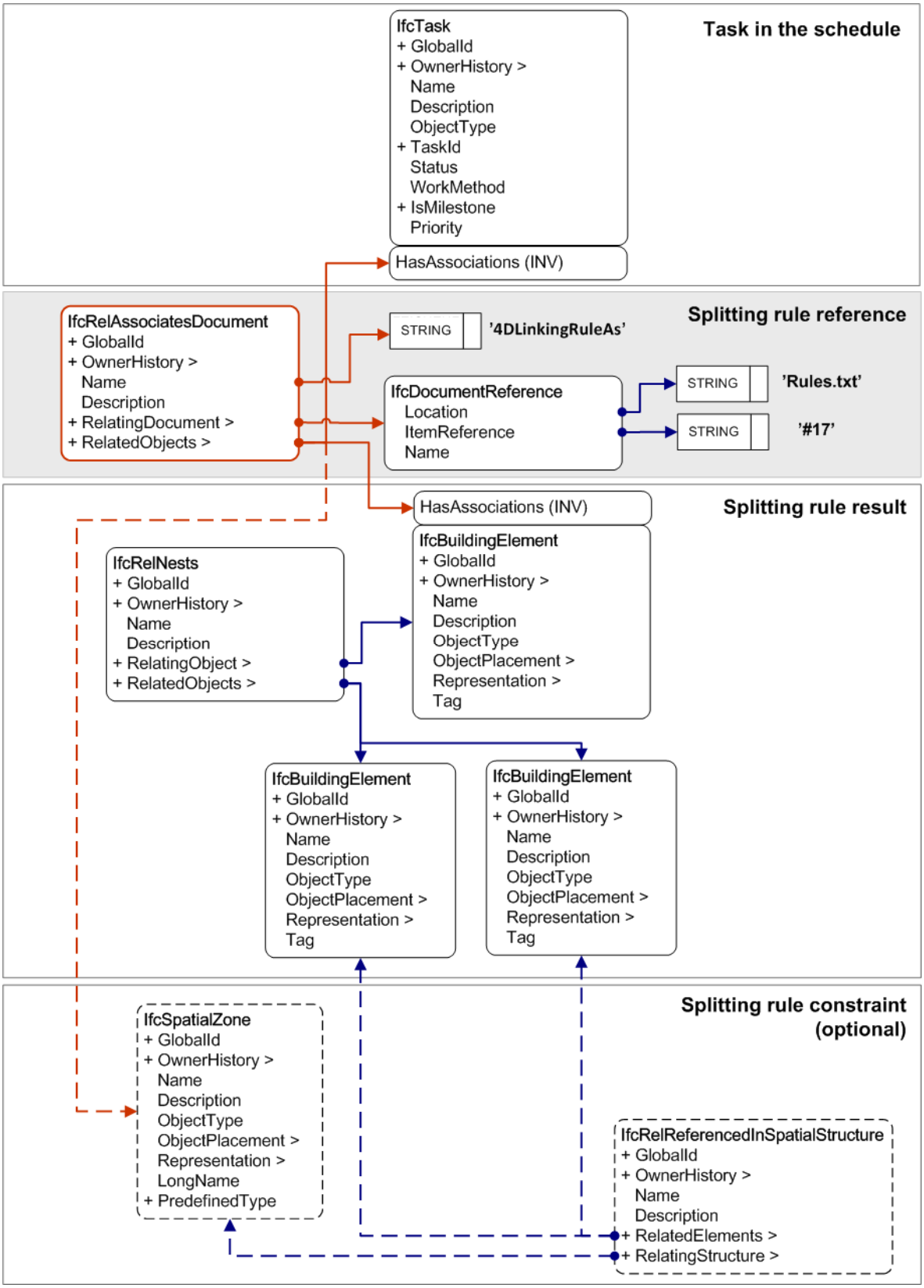


Figure 8: Instantiation diagram of a document reference that defines a link to a splitting rule (option 1).

2.) Implicit definition of splitting rules through geometric boundaries of construction zones

In cases there element splitting can be defined by the geometry of construction zones a "containment relationship", which is defined between the construction zone and contained child elements, actually defines the splitting rule. It might be seen as a smart way of defining splitting rules, but there are some problems for defining the "containment relationship" in IFC:

- *IfcSpatialZone* is not a subtype of *IfcSpatialStructureElement* and thus cannot be used with *IfcRelContainedInSpatialStructure.* This would require changing *IfcRelContainedInSpatialStructure.RelatingStructure* from *IfcSpatialStructureElement* to *IfcSpatialElement.*
- Aggregation of elements (*IfcRelAggregates*) is a hierarchical relationship and thus might collide with the spatial structure of the building. Such collision arises if a building element is contained in a building storey and a construction zone[17].
- Besides the containment relationship and the element aggregation there is another relationship (*IfcRelReferencedInSpatialStructure*) that can be used to link building elements to construction zones. But contrary to above mentioned options it does not define any restriction regarding the element hierarchy and therefore might also be used to reference elements that are only partially contained in a construction zone. Accordingly, such work-around for defining splitting rules might lead to misinterpretations if no further implementation agreements are made.

Due to mentioned restrictions and possible misinterpretation an implicit coding of splitting rules seems to be an inadequate solution to the problem.

3.) Use of the IfcConstraintResource schema

The *IfcConstraintResource* schema defines functionalities that are interesting for the definition of derivation rules. A derivation rule might be seen as a constraint (e.g. as a design intent) and thus would fit to the purpose of *IfcConstraint.* Similar to document references an *IfcConstraint* can be associated with subtypes of *IfcObject*, i.e. building elements and construction zones. Additionally, it might also be usable for describing derivation rules since the *IfcConstraintResource* schema enables to define benchmark values. However, it is not yet in scope to capture derivation rules in IFC so that this option was not further investigated.

Conclusion:

First option seems to be the best choice and thus is suggested for implementation. It does not require extensions to the IFC schema. However, implementation agreements are needed for identification of derivation rule references and the IFC objects that shall be linked with a derivation rule. In case a referenced document contains several rules there also need to be an agreement how to select the proper rule, i.e. the rule that fits to the document reference relationship.

[17] Although not explicitly forbidden, we assume that due to a similar meaning of *IfcRelAggregates* and *IfcRelContainedInSpatialStructure* only one of them should be used to define the element's hierarchy.

Cross-model links – proposed extensions	
Concept	**Possible solutions**
Link between grids (axis) and construction zones	There are several options that require further discussion: 1.) Reuse of grid axes (*IfcCurve*) for the geometric representation of construction zones (probably beside a solid representation, which would require an agreement about the representation context settings); 2.) Desribe the link via the local placement settings of the construction zone (*IfcGridPlacement*) 3.) *IfcRelReferencedInSpatialStructure* connection with *IfcGrid* (but without reference to grid axes) + agreement about description of grid axes names (e.g. via properties)

1.) Reuse of grid axes geometry

Dependencies between grid axes and constructions zones might be specified as an own geometric representation, which reuses the geometric representation of grid axes (*IfcCurve* or subtypes). In such case the grid axes geometry represents the borders of the construction zone in the plan view. Accordingly, the used set of grid axes must define a closed polygon[18]. To recognize the purpose of such geometric representation, the following implementation agreements are made:

- the representation identifier shall be “FootPrint”
- the representation context type shall be “Plan” (as suggested in the IFC documentation of *IfcGrid*)

The height of the construction zone shall be derived from the building storey in which the grid is contained. The relationship between the grid and the building storey is defined by *IfcRelContainedInSpatialStructure*. The height of the building storey is stored as an element quantity with the name *“NominalHeight”* (see IFC documentation of *IfcBuildingStorey*).

If the construction zone is defined by grid axes geometry it should also use the grid placement to avoid inconsistencies.

Figure 9 and Figure 10 demonstrate the reuse of grid axes geometry.

[18] The grid axes geometry will be trimmed by the intersection points of the used grid axes.

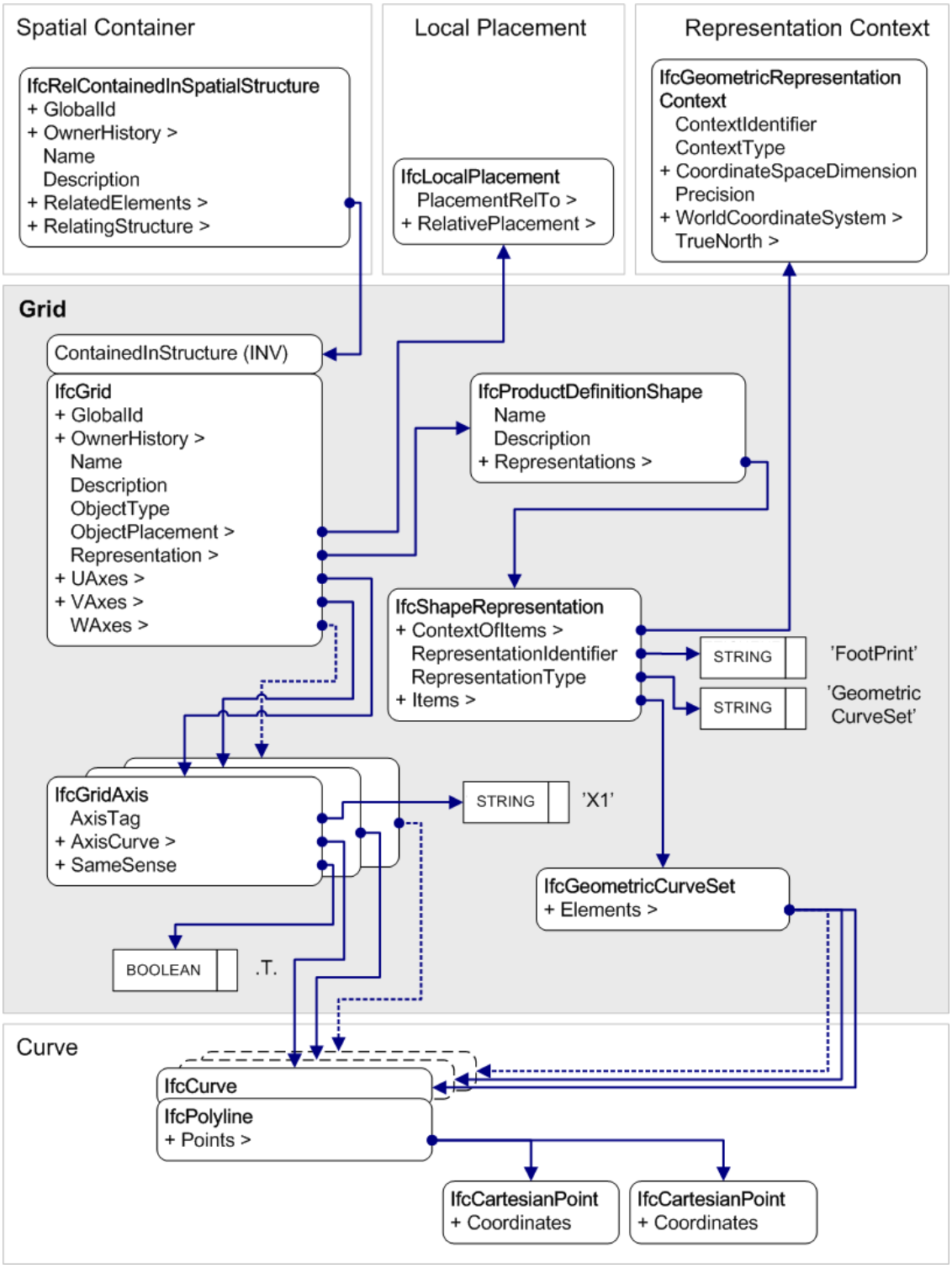


Figure 9: Instantiation diagram for the definition of grids and grid axes (option 1).

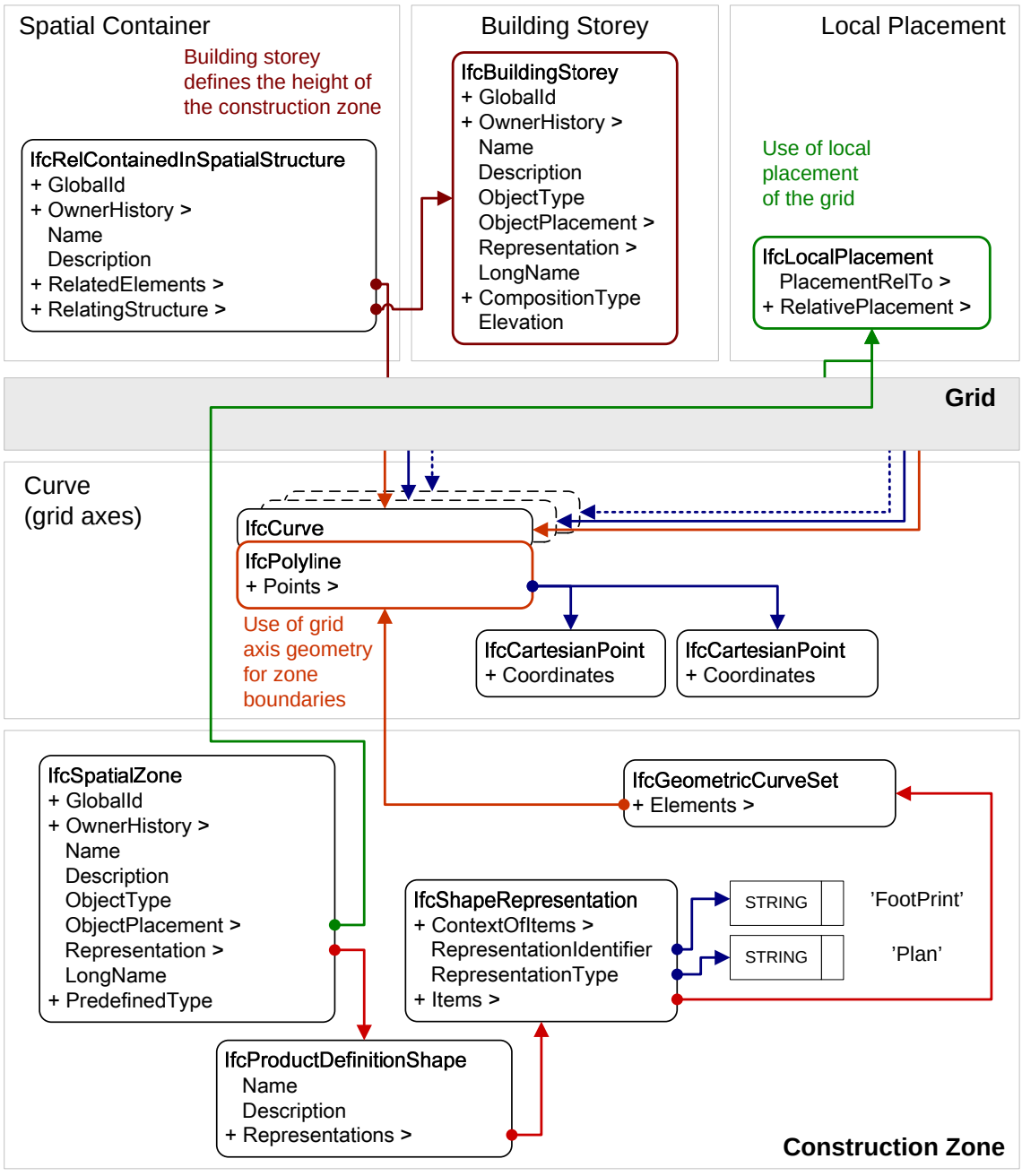


Figure 10: Instantiation diagram for the definition of dependencies between construction zones and grids (option 1).

2.) Grid placement for construction zones

Grids can be used to define the local placement of other elements such as construction zones. However, as the local placement is defined using the intersection of two grid axes it is not sufficient for the definition of a construction zone[19]. But it might be used to define the dependency between the construction zone and the grid, which otherwise has to be defined using *IfcRelReferencedInSpatialStructure* as shown in Figure 11.

[19] We need a set of grid axes that define the boundaries of the construction zone.

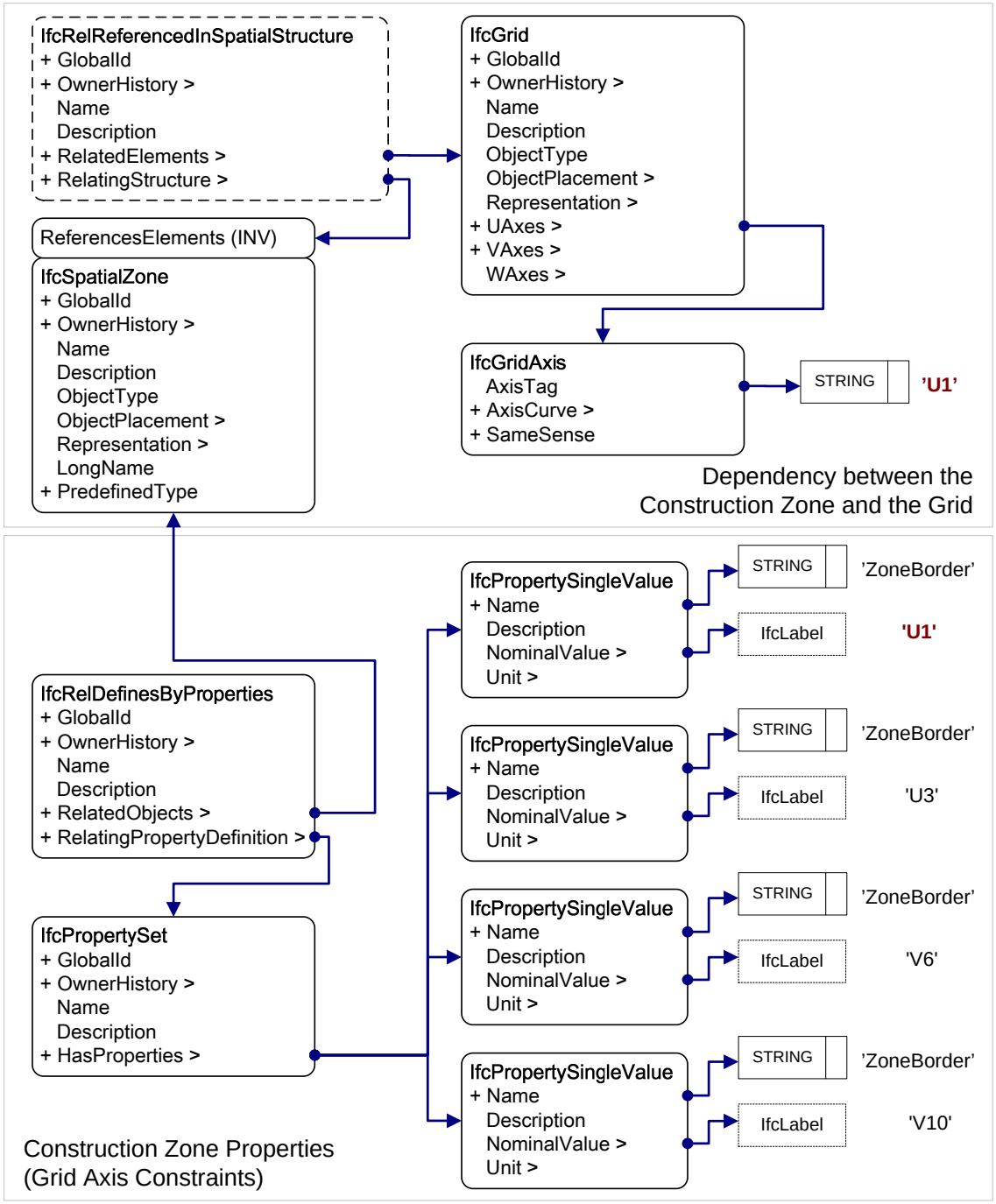


Figure 11: Instantiation diagram for the definition of construction zone properties (option 3)

3.) Spatial reference and property sets

The definition of construction zone boundaries can also be defined using properties that link to the grid axes by using the axis names (reference to the axis tag attribute). This way of defining grid axis constraints is only feasible if the axis name uniquely identifies the grid axis geometry. Thus, if the same axis name is used in different grids an additional link to the grid has to sort out wrong grid axes. Such link might be defined by the *IfcRelReferencedInSpatialStructure* relationship as shown in Figure 11, through relative placement as discussed in option 2) or other agreements.

Conclusion:

No final decision has been taken so far as the link between grid axes and spatial zones is not seen as mission critical information. Option 1) is used in our example (see chapter 4.3) to define the borders of the construction zone in the plan view. But independent of these options an explicit 3D model shape can be given for the construction zone that might be used for element splitting. Therefore, definition of grid constraints is not mandatory but might help to understand design decisions.

Cross-model links – proposed extensions	
Concept	**Possible solutions**
Link between work tasks and quantity/cost information	From the view point of model-based scheduling (where quantities are of main interest) it is suggested to use *IfcRelAssignsToProcess*, because the direction of the link (from one process to many cost/quantity items) is also used to set the link to building elements (and calculated element quantities) and is easier to follow from work tasks (defined by *IfcProcess.OperatesOn* instead of more generic *IfcObjectDefinition.HasAssignments*). Nevertheless, *IfcRelAssignsToControl* might be used to define a cost item (most likely a cost value instead of a quantity) that controls the work tasks and thus might be defined by the cost manager.

Cross-model links – proposed extensions	
Concept	**Possible solutions**
Link between building elements and quantity/cost information	It is suggested that only cost items/cost groups and no cost values are linked to building elements. The reason for that decision is that cost values shall be managed within the cost management domain and shall not be mixed with the architectural domain. Accordingly, if cost values must be associated to building elements a cost item is necessary in between, which defines the link to the cost value via *IfcRelAssociatesAppliedValue* and the link to the building element via *IfcRelAssignsToControl.*

3.3 Overview of relationships between main domain elements

Figure 12 gives an overview of agreements which were made for interlinking main element types of involved domains. It also shows the purpose of therefore used relationships in context of model-based scheduling. The following links are of main interest:

- **Work tasks – CAD elements**
  => enables 4D simulation, i.e. showing the planned progress of the construction according to the construction schedule
- **Work tasks – Cost items**
  => enables calculation of task duration, which is based on cost and quantity information but also involves experiences of the construction company
- **CAD elements – Cost items**
  => enables quantity take-off, i.e. use of CAD-based quantities as input for calculation of cost-related quantities

Furthermore, each element type can be decomposed using the general nesting feature of IFC. The nesting feature enables a hierarchical refinement so that requirements of each domain regarding a proper information structure can be fulfilled. Additionally, it enables to adjust the level of detail so that elements of different domains can be linked together to gain the functionality that is described above. Whereas the first aspect is dealt by the domains themselves and thus not discussed here, the proper level of detail for interlinking domain data is described in the following section.

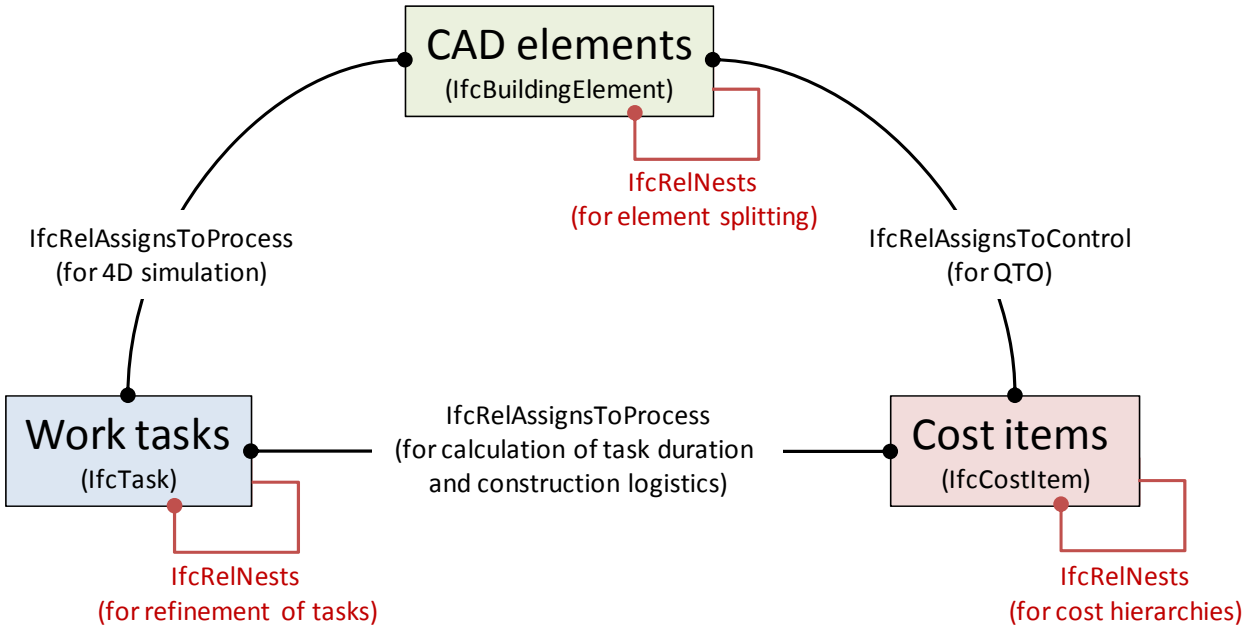


Figure 12: Relationships between main domain elements.

3.4 Assertion for full BIM functionality

To get the full functionality of model-based scheduling the following constraints can be defined, in particular for a proper level of detail between linked domain elements:

- **Constraint #1:** a cost item should belong to only one work task – otherwise no reliable work task duration can be calculated
- **Constraint #2:** for each CAD element that is used by a work task there should be at least one cost item, which furthermore is/are connected with the work task – if there is no such a cost item no reliable work task duration can be calculated
- **Constraint #3:** the set of cost items that are linked to work tasks through CAD elements should be equal to or a subset of the set of cost items that are directly linked to work tasks
- **Constraint #4:** a cost item should belong to only one CAD element – otherwise there is no clear dependency in case of modifications
- **Constraint #5:** each CAD element shall be linked to a work task – otherwise the construction schedule might be incomplete

It is not always possible to manage the three domains on a proper level of detail. Accordingly, a failed constraint does not necessarily mean that there is a logical error. Instead, violated constraints only indicate that there is a need for refinements to get full functionality of model-based scheduling as suggested in this document.

4 EXAMPLE

This section explains suggested mapping of identified exchange requirements to IFC on the basis of a small one storey example[20], which is focused on technical issues and may contain questionable decisions in terms of good construction processes. It follows the steps described by Tulke in Fijneman et al. (2008) and therefore starts with results from architectural design (see Figure 13). According to typical design processes the first step is to add the bill of quantities, which is based on and linked to results of the architectural design. Beside a QTO engineer the scenario also includes a scheduler who:

- defines the construction schedule,
- refines CAD elements and quantity items as needed for 4D simulation and calculation of task duration and finally
- establishes links between elements of the three domains.

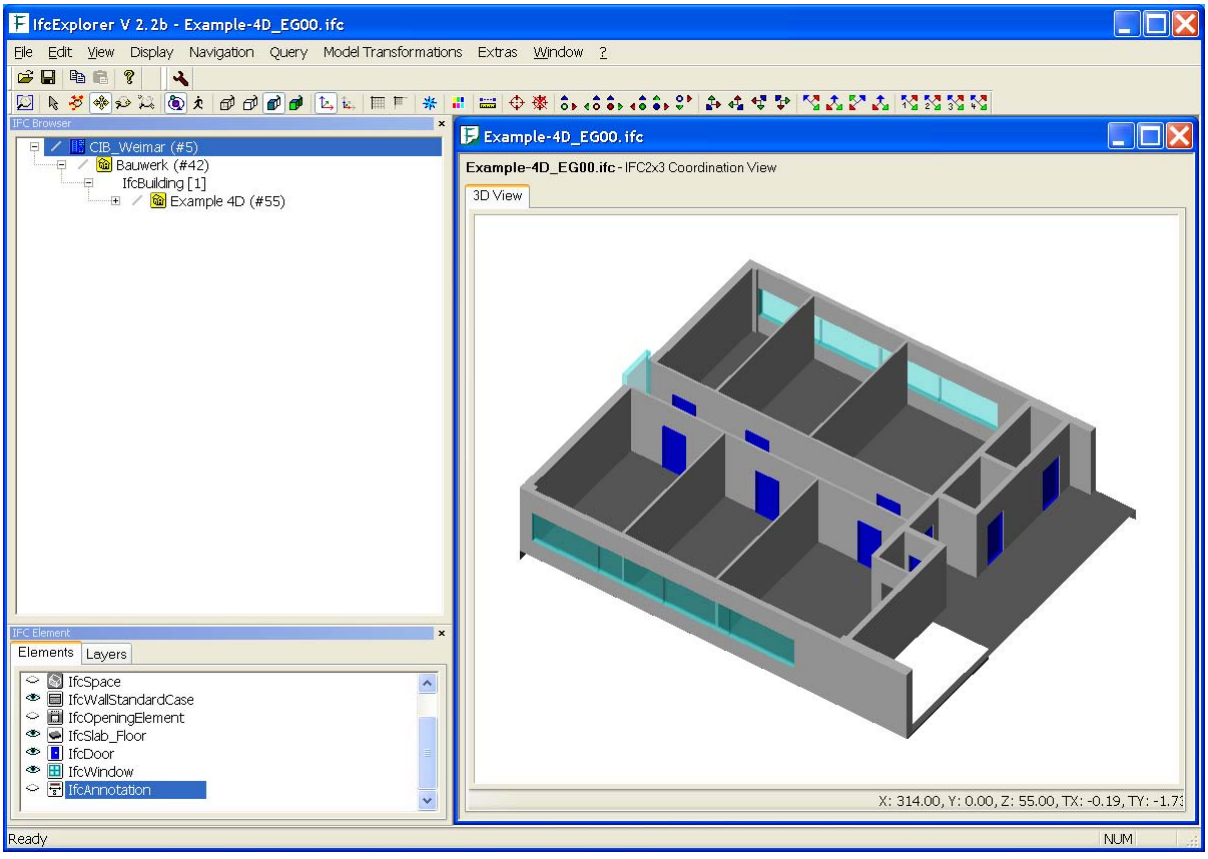


Figure 13: CAD view of the discussed example.

[20] Additional instantiation diagrams, in particular for definition of quantities, can be found in the appendix.

4.1 Model-based QTO and link to CAD elements

It is expected that the QTO engineer does not make changes to CAD elements, i.e. there is no refinement of architectural design data. Furthermore it is expected that base element quantities that can be derived from element geometry are available in the IFC domain model of the architect, i.e. they are provided by the CAD[21] application as an alphanumeric extensions to the CAD elements. Defining the bill of quantities is subdivided into three logical steps for further discussion.

Defining the bill of quantities structure

The bill of quantities follows a hierarchical structure according to relevant regulations and project dependent requirements. Accordingly, the structure in terms of hierarchy depth, number and types of items, etc. will differ between projects, but main concepts are used in the same way. There are basically two IFC entity types that are needed for describing bill of quantities:

- *IfcCostSchedule*
- *IfcCostItem*

Both entity types can define a hierarchical structure using *IfcRelNests*. Additionally, cost items can be assigned to cost schedules via *IfcRelSchedulesCostItems*.

There are several reasons for hierarchical refinement of cost schedules. However, for our example we assume to have only one cost schedule for describing the bill of quantities for our small example. Accordingly, there is only one instance of *IfcCostSchedule* which enables to set a number of general management data such as time stamps, involved actors etc. More details about such data can be found in the IFC documentation and the Functional Part *fp_model_cost_schedule*. As such kind of data is not in scope of this discussion the cost schedule example only defines mandatory attributes, including an *ID* ('BoQ_EG00') and the *PredefinedType* (.UNPRICEDBILLOFQUANTITIES.). The definition of an IFC cost schedule (instance with SPF-ID #50000) is shown in SPF-Snippet 1.

The IFC cost schedule defines the root element for the bill of quantities. All other elements are modeled as cost items[22]. For instance our example includes cost items for:

-construction work

- o slabs
- o exterior walls
  - concrete work for exterior walls (528,2 m³)
  - form work for straight exterior walls (4255,0 m²)
- o columns
  - concrete work for rectangular columns (16,4 m³)
  - form work for rectangular columns

[21]We also use the more precise abbreviation CAAD, which stands for Computer Aided Architectural Design. However, CAD and CAAD are used as synonyms here.

[22]Differentiation between cost item and cost schedule is not always clear. It might be also possible to model shown general elements as cost schedule.

- concrete work for round columns (50,8 m³)
- form work for round columns

o etc.

-finish work

-etc.

Main information of these elements are (1) an id or position number[23], (2) a name or short description and, if quantifiable, (3) an element quantity. The element hierarchy is modeled by instances of *IfcRelNests*. According to above given structure SPF-Snippet 1 shows the definition of cost items (#51000 to #51100) and the relationships defining the cost item hierarchy (#52000 ff.).

```
/* cost schedule */
#50000=IFCCOSTSCHEDULE('0J$yGtHBD12v72y4qF6XcD',#4,'Bill of quantity EG-00',
                       'InPro example',$,$,$,$,'PLANNED',$,$,'BoQ_EG00',
                       .UNPRICEDBILLOFQUANTITIES.);

/* relate the cost items to the cost schedule */
/* (includes only top level elements of the cost item hierarchy) */
#50001=IFCRELSCHEDULESCOSTITEMS('0J$yGtHBD12v72y4qF6R01',#4,$,$,
                                (#51000,#51100),$,#50000);

/* general cost items */
#51000=IFCCOSTITEM('0J$yGtHBD12v72y4qF6X01',#4,
                   '1','Construction work',$);
#51001=IFCCOSTITEM('0J$yGtHBD12v72y4qF6X02',#4,
                   '1.1.1','Exterior walls',$);
#51002=IFCCOSTITEM('0J$yGtHBD12v72y4qF6X03',#4,
                   '1.1.1.1','Concret work of exterior walls',$);
#51003=IFCCOSTITEM('0J$yGtHBD12v72y4qF6X04',#4,
                   '1.1.1.2','Form work for straight exterior walls',$);
#51004=IFCCOSTITEM('0J$yGtHBD12v72y4qF6X05',#4,
                   '1.4.2','Columns',$);
#51005=IFCCOSTITEM('0J$yGtHBD12v72y4qF6X06',#4,
                   '1.4.2.1.1','Concret work of columns (rectangular shape)',$);
#51006=IFCCOSTITEM('0J$yGtHBD12v72y4qF6X07',#4,
                   '1.4.2.1.2','Form work for columns (rectangular shape)',$);
#51007=IFCCOSTITEM('0J$yGtHBD12v72y4qF6X08',#4,
                   '1.4.2.2.1','Concret work of columns (round shape)',$);
#51008=IFCCOSTITEM('0J$yGtHBD12v72y4qF6X09',#4,
                   '1.4.2.2.2','Form work for columns (round shape)',$);
#51100=IFCCOSTITEM('0J$yGtHBD12v72y4qF6100',#4,
                   '2','Finish work',$);

/* nesting the cost schedules */
#52000=IFCRELNESTS('0J$yGtHBD12v72y4qF6R02',#4,$,$,#51000,
                   (#51001,#51004));
#52001=IFCRELNESTS('0J$yGtHBD12v72y4qF6R03',#4,$,$,#51001,
                   (#51002,#51003));
#52002=IFCRELNESTS('0J$yGtHBD12v72y4qF6R04',#4,$,$,#51004,
                   (#51005,#51006,#51007,#51008));
```

SPF-Snippet 1: Main structure for bill of quantities.

[23]Position number is typically derived from the element hierarchy.

If a quantity is available for a cost item it can defined and attached to the cost item as shown in SPF-Snippet 2. Each quantity is described by an instance of *IfcElementQuantity*, which contains a link to the cost value (*IfcQuantityVolume*, *IfcQuantityArea*) and enables to specify the method of measurement. There are no further agreements about proper *MethodOfMeasurement* settings. For shown main cost items it could be something like "*sum of child cost items"* or even an equation. Last not least the quantity elements are linked to cost items via *IfcRelDefinesByProperties*.

```
/* quantities of general cost items */
#53000=IFCELEMENTQUANTITY('0J$yGtHBD12v72y4qFQQ01',#4,$,$,
                         'Method of measurement can be described here',(#53001));
#53001=IFCQUANTITYVOLUME('QtoVolume',$,#28,528.242);
#53002=IFCRELDEFINESBYPROPERTIES('0J$yGtHBD12v72y4qF6R10',#4,$,$,(#51002),#53000);

#53010=IFCELEMENTQUANTITY('0J$yGtHBD12v72y4qFQQ02',#4,$,$,
                         'Method of measurement can be described here',(#53011));
#53011=IFCQUANTITYAREA('QtoArea',$,#27,4255.009);
#53012=IFCRELDEFINESBYPROPERTIES('0J$yGtHBD12v72y4qF6R11',#4,$,$,(#51003),#53010);

#53020=IFCELEMENTQUANTITY('0J$yGtHBD12v72y4qFQQ03',#4,$,$,
                         'Method of measurement can be described here',(#53021));
#53021=IFCQUANTITYVOLUME('QtoVolume',$,#28,16.406);
#53022=IFCRELDEFINESBYPROPERTIES('0J$yGtHBD12v72y4qF6R12',#4,$,$,(#51005),#53020);

#53030=IFCELEMENTQUANTITY('0J$yGtHBD12v72y4qFQQ04',#4,$,$,
                         'Method of measurement can be described here',(#53031));
#53031=IFCQUANTITYVOLUME('QtoVolume',$,#28,50.848);
#53032=IFCRELDEFINESBYPROPERTIES('0J$yGtHBD12v72y4qF6R13',#4,$,$,(#51007),#53030);
```

SPF-Snippet 2: Definition of quantities that are linked to main cost items.

Adding "physical" elements to the bill of quantities

Main cost items can be further subdivided into cost item elements, which contribute to the quantity of main cost items and can be linked to building elements. Ideally, the granularity of such cost item elements is on the same level as building elements, i.e. one cost item element can be linked to one building element[24].

```
/* cost item elements for exterior walls */
#51200=IFCCOSTITEM('0J$yGtHBD12v72y4qF6200',#4,'1.1.1.1.1','Wall #280',$);
#51201=IFCCOSTITEM('0J$yGtHBD12v72y4qF6201',#4,'1.1.1.1.2','Wall #504',$);
#51202=IFCCOSTITEM('0J$yGtHBD12v72y4qF6202',#4,'1.1.1.1.3','Wall #3006',$);
#51203=IFCCOSTITEM('0J$yGtHBD12v72y4qF6203',#4,'1.1.1.1.4','Wall #3318',$);
#51204=IFCCOSTITEM('0J$yGtHBD12v72y4qF6204',#4,'1.1.1.1.5','Wall #14842',$);
#51205=IFCCOSTITEM('0J$yGtHBD12v72y4qF6205',#4,'1.1.1.1.6','Wall #17511',$);
#51206=IFCCOSTITEM('0J$yGtHBD12v72y4qF6206',#4,'1.1.1.1.7','Wall #17735',$);
#51207=IFCCOSTITEM('0J$yGtHBD12v72y4qF6207',#4,'1.1.1.1.8','Wall #17934',$);

/* nesting the cost item elements */
#52010=IFCRELNESTS('0J$yGtHBD12v72y4qF6R05',#4,$,$,#51002,
                  (#51200,#51201,#51202,#51203,#51204,#51205,#51206,#51207));
```

SPF-Snippet 3: Definition of cost item elements that can be linked to building elements.

[24]Of course it is possible that there is no (geometrical) representation for cost item elements. In such case, there is no link to building elements. Please also note that one building element is often linked to more than one cost item element (concrete work, form work, etc.).

The definition of cost item elements is similar to main cost elements, i.e. instances of *IfcCostItem* are used and linked to main cost items via *IfcRelNests*. In the example there are eight exterior walls so that the main cost item *"concrete work for exterior walls"* is divided into eight cost item elements as shown in SPF-Snippet 3.

Setting links to CAD elements and adding quantities

Last step of the QTO engineer is to set the link to CAD elements and to calculate the element quantities. The link to CAD elements is defined by *IfcRelAssignsToControl* as shown in SPF-Snippet 4. Since there is an own cost item element for each exterior wall the example only shows 1:1 relationships. However, it is also possible to assign one cost item element to several CAD elements. Such 1:n relationship can be defined by one instance of *IfcRelAssignsToControl*[25].

```
/* quantities of cost item elements */
#53200=IFCELEMENTQUANTITY('0J$yGtHBD12v72y4qFQQ10',#4,'QTO',$,
                         '3.10*2.99*0.42',(#53201));
#53201=IFCQUANTITYVOLUME('QtoVolume',$,#28,3.893);
#53202=IFCRELDEFINESBYPROPERTIES('0J$yGtHBD12v72y4qFRR10',#4,$,$,(#51200),#53200);

#53210=IFCELEMENTQUANTITY('0J$yGtHBD12v72y4qFQQ11',#4,'QTO',$,
                         '((3.10*13.20)-(1.40*11.35))*0.21',(#53211));
#53211=IFCQUANTITYVOLUME('QtoVolume',$,#28,5.256);
#53212=IFCRELDEFINESBYPROPERTIES('0J$yGtHBD12v72y4qFRR11',#4,$,$,(#51201),#53210);

…

/* link between cost item elements and exterior walls */
#54000=IFCRELASSIGNSTOCONTROL('0J$yGtHBD12v72y4qFRR50',#4,$,$,(#280),
                              .PRODUCT.,#51200);
#54001=IFCRELASSIGNSTOCONTROL('0J$yGtHBD12v72y4qFRR51',#4,$,$,(#504),
                              .PRODUCT.,#51201);
…

/* wall elements that are linked with cost item elements of the QTO domain */
#280= IFCWALLSTANDARDCASE('0Qg2PoECH4hR2S0TM273Pm',#4,$,$,$,#112,#115,$);
#504= IFCWALLSTANDARDCASE('2ES3Pc9Fz60AIFGhvRIr63',#4,$,$,$,#387,#390,$);
…
```

SPF-Snippet 4: Quantities of cost item elements and link to building elements.

Quantities are defined by *IfcElementQuantity* as shown in SPF-Snippet 2. Furthermore, the *MethodOfMeasurement* attribute might be used to describe the calculation rule. The example shown in SPF-Snippet 4 describes the calculation approach together with values that were derived from wall geometry. The style that is used in the example such as '3.10*2.99*0.42' and '((3.10*13.20)-(1.40*11.35))*0.21' helps the QTO engineer to verify element quantities, e.g. to support double checks or to react on design changes. If the method of measurement shall be used for implementation it is recommended to coordinate further implementation agreements with the German IAI group "Model-based quantities" that has developed a proposal for definition of calculation rules.

[25]The link to CAD elements (*IfcRelAssignsToControl.RelatedObjects*) is defined as a set of elements.

Summary

Shown use of IFC for QTO purposes only covers the information that is needed for model-based scheduling. The example describes how to define (1) the bill of quantities structure, (2) quantities of QTO elements and (3) the link between QTO elements and CAD elements. It furthermore gives recommendations about a proper level of detail of QTO elements so that they can be linked to CAD elements. But it should also be noticed that there are many more aspects for QTO and related cost management that have not been discussed in our example as they go beyond the scope of our use case scenario.

4.2 Initial set-up of construction schedule

Architectural design and results from QTO as discussed in the previous section provide the basis for model-based scheduling. Similar to architectural design and QTO there are domain specific elements that define an own hierarchical structure and have to be linked to elements from other domains. Main difficulty is to adjust the level of detail between these elements so that they support the aim of suggested links.

Before discussing adjustment of element granularity the domain elements of the scheduler are briefly introduced in context of our example. It shows IFC entity types that describe the construction schedule and are later on used to specify the links.

Defining the construction schedule

Work tasks are the main elements of a construction schedule. They basically define what has to be done, when something is planned to be started and finished and what dependencies exist within the construction process. The relationship between a construction schedule and work tasks is similar to a cost schedule and its cost items. It is defined by an instance of *IfcRelAssignsTasks*, which also enables to set a link to time constraints of work tasks (*IfcScheduleTimeControl*). Hierarchical structures are defined by instances of *IfcRelNests* and work task dependencies by instances of *IfcRelSequence*. The use of these IFC entity types is shown in SPF-Snippet 5.

The small example as shown in SPF-Snippet 5 contains the following elements:

- one work schedule (#44)
- three work tasks, whereas the task #47 is the parent of task #52 and task #58
- time constraints for each work task – scheduled start and scheduled finish (please notice that there constraints due to the task hierarchy)
- task sequence relationship (#70) defining the logical dependency (type FINISH_START) between task #58 and task #52

More details about use of these types can be found in the mapping document of Serén & Karstila (2001).

```
/* define a work schedule */
#44 = IFCWORKSCHEDULE('39$2dkq1vAKvhT8yh1KTlc', #2, $, '4D work schedule', $,
                      '1', #45, $, '4D modelling', $, $, #46, $, $, $);

/* define a work tasks */
#47 = IFCTASK('1tObBzN7TqKuAzVjd$vJFo', #2, 'Wall construction', $, $,
              '8', 'NOTDEFINED', $, .F., $);
#52 = IFCTASK('1EeYZSxlSbHPwVDY0iDWUk', #2, 'Form work', $, $, '9',
              'NOTDEFINED', $, .F., $);
#58 = IFCTASK('2Xalkzx7HFHCjlHeWyMTRo', #2, 'Concrete work', $, $,'10',
              'NOTDEFINED', $, .F., $);

/* define a work task hierarchy */
#53 = IFCRELNESTS('2jLQ0rkw3NJAuhOvX4cFoT', #2, $, $, #47, (#52, #58));

/* assign work tasks to a work schedule and set time constraints */
#51 = IFCRELASSIGNSTASKS('2QdZFOE0RiHSokeGyp5E$C',#2, $, $,(#47),.PROCESS.,#44,#48);
#57 = IFCRELASSIGNSTASKS('1CQTJwMO06HQzBkF9brDxA',#2, $, $,(#52),.PROCESS.,#44,#54);
#62 = IFCRELASSIGNSTASKS('1OUwodf3a3HCmX5LeQ$KZ5',#2, $, $,(#58),.PROCESS.,#44,#59);

/* define time constraints */
#45 = IFCCALENDARDATE(13, 10, 2008);
#46 = IFCCALENDARDATE(01, 12, 2008);
#48 = IFCSCHEDULETIMECONTROL('2ydIbGeGJLHwR5IzvF2zH2', #2, $, $, $, $, $, $, #49,
          $, $, $, #50, $, $, $, $, $, $, $, $, $, $);
#49 = IFCCALENDARDATE(10, 12, 2008);
#50 = IFCCALENDARDATE(22, 12, 2008);
#54 = IFCSCHEDULETIMECONTROL('3R5wblicCcHw51Kk1TvnwO', #2, $, $, $, $, $, $, #55,
          $, $, $, #56, $, $, $, $, $, $, $, $, $, $);
#55 = IFCCALENDARDATE(10, 12, 2008);
#56 = IFCCALENDARDATE(11, 12, 2008);
#59 = IFCSCHEDULETIMECONTROL('0GbkyzS4t$IRd9sRXO0L6a', #2, $, $, $, $, $, $, #60,
          $, $, $, #61, $, $, $, $, $, $, $, $, $, $);
#60 = IFCCALENDARDATE(12, 12, 2008);
#61 = IFCCALENDARDATE(22, 12, 2008);

/* task sequence */
#70=IFCRELSEQUENCE('2$_3qPuwjeJRP2OdEXihZk',#2,$,$,#52, #58, .0,.FINISH_START.);
```

SPF-Snippet 5: Use of IFC entity types for definition of work schedule, work tasks, time constraints, work task hierarchy and logical dependency of work tasks.

4.3 Refinement of CAD elements based on linking rules

Links between work tasks and CAD elements as well as between work tasks and cost items are essential for 4D simulation and for estimation of task durations. Since these elements are often not on the same level of granularity (work tasks are typically more detailed than CAD and QTO) they have to be adjusted to be in line with scheduling data. Consequently, the scheduler needs to refine CAD elements and cost items to be able to start the linking process. Therefore we first describe how to refine CAD elements and cost items, and then go on with the definition of element links.

Defining construction zones and setting links to grid axes

Construction zones shall provide the basis for splitting rules of CAD elements. They might be derived from a grid, which is usually used for identification of building sections. The link from construction zones to grid axes is not needed for element splitting but helps to explain the planned construction process to architects, con-

struction workers, clients etc. Therefore, we start with representation of grids that might be provided by the architect or created by the scheduler.

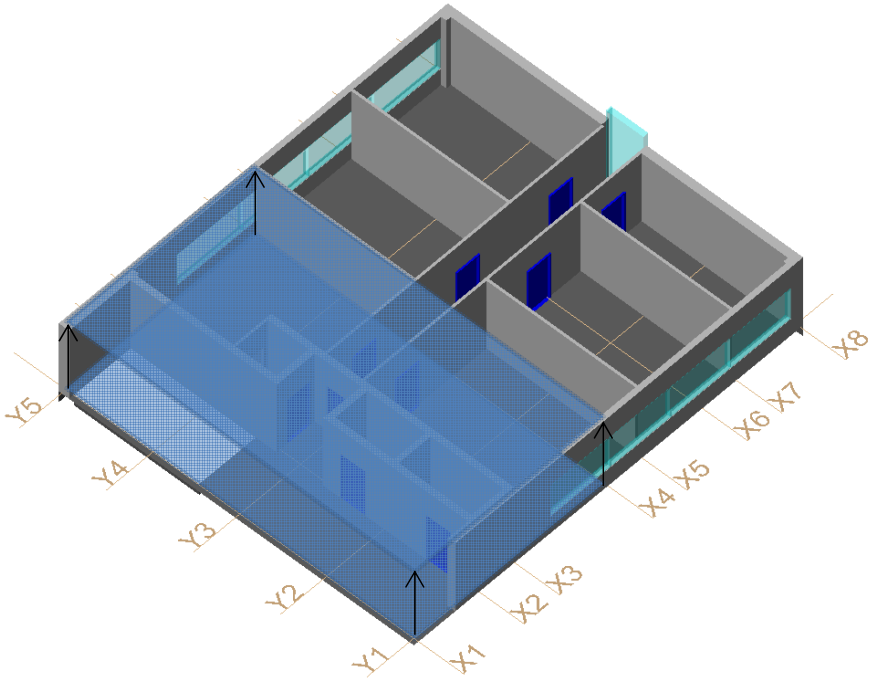


Figure 14: Grid, grid axes and derived construction zone of the example.

The definition of grids is shown in SPF-Snippet 6. Main IFC elements are *IfcGrid* (#30010), *IfcGridAxes* (#30089, #30096, ...) and *IfcRelContainedInSpatialStructure* (#30500).

The shown instance of *IfcGrid* contains the information that is needed for definition of grid geometry (local placement and shape) and assignment of grid axes (two sets for U and V axes). Further information such as name, description, type settings etc. can be assigned if necessary, but are not set for explaining the link to construction zones.

Instances of *IfcGridAxes* are defined by a unique[26] axis tag such as “X1”, “X2” etc., a polyline for geometric definition relative to the local placement of the grid and a flag for adjusting the sense of the polyline. Please note that there are two geometric definitions, one for the grid and one for each grid axis. However, they use the same polylines for definition of grid axes (see geometric curve set #30032) so that there is no mismatch between both definitions (see also IFC documentation).

Last not least there is a containment relationship between the grid and the building storey (#30500), which defines geometrical constraints of construction zones (height and location). Please note that the grid and the building storey in which the grid is contained use the same local placement (#83).

The grid and its grid axis as defined in SPF-Snippet 6 provide the basis for the definition of a construction zone, which is an instance of *IfcSpatialZone* from the new IFC 2x4 release. According to Figure 10 (option 1 for definition of dependencies between a grid and a construction zone), SPF-Snippet 7 shows its implementation for our example. It defines the plan view of the construction zone via the polylines (#30122, #30143,

[26] Axis tags shall be unique within the same grid.

#30087, #30115) of the grid axes. But the shown example does not include a 3D representation using an extruded area solid and a link to the building storey object via an instance of *IfcRelContainedInSpatialStructure.*

```
/*grid definition*/
#30010= IFCGRID('1tTSgyyDb2j8r9EiHHGhi$',#4,$,$,$,#30020,#30030,
               (#30089,#30096,#30103,#30110,#30117),
               (#30124,#30131,#30138,#30145,#30152,#30238,#30259,#30159),$);
#30020= IFCLOCALPLACEMENT(#83,#30021);
#30030= IFCPRODUCTDEFINITIONSHAPE($,$,(#30031));
#30031= IFCSHAPEREPRESENTATION(#11,'FootPrint','GeometricCurveSet',(#30032));
#30032= IFCGEOMETRICCURVESET((#30087,#30094,#30101,#30108,#30115,#30122,
                        #30129,#30136,#30143,#30150,#30157,#30236,#30257));
#30021= IFCAXIS2PLACEMENT3D(#30520,#30510,#30511);

/*coordinates for local placement of grid*/
#30510= IFCDIRECTION((0.,0.,1.));
#30511= IFCDIRECTION((0.,1.,0.));
#30512= IFCDIRECTION((1.,0.,0.));
#30520= IFCCARTESIANPOINT((7550.,0.,0.));

/*grid axes*/
#30083= IFCCARTESIANPOINT((0.,0.));
#30085= IFCCARTESIANPOINT((20000.,0.));
#30087= IFCPOLYLINE((#30083,#30085));
#30089= IFCGRIDAXIS('Y1',#30087,.T.);
#30090= IFCCARTESIANPOINT((0.,3480.));
#30092= IFCCARTESIANPOINT((20000.,3480.));
#30094= IFCPOLYLINE((#30090,#30092));
#30096= IFCGRIDAXIS('Y2',#30094,.T.);
..
#30118= IFCCARTESIANPOINT((2000.,-2000.));
#30120= IFCCARTESIANPOINT((2000,16000.));
#30122= IFCPOLYLINE((#30118,#30120));
#30124= IFCGRIDAXIS('X1',#30122,.T.);
#30125= IFCCARTESIANPOINT((4600.,-2000.));
#30127= IFCCARTESIANPOINT((4600.,16000.));
#30129= IFCPOLYLINE((#30125,#30127));
#30131= IFCGRIDAXIS('X2',#30129,.T.);
..
/*containment in spatial structure*/
#30500= IFCRELCONTAINEDINSPATIALSTRUCTURE('3_yYAG3h1CS9hMALjK7MGR',
                        #4,$,$,(#30010),#86);
#86= IFCBUILDINGSTOREY('2Kik5LzlzCThA$4hXNvVu$',#4,'EG00',$,$,#83,
                        $,$,.ELEMENT.,11899.99);
#83= IFCLOCALPLACEMENT(#65,#68);
```

SPF-Snippet 6: Definition of grid, grid axes and link to the building storey.

```
/*construction zone (IFC2x4)*/
#31000= IFCSPATIALZONE('3_yYAG3h1CS9hMALjK7MCZ',#4,'Construction zone 1',
              $,'ConstructionZone',#31020,#31030,$,.USERDEFINED.);
#31020= IFCLOCALPLACEMENT(#83,#31021);
#31030= IFCPRODUCTDEFINITIONSHAPE($,$,(#31031));
#31031= IFCSHAPEREPRESENTATION(#11,'FootPrint','Plan',(#31032));
#31032= IFCGEOMETRICCURVESET((#30122,#30143,#30087,#30115));
#31021= IFCAXIS2PLACEMENT3D(#31520,#31510,#31511);

/*coordinates for local placement of construction zone*/
#31510= IFCDIRECTION((0.,0.,1.));
#31511= IFCDIRECTION((0.,1.,0.));
#31520= IFCCARTESIANPOINT((7550.,0.,0.));
```

SPF-Snippet 7: Definition of construction zone in new IFC release 2x4.

Refinement of CAD elements and setting references to external rules

The principle of element refinement (e.g. splitting of a wall element according to a construction zone) and the link to externally defined splitting rule documents is shown in SPF-Snippet 8.

The wall #504 (not shown) is split into two child elements (wall #100504 and wall #100604) using the nesting functionality of IFC (#101000). But contrary to the implementation agreement #CV-2x3-119 of the Coordination View both child elements define their own geometry in order to be usable for 4D simulation. This is necessary due to the decision to do not change architectural design data, which means that it is not possible to remove the geometry definition from the parent wall #504 and therefore violates the implementation agreement #CV-2x3-119. Furthermore we suggest that child elements should reuse the local placement of the parent element, which is shown in the example by sharing the local placement #387. However, sharing of *IfcLocalPlacement* instances is possible since IFC2x4 so that IFC2x3 implementations need to create a copy of #387 for each child element.

```
/*splitting of building elements */
#100504=IFCWALLSTANDARDCASE('2ES3Pc9Fz60AIFGhvRISP1',#4,
               'Part of wall #504',$,$,#387,#100390,$);
#100390=IFCPRODUCTDEFINITIONSHAPE($,$,(#394,#100412));
...
#100604=IFCWALLSTANDARDCASE('2ES3Pc9Fz60AIFGhvRISP2',#4,
               'Part of wall #504',$,$,#387,#100490,$);
#100490=IFCPRODUCTDEFINITIONSHAPE($,$,(#394,#100512));
...
#101000=IFCRELNESTS('0J$yGtHBD12v72y4qFNR01',#4,
               'Splitting of wall #504',$,#504,(#100504,#100604));

/*reference to external splitting rule
  (link to child and parent walls + construction zone) */
#101100=IFCRELASSOCIATESDOCUMENT('0J$yGtHBD12v72y4qFNR02',#4,
               'Link to splitting rule of wall #504',$,
               (#504,#100504,#100604,#31000),#101200));
#101200=IFCDOCUMENTREFERENCE(
        'http://inpro-project.eu/splitting_rules/example_project.rules',
        'GUID=3_yYAG3h1CS9hMALjK7MCZ',
        'Splitting rules for construction zone 1');
```

SPF-Snippet 8: Splitting of wall #504 and link to external splitting rules

Further adjustments are necessary for the element quantities that have to be calculated on the basis of the child element geometry. We also suggest to define a “containment” relationship between the child elements and the construction zone via *IfcRelReferencedInSpatialStructure* to show its dependency within the IFC model. Such relationship might be redundant when defining a link to external derivation rules as defined by the document reference #101200, which is associated via *IfcRelAssociatesDocument* (#101100) with the parent wall (#504), the two child walls (#100504 and #100604) and the construction zone (#31000).

Refinement of cost items (adjustment to CAD elements)

Cost items as defined by the QTO engineer not necessarily fit to the element granularity that is needed to describe construction schedules and thus to link cost items with work tasks and building elements as shown in Figure 12 and explained in chapter 3.4. An appropriate level of detail can be reached by subdivision of cost items according to the building element refinement. The principle is quite similar and also uses the nesting feature of IFC.

Splitting of the cost item #51201 into #151201 and #151202 via *IfcRelNests* (#152010) is shown in SPF-Snippet 9. These two new cost items are both linked through a 1:1 relationship (*IfcRelAssignsToControl*) to a wall object that enables to derive cost item quantities. Ideally, such (1:1) link also exists between both parent objects, i.e. the parent wall #504 (see SPF-Snippet 8) and the parent cost item #51201. This might help to automatically derive cost item quantities based on the ratio of the element quantities. However, as refinement is done by a scheduler and not by a QTO engineer the cost item quantities, if important for controlling the construction process and the supply chain, should be controlled or provided by the QTO engineer. Accordingly, this might require additional data exchange and coordination activities in our use case scenario.

```
/* splitting of cost items according to the construction zones */
#151201=IFCCOSTITEM('0J$yGtHBD12v72y4qF6211',#4,'1.1.1.1.2.1',
                'Wall #100504',$);
#151202=IFCCOSTITEM('0J$yGtHBD12v72y4qF6212',#4,'1.1.1.1.2.2',
                'Wall #100604',$);

/* nesting the cost items */
#152010=IFCRELNESTS('0J$yGtHBD12v72y4qF7R01',#4,$,$,#51201,(
                #151201,#151202));

/* quantities of cost item elements */
#153210=IFCELEMENTQUANTITY('0J$yGtHBD12v72y4qFWW11',#4,'QTO',$,
                '((3.10*5.50)-(1.40*3.50))*0.21',(#153211));
#153211=IFCQUANTITYVOLUME('QtoVolume',$,#28,2.552);
#153212=IFCRELDEFINESBYPROPERTIES('0J$yGtHBD12v72y4qFRR88',#4,$,$,
                (#151201),#153210);

#153220=IFCELEMENTQUANTITY('0J$yGtHBD12v72y4qFWW12',#4,'QTO',$,
                '((3.10*7.70)-(1.40*7.85))*0.21',(#153221));
#153221=IFCQUANTITYVOLUME('QtoVolume',$,#28,2.705);
#153222=IFCRELDEFINESBYPROPERTIES('0J$yGtHBD12v72y4qFRR89',#4,$,$,
                (#151202),#153220);

/* link between splitted cost item elements and splitted exterior walls */
#154000=IFCRELASSIGNSTOCONTROL('0J$yGtHBD12v72y4qFRR60',#4,$,$,
                (#100504),.PRODUCT.,#151201);
#154001=IFCRELASSIGNSTOCONTROL('0J$yGtHBD12v72y4qFRR61',#4,$,$,
                (#100604),.PRODUCT.,#151202);
```

SPF-Snippet 9: Refinement of cost items and

Linking work tasks with building elements and cost items

After refinement of building elements and cost items the links to work tasks (construction processes) can be established using instances of *IfcRelAssignsToProcess* (see also Figure 12). The work task (#160000) is linked via the assignment relationship #160200 to the wall part (#100504). The wall part (*IfcWallStandard-Case*) is a subtype of *IfcProduct* so that the attribute *IfcRelAssignsToProcess.RelatedObjectsType* is set to *PRODUCT*. This is different to the link to the cost item #151201 (see SPF-Snippet 9), which is a subtype of *IfcControl* and thus is a *CONTROL* type.

We generally recommend establishing 1:1 relationships between tasks and building elements/cost items (i.e. they should be on the same level of detail) so that we do not need the attribute *IfcRelAssignsToProcess.QuantityInProcess,* which is defined to be the *"Quantity of the object specific for the operation by this process"*. Instead, as each building element/cost item is linked with only one work task we can use the quantities that are defined via *IfcRelDefinesByProperties* as described in SPF-Snippet 9.

```
/*work task*/
#160000=IFCTASK('1EeYZSxlSbHPwVDY0Task1', #4,
                'Concret work of exterior wall in construction zone 1',
                $, $, '24','NOTDEFINED', $, .F., $);

/*link between work tasks and CAD elements*/
#160200=IFCRELASSIGNSTOPROCESS('0J$yGtHBD12v72y4qFRR90',#4,$,$,
                (#100504),.PRODUCT.,#160000, $);

/*link between work tasks and cost item elements*/
#160300=IFCRELASSIGNSTOPROCESS('0J$yGtHBD12v72y4qFRR91',#4,$,$,
                (#151201),.CONTROL.,#160000, $);
```

SPF-Snippet 10: Linking work tasks with building elements and cost items via *IfcRelAssignsToProcess*

4.4 Adding 4D simulation parameter + tool support parameter

After having established the construction schedule and the links between the three domains 4D visualization parameters can be added to deal with visualization issues such as color and transparency. Required parameters are added as properties using the dynamic extension functionality of IFC (*IfcPropertySet*) so that it basically means to agree upon used parameter names (key) and the data types (range of values). Parameter names and data types are discussed in Figure 6 and Figure 7 and, assuming to be familiar with the property concept of IFC, should be clear. SPF-Snippet 11 additionally shows 4D-visualization and tool support parameters in the SPF format.

```
/*4D visualization properties & tool support parameter */
#160000=IFCTASK('1EeYZSxlSbHPwVDY0Task1', #4,
               'Concret work of exterior wall in construction zone 1',
               $, $, '24','NOTDEFINED', $, .F., $);
#160001=IFCRELDEFINESBYPROPERTIES('2K5FvWlVz8BewqgTQUfMYL',#4,$,$,
               (#160000),#160002);
#160002=IFCPROPERTYSET('09OO0kRDDDnuATp7E65iLQ',#4,'4DParameter',$,
               (#160003,#160004,#160005,#160006,#160007,#160009));
#160003=IFCPROPERTYSINGLEVALUE('isVisible',$,IFCBOOLEAN(.T.),$);
#160004=IFCPROPERTYSINGLEVALUE('RGB_Red',$,IFCINTEGER(0),$);
#160005=IFCPROPERTYSINGLEVALUE('RGB_Green',$,IFCINTEGER(0),$);
#160006=IFCPROPERTYSINGLEVALUE('RGB_Blue',$,IFCINTEGER(255),$);
#160007=IFCPROPERTYSINGLEVALUE('Transparency',$,IFCINTEGER(0),$);
#160009=IFCPROPERTYENUMERATEDVALUE('ConstructionType',$,
               (IFCLABEL('CONSTRUCT')),#160010);
#160010=IFCPROPERTYENUMERATION('ConstructionTypeEnumeration',
               (IFCLABEL('CONSTRUCT'),IFCLABEL('TEMPORARY'),
                IFCLABEL('DEMOLISH'),IFCLABEL('START_PERIOD'),
                IFCLABEL('MIDDLE_PERIOD'),IFCLABEL('END_PERIOD'),
                IFCLABEL('PERIOD')),$);

#160101=IFCRELDEFINESBYPROPERTIES('2K5FvWlVz8BewqgTQUfMAS',#4,$,$,(
               #160000),#160102);
#160102=IFCPROPERTYSET('09OO0kRDDDnuATp7E65iLR',#4,,MSProjectSupport',
               $,(#160103,#160104));
#160103=IFCPROPERTYSINGLEVALUE('uniqueNumber',$,IFCINTEGER(321),$);
#160104=IFCPROPERTYSINGLEVALUE('PSP_code',$,IFCLABEL('1.2.4'),$);
```

SPF-Snippet 11: Adding property sets for definition of 4D visualization and tool support parameters

5 REFERENCES

Fijneman M., Matthyssen A., Verhofstad F. & Tulke J. (2008): *D49B – Process Mapping for Key Process 4 – Construction Schedule.* Intermediate Deliverable (restricted report) of the NMP-EU project InPro (IP 026716-2), 8 August 2008.

Firmenich B. (2004): *A Novel Modelling Approach for the Exchange of CAD Information in Civil Engineering.* In: Proceedings of the 5th European Conference on Product and Process Modelling in the Building and related Industries. A. A. Balkema Publishers, September 2004.

Froese T. & Yu K. (1999): *Industry Foundation Class Modeling for Estimating and Scheduling.* Published in CIB W78 1999, Ottawa, Canada.

Heesom D. & Mahdjoubi L. (2004): *Trends of 4D CAD applications for construction planning.* Construction Management and Economics, Volume 22, Number 2, February 2004.

Hietanen J. (2006): *IFC Model View Definition Format.* International Alliance for Interoperability, April 2006, available at: http://www.iai-international.org/software/ MVD_060424/IAI_IFCModelViewDefinitionFormat.pdf

Karstila K. & Serén K. (2005): *IFC Aspect Card Library.* Result of the Pro-IT project, Eurostep.

Kähkönen K. & Leinonen J. (2003): *Visual product Chronology as a Solution for Accessing Building Product Model Data.* In: Proceedings of the CIB W78 conference, Auckland, New Zealand.

Koch C., Firmenich B. (2006): *Operative models for the introduction of additional semantics into the cooperative planning process.* In: Proceedings of the 13th EG-ICE Workshop "Intelligent Computing in Engineering and Architecture". Ascona, Schweiz, June 2006.

Kiviniemi A. Tarandi V. Karlshøj R. Bell H. Karud O.J. (2008): *Review of the Development and Implementation of IFC compatible BIM.* Erabuild report.

Liebich T. ed. (2004): *IFC2x Edition 2 – Model Implementation Guide.* Version 1.7, March 18, 2004, available at: www.iai-international.org.

Porkka & Kähkönen (2007): *Software Development Approaches and Challenges of 4D Product Models.* Paper of the CIB-W78 conference 2007,
available at: http://itc.scix.net/data/works/att/w78-2007-013-096-Porkka.pdf

Serén K. & Karstila K. (2001): *MS Project – IFC Mapping Specification.* Report from VTT and Eurostep for IFC1.5.1, IFC2.0 and IFC2x.

Houttu J.-M. ed. (2009): *D15 – Early Design Applications.* Restricted report of the NMP-EU project InPro (IP 026716-2), to be published in 2009.

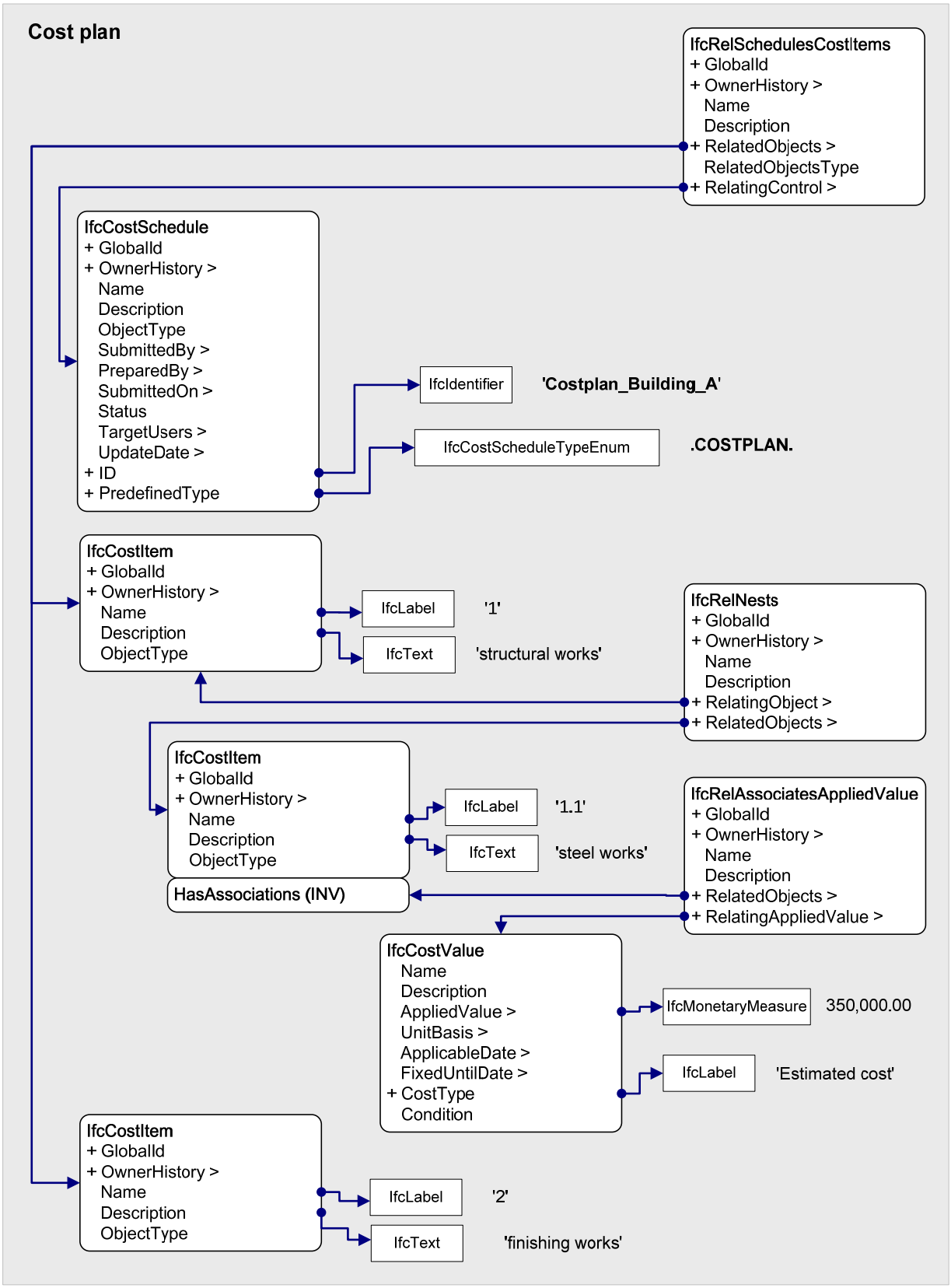
Tulke J. Nour M. and Beuke K. (2008): *Decomposition of BIM objects for scheduling and 4D simulation.* In: Proceedings of the 7th European Conference on Product and Process Modelling in the Building and related Industries. A. A. Balkema Publishers, September 2008.

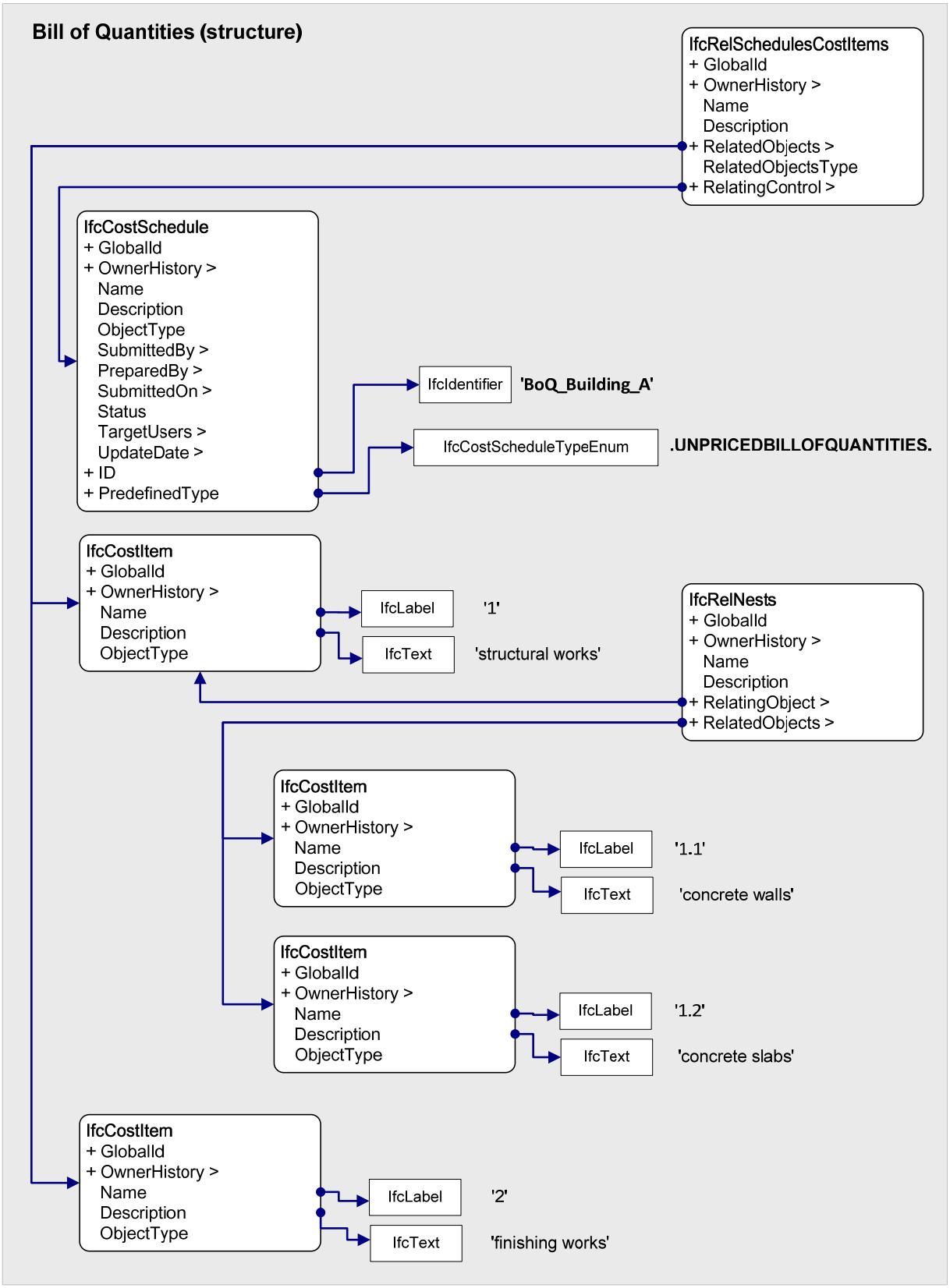
Wix J. ed (2006): *Information Delivery Manual: Guide to Components and Development Methods.* Available at: http://idm.buildingsmart.no

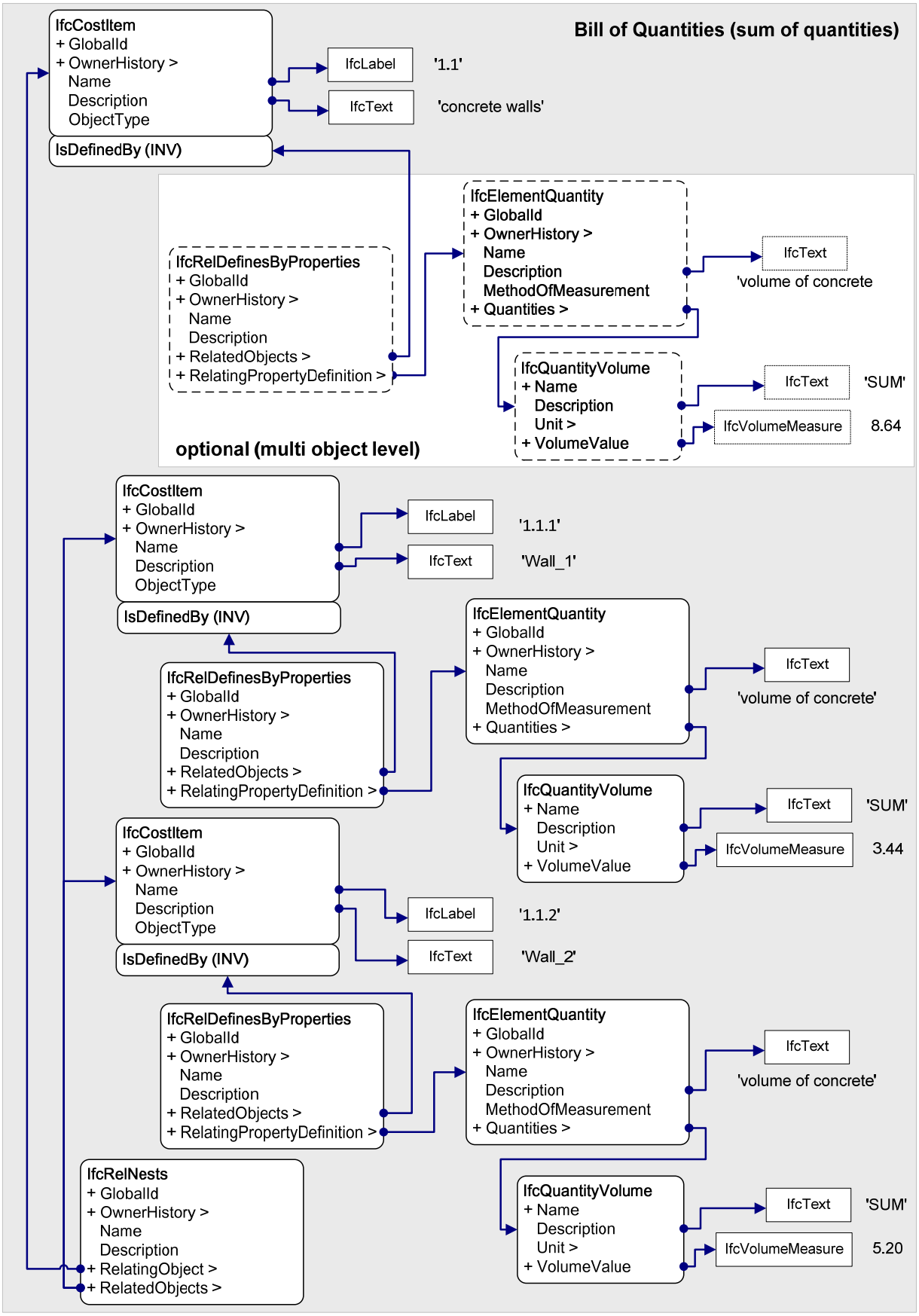
Web links:

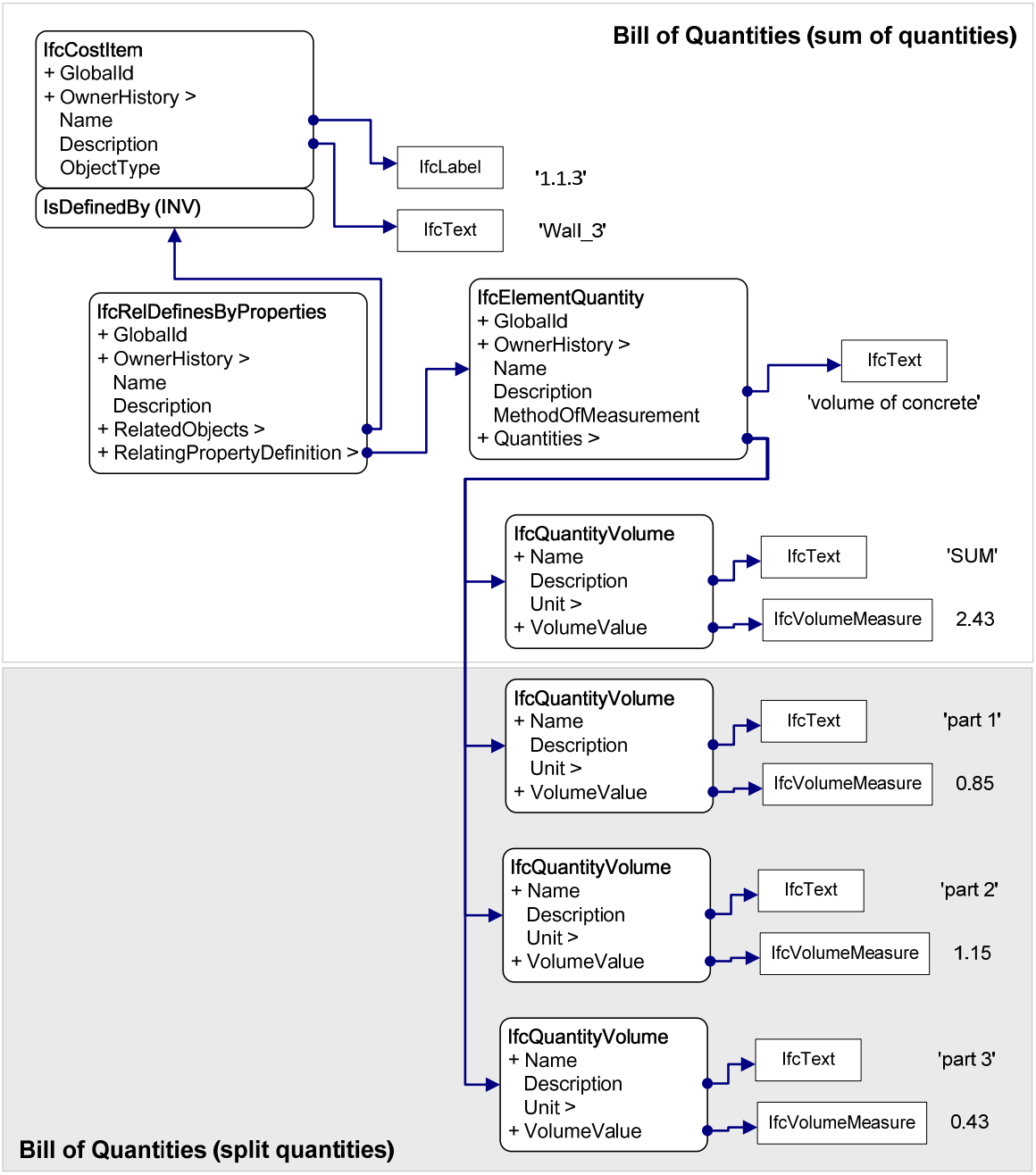
IAI, IFC	http://www.iai-international.org/
IDM	http://idm.buildingsmart.no/confluence/homepage.action
IFC2x3	http://www.iai-tech.org/products/ifc_specification/ifc-releases/Ifc2x3-tc1-release/summary
IFC2x4 (alpha)	http://www.iai-tech.org/products/ifc_specification/ifc-releases/ifc2x4-release/alpha-release
IFD	http://dev.ifd-library.org/index.php/Main_Page/
ISG	http://www.iai.hm.edu/
MVD	http://www.blis-project.org/IAI-MVD/
MSG	http://www.iai-tech.org/

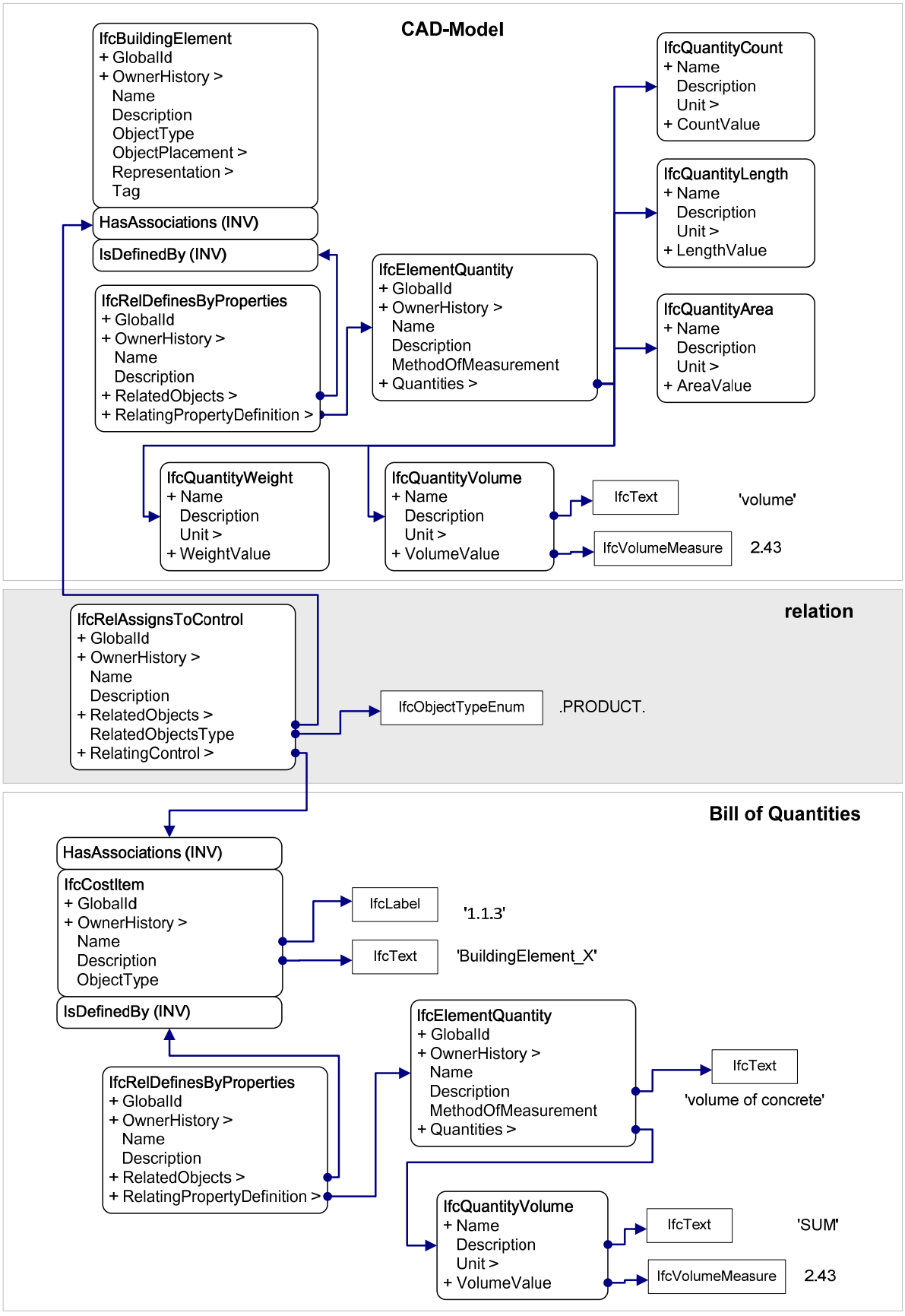
6 APPENDIX


Cost plan
IfcRelSchedulesCostItems
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
RelatedObjectsType
+ RelatingControl >
IfcCostSchedule
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
SubmittedBy >
PreparedBy >
SubmittedOn >
Status
TargetUsers >
UpdateDate >
+ ID
+ PredefinedType
IfcIdentifier
'Costplan_Building_A'
IfcCostScheduleTypeEnum
.COSTPLAN.
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IfcLabel
'1'
IfcText
'structural works'
IfcRelNests
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatingObject >
+ RelatedObjects >
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IfcLabel
'1.1'
IfcText
'steel works'
HasAssociations (INV)
IfcRelAssociatesAppliedValue
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
+ RelatingAppliedValue >
IfcCostValue
Name
Description
AppliedValue >
UnitBasis >
ApplicableDate >
FixedUntilDate >
+ CostType
Condition
IfcMonetaryMeasure
350,000.00
IfcLabel
'Estimated cost'
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IfcLabel
'2'
IfcText
'finishing works'


Bill of Quantities (structure)
IfcRelSchedulesCostItems
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
RelatedObjectsType
+ RelatingControl >
IfcCostSchedule
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
SubmittedBy >
PreparedBy >
SubmittedOn >
Status
TargetUsers >
UpdateDate >
+ ID
+ PredefinedType
IfcIdentifier
'BoQ_Building_A'
IfcCostScheduleTypeEnum
.UNPRICEDBILLOFQUANTITIES.
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IfcLabel
'1'
IfcText
'structural works'
IfcRelNests
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatingObject >
+ RelatedObjects >
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IfcLabel
'1.1'
IfcText
'concrete walls'
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IfcLabel
'1.2'
IfcText
'concrete slabs'
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IfcLabel
'2'
IfcText
'finishing works'


Bill of Quantities (sum of quantities)
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IsDefinedBy (INV)
IfcLabel
'1.1'
IfcText
'concrete walls'
IfcRelDefinesByProperties
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
+ RelatingPropertyDefinition >
IfcElementQuantity
+ GlobalId
+ OwnerHistory >
Name
Description
MethodOfMeasurement
+ Quantities >
IfcText
'volume of concrete
IfcQuantityVolume
+ Name
Description
Unit >
+ VolumeValue
IfcText
'SUM'
IfcVolumeMeasure
8.64
optional (multi object level)
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IsDefinedBy (INV)
IfcLabel
'1.1.1'
IfcText
'Wall_1'
IfcRelDefinesByProperties
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
+ RelatingPropertyDefinition >
IfcElementQuantity
+ GlobalId
+ OwnerHistory >
Name
Description
MethodOfMeasurement
+ Quantities >
IfcText
'volume of concrete'
IfcQuantityVolume
+ Name
Description
Unit >
+ VolumeValue
IfcText
'SUM'
IfcVolumeMeasure
3.44
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IsDefinedBy (INV)
IfcLabel
'1.1.2'
IfcText
'Wall_2'
IfcRelDefinesByProperties
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
+ RelatingPropertyDefinition >
IfcElementQuantity
+ GlobalId
+ OwnerHistory >
Name
Description
MethodOfMeasurement
+ Quantities >
IfcText
'volume of concrete'
IfcQuantityVolume
+ Name
Description
Unit >
+ VolumeValue
IfcText
'SUM'
IfcVolumeMeasure
5.20
IfcRelNests
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatingObject >
+ RelatedObjects >


Bill of Quantities (sum of quantities)
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IsDefinedBy (INV)
IfcLabel
'1.1.3'
IfcText
'Wall_3'
IfcRelDefinesByProperties
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
+ RelatingPropertyDefinition >
IfcElementQuantity
+ GlobalId
+ OwnerHistory >
Name
Description
MethodOfMeasurement
+ Quantities >
IfcText
'volume of concrete'
IfcQuantityVolume
+ Name
Description
Unit >
+ VolumeValue
IfcText
'SUM'
IfcVolumeMeasure
2.43
IfcQuantityVolume
+ Name
Description
Unit >
+ VolumeValue
IfcText
'part 1'
IfcVolumeMeasure
0.85
IfcQuantityVolume
+ Name
Description
Unit >
+ VolumeValue
IfcText
'part 2'
IfcVolumeMeasure
1.15
IfcQuantityVolume
+ Name
Description
Unit >
+ VolumeValue
IfcText
'part 3'
IfcVolumeMeasure
0.43
Bill of Quantities (split quantities)


CAD-Model
IfcBuildingElement
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
ObjectPlacement >
Representation >
Tag
HasAssociations (INV)
IsDefinedBy (INV)
IfcRelDefinesByProperties
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
+ RelatingPropertyDefinition >
IfcElementQuantity
+ GlobalId
+ OwnerHistory >
Name
Description
MethodOfMeasurement
+ Quantities >
IfcQuantityCount
+ Name
Description
Unit >
+ CountValue
IfcQuantityLength
+ Name
Description
Unit >
+ LengthValue
IfcQuantityArea
+ Name
Description
Unit >
+ AreaValue
IfcQuantityWeight
+ Name
Description
Unit >
+ WeightValue
IfcQuantityVolume
+ Name
Description
Unit >
+ VolumeValue
IfcText
'volume'
IfcVolumeMeasure
2.43
relation
IfcRelAssignsToControl
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
RelatedObjectsType
+ RelatingControl >
IfcObjectTypeEnum
.PRODUCT.
Bill of Quantities
HasAssociations (INV)
IfcCostItem
+ GlobalId
+ OwnerHistory >
Name
Description
ObjectType
IsDefinedBy (INV)
IfcLabel
'1.1.3'
IfcText
'BuildingElement_X'
IfcRelDefinesByProperties
+ GlobalId
+ OwnerHistory >
Name
Description
+ RelatedObjects >
+ RelatingPropertyDefinition >
IfcElementQuantity
+ GlobalId
+ OwnerHistory >
Name
Description
MethodOfMeasurement
+ Quantities >
IfcText
'volume of concrete'
IfcQuantityVolume
+ Name
Description
Unit >
+ VolumeValue
IfcText
'SUM'
IfcVolumeMeasure
2.43