

# IFC Model View Definition Format

Author	Jiri Hietanen
Date	08.04.2006
Version	1.0
Status	Final
History	04.04.2006 – Officially accepted by IAI ITM 07.04.2006 – Officially accepted by IAI IC 08.04.2006 – Final version
Copyright	Copyright © 2006 International Alliance for Interoperability

## Table of contents

Executive summary .....	2
Acknowledgements .....	3
Background .....	4
The larger picture.....	4
Re-using definitions .....	6
Definitions and configurations.....	7
Patterns .....	9
Extensibility.....	9
Partial use of the format.....	10
Implementers agreements .....	10
Quality control.....	11
The role of IAI .....	12
Lifecycle of IFC Model View Definitions .....	13
Process of creating IFC Model View Definitions .....	14
Roadmap for IFC Model View Definition work .....	14
IFC Model View Definition format .....	16
Overview.....	16
Concepts.....	17
Documents.....	18
Diagrams .....	22

## ***Executive summary***

Traditionally IFC Model View Definitions have been understood as subsets of the IFC Model Specification and have been defined primarily for IFC implementation purposes. The format defined in this document covers the same scope. However, it is important to understand the connections it has to a larger picture (IDM<sup>1</sup>), in which requirements come from the value chain of the end user and the primary role of IFC Model View Definitions is to ensure that IFC implementations support those requirements.

For definition of IFC Model Views the goal was set to “finding a useful balance between the wishes of users/customers and the possibilities of software developers, and documenting the outcome clearly.” The IFC Model View Definition Format is used for documenting this outcome.

The format must be well defined and unambiguous, but the format is only one part of what is needed.

- **Format:** The type of data that needs to be captured and how that data is structured
- **Content:** The data that is needed in a specific case. For example the IFC Schema is content that is captured using the EXPRESS format and an IFC Model View Definition is content that is captured using the IFC Model View Definition format.
- **Process:** The roles and responsibilities of different involved parties, for example how a model view definition becomes official and how certification is organized.
- **Tools:** The tools used for creating content, e.g. Process Maps and Exchange Requirements, and managing the process of creating content. Tools are highly important, but the format itself must be independent from any specific tools.

Although the format is in theory independent from the other parts it must in practice support all of them. It is also clear that the format is not the full answer, but having a commonly agreed format is the starting point. Without a common format it is very difficult to reuse content and tools, or to define a clear process.

For the process defining the role of IAI is the most important single factor. This is largely a question of resources IAI has at its disposal and how those resources are applied.

This is the first version of the official format, and it is to be expected that some details have to be adjusted later. However, the ideas behind this format have been thought out very carefully and should be stable. This format is based on the IFC Model View Definition and related formats developed by the BLIS<sup>2</sup>, ProIT<sup>3</sup> and IDM<sup>4</sup> projects and it has been developed and validated by the people behind these efforts.

---

<sup>1</sup> Information Delivery Manual

<sup>2</sup> <http://www.blis-project.org>

<sup>3</sup> <http://virtual.vtt.fi/proit>

## ***Acknowledgements***

A large number of people and organizations have contributed to the content of this document. In general this document does not present anything groundbreaking or completely new. The work can rather be characterized as an attempt to harmonize and connect different ideas and efforts related to the implementation of the IFC Model Specification and the practical use of such implementations. Anyone active in this area has contributed in one way or another, especially people involved in BLIS, ProIT, IDM and IFC implementation groups.

The work on this official IFC Model View Definition format was initiated by a proposal from BLIS at ITM Summit #29 in Madrid, February 2005. That proposal and all subsequent work by the author were made possible through funding from the Finnish VBE2 project. In the first stage Kari Karstila and Jeff Wix contributed to the harmonization work between the approaches of BLIS, ProIT and IDM. Later also Janne Marit Aas-Jakobsen, Kjetil Espedokken, Ole Kristian Kvarsvik and Sakari Lehtinen had a major impact on the outcome of this work. As a result of this harmonization the format has been improved to enable a much stronger end user focus than was originally envisioned. Valuable feedback was received from discussions with Vladimir Bazjanac, Chuck Eastman, Kent Reed and Richard See from IAI TAG. In addition a large number of people participated actively in related meetings, online training sessions and presentations, creating and refining the material that is used in this document.

Make everything as simple as possible,  
but not any simpler.

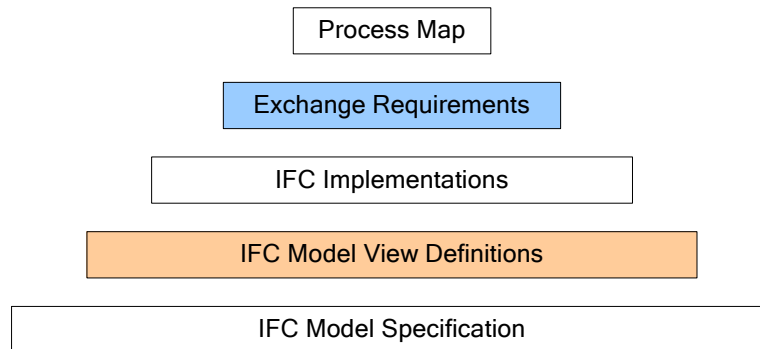
- *Albert Einstein* -

---

<sup>4</sup> <http://idm.buildingsmart.no>

## Background

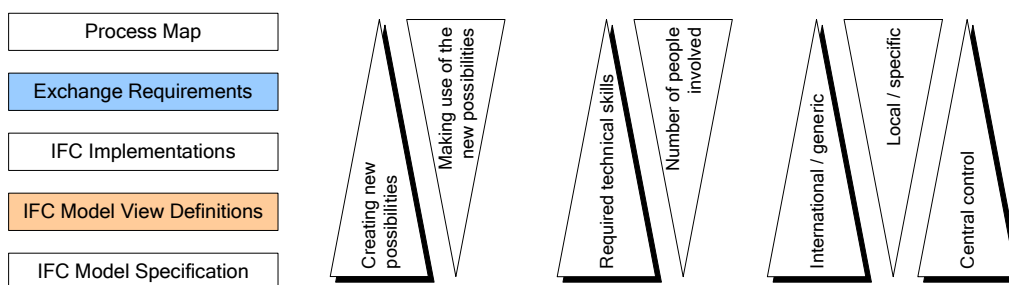
### The larger picture



**Figure 1** Steps required for reaching deployment of IFC based solutions <sup>5</sup>

The figure above shows the different steps that are needed for creating IFC based interoperable solutions that are successfully deployed in AEC/FM projects. It is like a ‘task list’ for all the things that must be taken care of. The picture is shaped like a pyramid, because the shortcomings of any level limit the possibilities of the levels above it.

- **IFC Model Specification** is the IFC schema and its documentation
- **IFC Model View Definitions** document how the IFC Model Specification is applied in the data exchange between different application types.
- **IFC Implementations** are the IFC import and export capabilities of software applications
- **Exchange Requirements** document the information that must be passed from one business process to enable another to happen.
- **Process Map** gives an overview of the end user process, describing its objective and describes the stages in a project at which the process is expected to be relevant.



**Figure 2** General trends

It is important to identify some general trends related to the larger picture. In general lower levels are creating new possibilities for the levels above them. For this effort to be successful there has to be demand on the higher levels for

<sup>5</sup> Based on “The Interoperability Pyramid” (Hietanen, 2003)

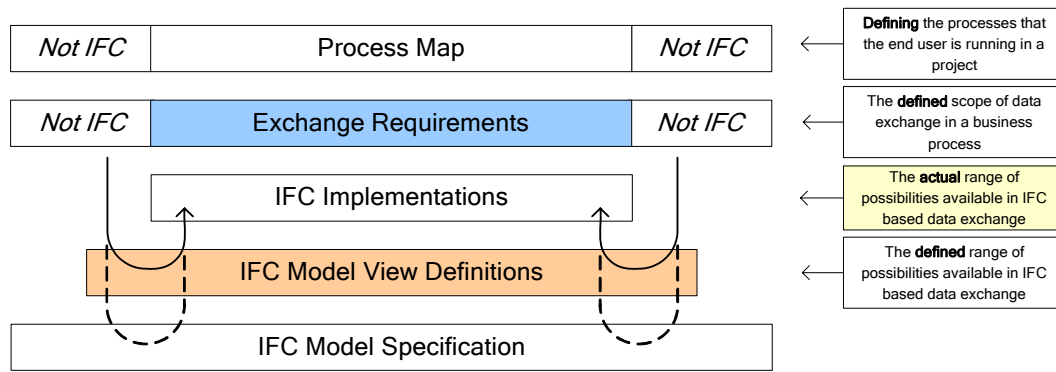
such new possibilities. In this sense all efforts should ultimately be driven by the requirements of deployment; without deployment the whole system has no reason to exist. But of course there are technological innovations, which nobody can really demand before they exist. In such cases the lower levels have to take a risk and assume that such new possibilities will eventually be accepted and deployed. Because of these dynamics each level needs to be in active dialog with at least the levels directly below and above. The formats and methodologies used must facilitate this dialog. For example the end result of software certification must enable software users to understand the possibilities and limitations of different software in IFC based data exchange. On the other side software implementers must understand the IFC Model View Definitions on which their IFC implementations are based, but they should not be required to know the whole IFC Model Specification.

The number of people involved will increase dramatically when IFC based interoperability becomes standard practice. Because of this it is important to build a system, in which the required level of technical skills decreases proportionally whenever the number of people increases. This may be as dramatic as having a dozen people working on the IFC Model Specification and several million in deployment. The success of deployment can not be allowed to be directly dependent on a small group of IFC experts.

The IFC Model Specification is an international standard and construction activity is by nature local. Somewhere in between a transition from international to local has to take place. Software is increasingly international and IFC Model View Definitions, which sit between the IFC Model Specification and IFC Implementations, are by this logic also international. Of course software is localized to specific markets and there is still a large number of purely local software. Exchange Requirements are much closer to local processes, including design contracts, but contain many universal elements. One major challenge is creating a system, which allows generic, internationally applicable solutions to be successfully reused and applied in specific local situations.

It is also natural that any central control gets weaker closer to deployment. The IFC Model Specification can be carried out as a centralized international effort and official software certification can at least be facilitated and controlled by the IAI. After that IAI International may at best have an unofficial role in e.g. creating guidelines or documenting and publishing experiences and best practices.

On the higher levels standardization of the Information Delivery Manual is to give the end users of a BIM a set of quality assured IDMs to pick from when modeling the localized company or project specific building or FM process. The standardization needs therefore to be both at the international level, with a core of international acceptable contracts of data interchange, and at the national level, where national requirements are built into the set of IDMs.



**Figure 3** Defined and actual possibilities

It is important to understand the role of IFC Implementations, because it is slightly different from the other layers. It is relatively easy to define the data that must be exchanged in a business process and how the IFC Model Specification should be used for exchanging the required data. But this data cannot be exchanged using IFCs before the corresponding capabilities and IFC implementations exist in software. IFC implementations also have to reach a certain level of robustness and general usability before they are accepted by the majority of users. From this perspective IFC implementations define what can really be done with IFC based data exchange at any given point in time. Exchange Requirements may however define data which is outside the scope of existing IFC implementations. In such cases the Exchange Requirement serves as a requirements definition for future IFC Model View Definitions, software development and IFC implementations. In the mean time such parts of Exchange Requirements have to be satisfied with other solutions than IFC based data exchange.

## Re-using definitions

The ability to reuse definitions is one of the major goals of the IFC Model View Definition format. The main enabling mechanism is concepts. All advanced view definition formats follow this idea; in ProIT they are called “aspects”, in IDM “functional parts” and “units of functionality” by ISG.

Concepts are independent from any IFC Model View Definition. Technically an IFC Model View Definition is created by choosing (or defining) a group of concepts and defining their relationships. For example a “rectangular profile” concept could be selected into an IFC Model View Definition, but defined to only be used with beams and columns, not spaces and walls.

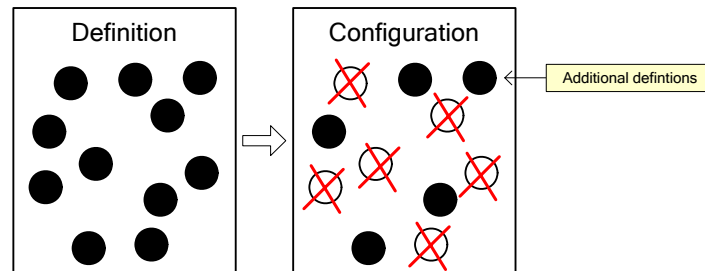
One form of re-use is to separate the idea of a concept (blue) from the IFC binding of that concept (orange). This makes it possible to re-use the same concept ideas when the underlying IFC Model Specification changes. For example the idea of a “space name” does not change if moved at some point from `IfcSpace.LongName` to some other location in the IFC Model Specification.

Although not a part of any official format, software tools are highly important for re-using definitions. The format defines a system which makes it possible to re-use definitions, but tools can make it much easier to know what has al-

ready been defined. Tools also help share the definitions with large groups making it less likely that the same definitions are reinvented.

## Definitions and configurations

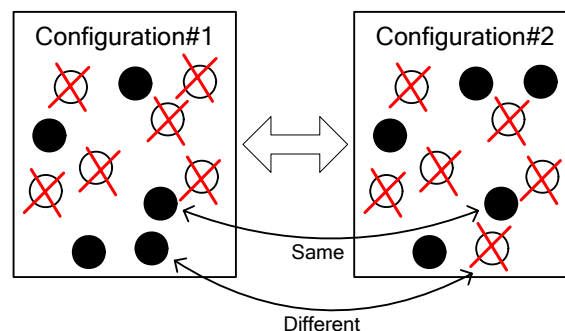
The IFC Model View Definition format makes extensive use of two very general ideas; definitions and configurations. Definitions capture a range of possibilities and configurations capture how those possibilities are used in a specific case. There can be many different configurations for any definition, because the same possibilities may be used in different ways in different situations.



**Figure 4** Definitions and configurations

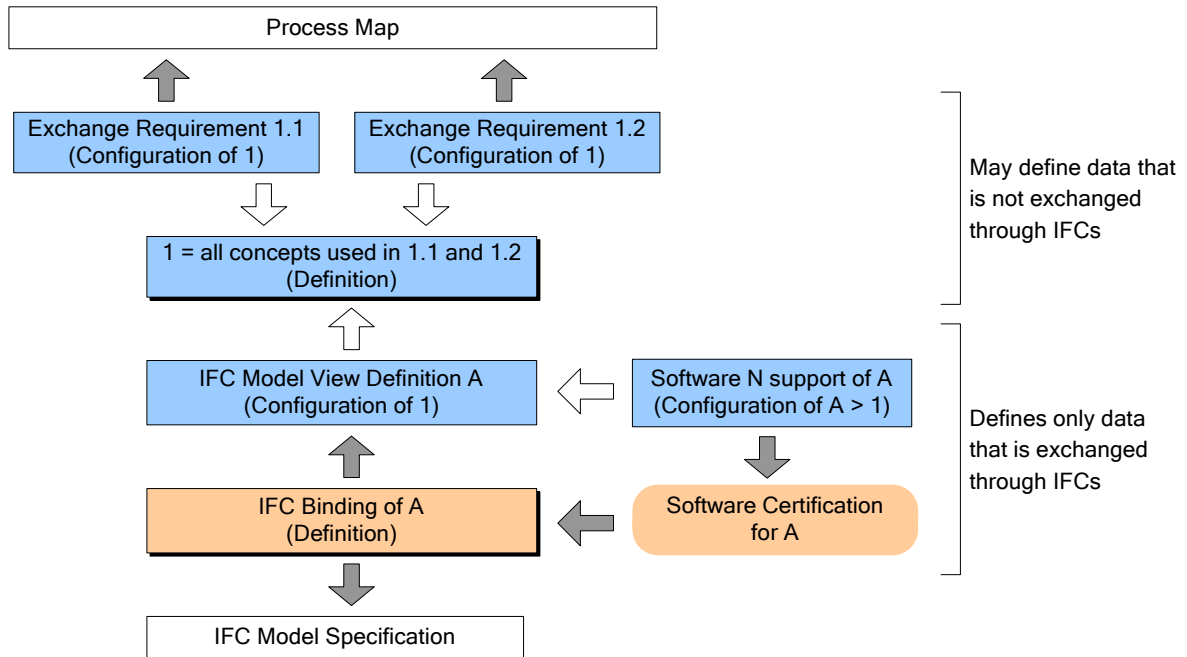
A configuration does two things; first it defines a subset of the available possibilities, then it defines in more detail how that subset is used. Configurations reduce scope, but are more specific about how the remaining scope is used. Additional definitions may not be in conflict with the original definition and it is not possible to extend the scope of the original definition with configurations.

Each IFC Model View Definition is a configuration of the IFC Model Specification. They define a subset of the IFC Model Specification and add new definitions, called implementers agreements, to it. The end results of software certification are software specific configurations of the IFC Model View Definition on which the certification was based. Software may not support the full scope of an IFC Model View Definition and often there is need for additional definitions, e.g. that imperial units are converted and imported as metric or some type of geometry is imported in a simplified form.



**Figure 5** Comparing configurations of the same definition

It is possible to compare configurations of the same definition. If the definitions and configurations are documented using a computer interpretable format (schema) it is possible to use software for such comparisons.



**Figure 6** Application of definitions and configurations

In the figure above all blue boxes are comparable, because they are all based on the same definition. This common definition can be created by merging business process driven Exchange Requirements that can be satisfied using the same type of software. The result is a collection of concepts and relationships between concepts.<sup>6</sup> When the system is built like this any blue box can be compared with any other blue box, which provides answers to some practical questions.

- What data can be exchanged between two software products using IFCs and what are the limitations of this data exchange? Compare the configurations of the software products for the same IFC Model View Definition.
- How well does a software product support an Exchange Requirement? Compare the configuration of the software product with the configuration of the Exchange Requirement.
- What new data is required when moving from one project stage to another? Compare the configurations of two Exchange Requirements.

<sup>6</sup> This corresponds in IDM terminology to a collection of functional parts. In this case to one functional part would typically be equal to a set of concepts.

## Patterns

Definitions, especially view definition diagrams, are only comparable if they follow the same basic structure. The official format doesn't define such a structure, but this can be solved with patterns.

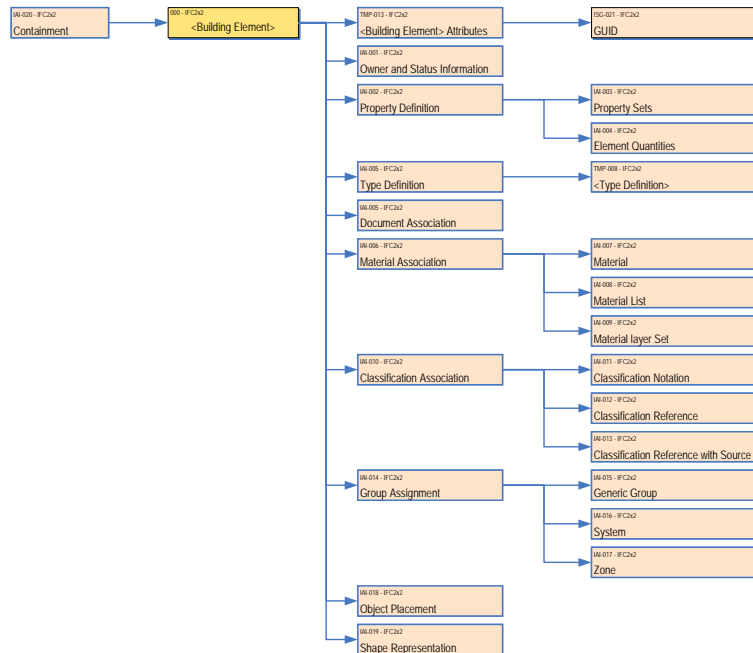


Figure 7 Example of a pattern for building elements

A pattern can be used as a starting point for a new definition. Using a pattern ensures that different definitions have at least the same basic structure and can be better compared and harmonized. Over time patterns will become less important, because most new definitions will anyway be extensions of existing definitions.

## Extensibility

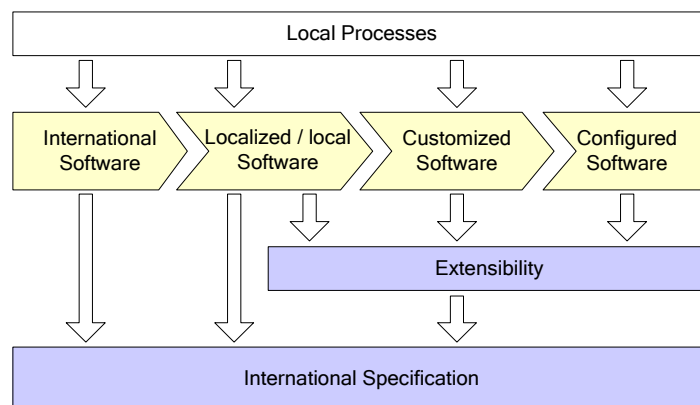


Figure 8 Local extensions in IFC based data exchange

When IFC based data exchange is used in a local process it common that the exchange has to be adjusted in some way. Software developers are already providing support for this. International software is typically localized for differ-

ent markets. Software can also be customized by end user organizations and configured for use in a project. The IFC Model Specification also contains a wide range of extension mechanisms, including property sets, element quantities, classification references, document references and proxy objects. For successful use of IFC based data exchange these two worlds must be connected.

Software localization may in some cases include extensions to IFC implementations but for customization and configuration this is not an option, because IFC implementations at least to date can't be extended by users. This problem can be solved by defining an "Extensibility" IFC Model View Definition, which defines in generic terms which extension mechanisms of the IFC Model Specification are made available to software users and how support for them is implemented in software. Software certified against this definition could be customized or configured by users for exchanging local content based on local agreements.

---

### **Partial use of the format**

Creating proper content all the way from IFC Model Specifications to deployment is a major task and takes some time to do. Especially with schedule and budget constraints there will be strong temptation to take shortcuts. Such shortcuts would lead to a partial use of the format.

One obvious shortcut is to create IFC Model View Definitions without having formalized the end user process and the data required by that process first (Process Map, Exchange Requirement). In some cases this may work well, but there is a real danger of neglecting important data and putting a lot of effort into supporting exchange largely irrelevant data. It would also be more difficult to communicate to the users what can be done with IFC based data exchange, because the exchange would not be directly related to their processes.

If the system cannot for practical reasons be fully used, it should at least be followed on a higher level. In the example above this would mean creating a few pages documentation about the processes for which the IFC Model View Definition is targeted and the data assumed to be required by those processes. If such ideas are clear documenting them on this level should not take more than a few days. If such ideas are not clear this should be a clear warning that the result of the work may actually not be very useful to end users.

---

### **Implementers agreements**

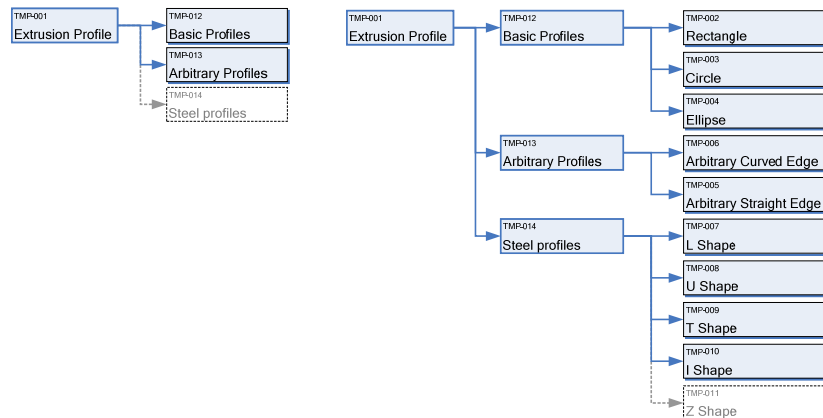
In the IFC Model View Definition format there are no separate implementers agreements, all agreements are captured in the IFC bindings.

The high level description of an IFC binding contains all agreements which are not specific to the use of individual concepts. This would cover cases like: use IFC2x2 property sets in IFC2x implementations. There will probably not be many agreements on this level, maybe even none.

Agreements about how the structure of the IFC Model Specification is used are captured in the IFC binding diagrams. This would cover cases like: all

spaces have to be contained by a building storey, or the area of a space is exchanged using element quantities.

One rule that should be of special interest to implementers is that a static concept has to be fully supported and there are no options inside a static concept. For software users the capabilities of IFC implementations are easier to understand if individual static concepts have a large scope. For implementations large concepts can be problematic because software is very different and a large concept may be discriminating.



**Figure 9** Large concepts vs. small concepts

In the example above the software user would like to know if steel profiles can be exchanged. However, if an application can support all other steel profiles but not 'Z Shape', the certification results would say that the application doesn't support the exchange of steel profiles. Whenever this is a problem concepts have to be defined on a more granular level.

Detailed agreements are captured in the IFC binding of individual concepts. This would cover cases like: the name of a space is exchange using `IfcSpace.LongName` and the number of a space using `IfcSpace.Name`

Implementers agreements provide the information used in software certification. Certification test cases cannot be generated automatically from the format, but the format allows capturing all information needed for creating these test cases manually.

## Quality control

In such a large and distributed system it is not possible to control the quality centralized or with any single mechanism. Instead each layer is responsible for the quality of its own output and the quality should be controlled when moving up to the next level.

- Each layer makes certain promises and quality is controlled by checking if those promises have been kept. Implementers for example promise to implement IFC support in a certain way and certification checks if they have done so.
- The quality of each level is largely dependent on the quality of the levels below it. Software users for example should be interested in proper

software certification, because quality problems in IFC implementations may, if unnoticed, affect the quality of their own work.

- The source level of any quality problem has to be identified. Problems should always be solved at the 'source' and not by workarounds on the higher levels.

The following are examples of questions need to be asked when the quality of the different levels is verified.

- **IFC Model Specification:** Is the specification a valid EXPRESS schema? Is the specification consistent within its own modeling rules and guidelines?
- **IFC Model View Definition:** Does the definition follow the model view definition guidelines? Is it documented using the official format? Does it make correct use of the IFC Model Specification? Is there overlap with other IFC Model View Definitions?
- **IFC Implementation:** Has the IFC Model View Definition been implemented correctly? What are the limitations of an implementation and how are users notified about such limitations during data exchange? This is the traditional IFC certification.
- **Exchange Requirements:** Has the Exchange Requirement been defined according to the exchange requirement guidelines? Is it documented using the official format? Is it realistic in terms of existing software? Exchange requirements may contain parts that are outside the scope of existing IFC implementations but it should still be possible to satisfy such requirements with some IT solution. If not, the Exchange Requirement is a requirements definition and can't be used e.g. as part of a design contract.
- **Deployment #1:** Is a user (person or organization) in general capable of creating and delivering data which fulfills Exchange Requirements? This could be personal professional qualification or organization certification (like ISO 9001)
- **Deployment #2:** is the user (person or organization) holding his/her end of a contract in a real project?

Such questions highlight the roles and responsibilities of the different parties involved. For example end users should not be accountable for mistakes made by software implementers and vice versa. Software certification is checking that certain data can be exchanged. It is the responsibility of an end user that the required data is exchanged.

---

## The role of IAI

Obviously the IAI has a role to play in the whole picture, but it can't be responsible for doing all the work of controlling all of it. It is necessary to distribute the work and responsibilities to a large number of people and organizations in a way that doesn't endanger the original goal of IFC based software interoperability. Because of this the role of IAI has to be thought out carefully.

There are some clear roles for the IAI.

- The IAI is defining, documenting and publishing the IFC Model Specification
- The IAI defines the official format for IFC Model View Definitions
- IAI International or IAI Chapters may define or commission IFC Model View Definitions, but such MVDs have to go through the same process as any other MVD.
- The IAI is not developing software
- The IAI is organizing or endorsing IFC software certification

In general the following tasks should be taken care of on each level

- Choosing or defining a format for content
- Creating content
- Integrating or harmonizing content
- Verifying the quality of the content
- Publishing and promoting content

---

### Lifecycle of IFC Model View Definitions

IFC Model View Definitions have a life cycle, which spans from the idea (which probably originates from Exchange Requirements) to the time the definition is superseded by another definition.

**Draft:** Some interested party creates and documents a new IFC Model View Definition or extends an existing definition. Any person or organization is allowed to do this without restrictions or limitation, as long as the definition is clearly marked to have Draft status.

**Proposal:** If the author of an IFC Model View Definition wants to get an official status for the definition, it must be submitted to the IAI. Once submitted to the IAI the definition has Proposal status.

**Candidate:** When an IFC Model View Definition has been submitted to the IAI, the IAI will review it using the following criteria.

- Has it been documented using the official format?
- Does it make correct use of the IFC Model Specification?
- Is there overlap or conflicts with existing official definitions?

The proposal may be refined based on the feedback from IAI. Once the definition satisfies the set criteria it has Candidate status.

**Official:** Official software certification may be organized only for IFC Model View Definitions that have reached Candidate status. When more than one application has successfully been certified against the definition the definition is elevated to Official status. After this any software can apply for certification against exactly the same definition.

**Deprecated:** When an IFC Model View Definition is superseded by another definition it gets Deprecated status. In practice this means that no certification is organized any more for that IFC Model View Definition

---

## Process of creating IFC Model View Definitions

The following is the recommended process for creating IFC Model View Definitions. It assumes that the goal is to create an official definition. If the definition was a local extension to an existing definition the process would be much simpler.

The need for new IFC Model View Definitions should come from Exchange Requirements, which define what data is exchanged in a business process. In the transition from Exchange Requirements to IFC Model View Definitions it is necessary to translate from 'local' 'process' to 'international' 'application types'. The first result of this work is a standard one page description of the new IFC Model View Definition.

The one page description should be reviewed by other parties creating IFC Model View Definitions. The purpose is to find out if a suitable definition already exists or if an existing definition can be expanded. It may also be possible to find other parties interested in sharing the work.

An IFC Model View Definition cannot become official before it is implemented in software and the implementations certified. Before continuing it would be a good idea to have implementers involved and get at least an initial commitment that the definition will be implemented.

The first detailed definition should be the IFC release independent part, consisting of the required concepts and their relationships. It is important to study existing concepts and to re-use them whenever possible. Also the structure of existing definition (patterns) should be re-used as much as possible.

When the generic definition is done, the next step is to define the binding to a specific IFC release. This task requires strong involvement from the implementers, because this binding defines the necessary implementers agreements. Also in this stage re-using existing concepts and patterns is very important.

When the IFC binding is done the definition can be implemented in software. In parallel the definition can be proposed to the IAI and the review process for making it official can begin. Once a definition is accepted as a candidate, certification for it can be organized.

After successful certification of at least one software application on each side of the IFC Model View Definition it becomes official and is published by the IAI.

---

## Roadmap for IFC Model View Definition work

Ultimately all IFC Model View Definition work should be guided by the needs of deployment, i.e. by data that is needed in business processes. This is not a new idea to the IAI, since the IAI in the beginning relied on the work of domain teams for defining the scope of the IFC Model Specification. The work of these domain teams is best captured in volume 1 of the IFC R2.0 documentation.<sup>7</sup> After 1999 there has been IFC implementation and software certification activity based on view definitions from different sources. Because IFC Model

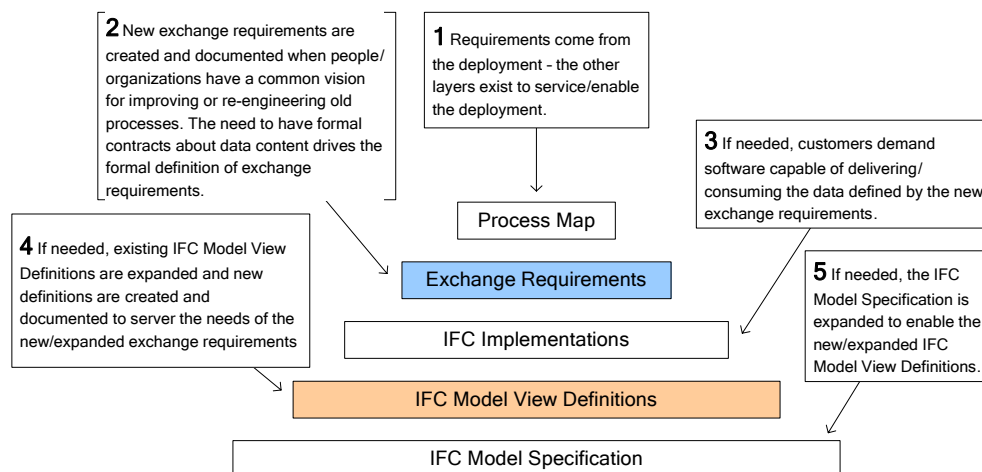
---

<sup>7</sup> IFC R2.0 "Vol. 1 AEC/FM Processes Supported by IFC" (See, 1999)

View Definition work is not starting from scratch there is need for a roadmap for the work.

1. Where are we today? Already implemented IFC Model View Definitions and related implementer's agreements should be documented using the official IFC Model View Definition format.
2. How can we make the most out of the existing possibilities? This generation of IFC Model View Definitions should mainly assume existing software and the current IFC Model Specification. This combination can already provide much better value than is available through existing IFC implementations.
3. What are the ultimate possibilities of IFC based exchange? This is a forward looking generation of IFC Model View Definitions, which require changes to existing software and/or the current IFC Model Specification. This stage assumes that software vendors are putting major effort into improving their software from the interoperability viewpoint. Typically strong customer demand is a prerequisite for this.

This roadmap does not mean a strict sequential approach, i.e. the different stages can be overlapping. The purpose is to give a general idea how to proceed towards the target process.



**Figure 10** Target process

## IFC Model View Definition format

### Overview

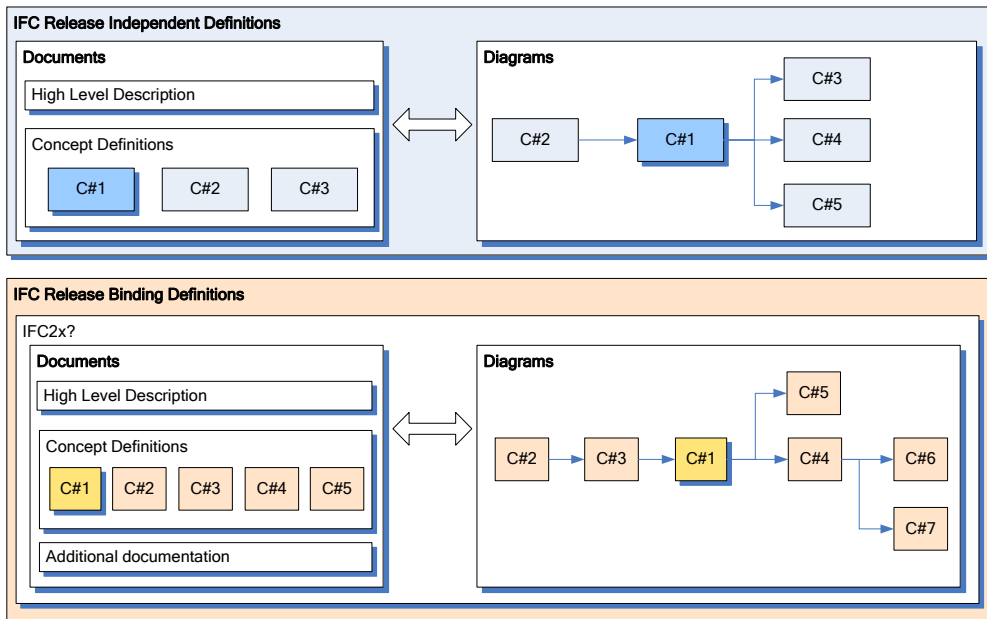


Figure 11 Format overview

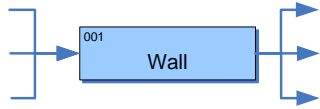
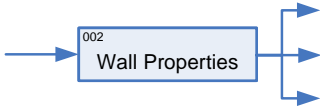
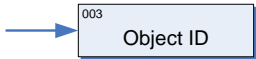
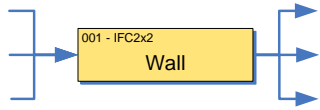
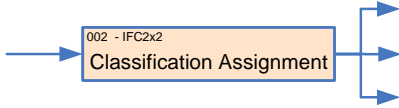
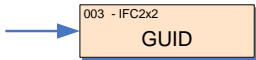
The format is divided into two main parts; IFC release independent (blue) and IFC release specific (orange). The IFC release independent definitions should be understandable for people without knowledge of IFCs or software implementation. The IFC release specific definitions are targeted towards people writing software code and require some knowledge of IFCs.

The IFC release independent part defines the concepts that are used in the data exchange in generic terms. It may even be used for defining concepts that are not exchanged through IFCs. The IFC release specific part is the binding of the generic concepts into a specific IFC release. It defines how the IFC Model Specification is used for exchanging the required data. Each supported IFC release will naturally have its own binding documentation.

Both parts are internally divided into documents and diagrams. Documents are used for capturing the idea of the view and the concepts used in the view. The same concepts can be reused in different views. The documents provide an unconnected collection of concepts. Diagrams are used for defining the relationships between the concepts in the context of a specific view. The IFC release independent diagram may be a configuration of an Exchange Requirement diagram.

## Concepts

The purpose of concepts is to allow a clear definition and reuse of ideas (blue) and software code (orange).

Generic Concepts	
<p>Variable Concept</p> 	<p>Variable concepts have the same name in different views, but their content may not be the same. Hence the variable concept must be configured separately for each case. This configuration is done by creating a diagram in which other concepts are connected to the variable concept.</p> <p>Examples: wall in architectural design to quantity take-off, wall in structural design to structural analysis</p>
<p>Group Concept</p> 	<p>Group concepts provide structure for the generic diagrams by grouping together static concepts and/or other group concepts. In some cases the group concepts themselves don't require any other definition than a name.</p> <p>Examples: wall geometry, door properties</p>
<p>Static Concept</p> 	<p>Static concepts remain the same in all scenarios in which they are used. They can be re-used without modification because they don't contain any options.</p> <p>Examples : Object ID, bounding box geometry</p>
IFC Binding Concepts	
<p>Variable Concept</p> 	<p>The IFC binding of a variable concept implements a generic variable concept with the same name.</p> <p>Example: the IFC Binding of the variable concept "Wall" is "Wall", not "Wall standard case"</p>
<p>Adapter Concept</p> 	<p>Adapter concepts are reusable parts of the IFC model specification that connect static concepts to a variable concept. There is no correspondence between adapter concept and generic group concept. Instead, adapter concepts provide a proposal how to structure software code in IFC implementations for reaching maximal code reuse.</p> <p>Examples: classification assignment, property set assignment</p>
<p>Static Concept</p> 	<p>The IFC binding of a static concept implements one or more generic static concepts. The names of generic and IFC binding static concepts don't have to match. The documentation of the IFC binding contains a detailed definition how to apply the IFC Model Specification.</p> <p>Example: "GUID" implements the generic static concept "Object ID".</p>

Each concept has an ID, which uniquely identifies the concept. The name is not used as the ID because definitions may be translated into different languages. The ID has the following format.

<Author ID>-<Concept Number>

### Examples

TEMP-001  
ABC-123  
IAI-057

The Author ID “IAI” is assigned only to concepts used in official IFC Model View Definitions. When a concept becomes official it gets the Author ID “IAI” and the next available number. The original ID is maintained in the history of the concept.

When used in a diagram each concept automatically receives a ‘fully qualified name’, which identifies it in the context of the diagram. This name is created by iterating from the concept through all parent concepts to the variable concept and finally to the IFC Model View Definition. The fully qualified name is used when definitions and configurations are compared with each other.



If the example above was from a IFC Model View Definition with the Reference “TEST-01”, the fully qualified name for “Construction Type Name” would be.

Test-01:TEMP-003:BLIS-004:BLIS-005:BLIS-006

---

## Documents

The official format for the documents is PDF. Microsoft Word templates are provided for creating the documents but any other software or system may be used as well.

The documents don’t contain a field for copyright. The document owner is the person or organization responsible for maintaining a document, i.e. the only one allowed to make changes to the document. This is done to prevent a situation in which there are several different, incompatible variations of the same definition.

<b>Generic AEC/FM View Description</b>					
<Title field>					
<b>Reference</b>	<Reference field>	<b>Version</b>	<Version field>	<b>Status</b>	<Status field>
<b>History</b>					
<b>Authors</b>					
<b>Document Owner</b>	<Company field>				
<b>Description</b>					
1 - What type of data is exchanged between what type of software					
2 - Diagram of the view					
3 - What is in scope for the view					
4 - What is out of scope for the view					
This document uses the official IAI View Definition Format version 1.0.6. The content of this document has to be certified by the IAI before becoming part of an Official IAI View Definition.					

**Figure 12** Template for the high level description of a view<sup>8</sup>

The purpose of the high level description is to provide a quick overview of the ideas of the view. The optimal length for this document is one page.

Field	Description
<Title >	The name of the IFC Model View Definition
Reference	The reference number of the IFC Model View Definition.  <Author ID>-<View Number>
Version	The sequential version number of the view
Status	The status of the view. Sample, Draft, Proposal, Candidate, Official or Deprecated
History	The history of the view, e.g. if the view has been created by harmonizing existing definitions.
Document Owner	The person or organization responsible for maintaining the document. Should contain some contact information, e.g. email address.
Description	1. What type of data is exchanged between what type of software 2. Diagram or picture explaining of the scope of the view 3. What is in scope for the view 4. What is out of scope for the view

<b>IFC Release Specific AEC/FM View Description (&lt;IFC Release field&gt;)</b>					
<Title field>					
<b>Reference</b>	<Reference field>	<b>Version</b>	<Version field>	<b>Status</b>	<Status field>
<b>History</b>					
<b>Authors</b>					
<b>Document Owner</b>	<Company field>				
<b>Description</b>					
1 – Which version of the generic view definition is being used					
2 – Basic principles applied when mapping the generic view to the specific IFC release, including implementer’s agreements.					
3 – Limitations relative to the generic definition					
This document uses the official IAI View Definition Format version 1.0.6. The content of this document has to be certified by the IAI before becoming part of an Official IAI View Definition.					

**Figure 13** Template for the high level description of the IFC binding of a view<sup>9</sup>

The purpose of the high level description of an IFC binding of a view is to document any general decisions that were made in the binding.

<sup>8</sup> GenericViewDescription.dot

<sup>9</sup> IfcReleaseSpecificViewDescription.dot

Field	Description
<IFC Release>	The IFC release for which the binding is defined
<Title >	The name of the IFC Model View Definition
Reference	The reference number of the IFC Model View Definition.  <Author ID>-<View Number>
Version	The sequential version number of the view
Status	The status of the view. Sample, Draft, Proposal, Candidate, Official or Deprecated
History	Any history specific to the IFC binding
Document Owner	The person or organization responsible for maintaining the document. Should contain some contact information, e.g. email address.
Description	<ol style="list-style-type: none"> <li>Which version of the generic view definition is being used</li> <li>Basic principles applied when mapping the generic view to the specific IFC release, including implementer's agreements.</li> <li>Limitations relative to the generic definition</li> </ol>

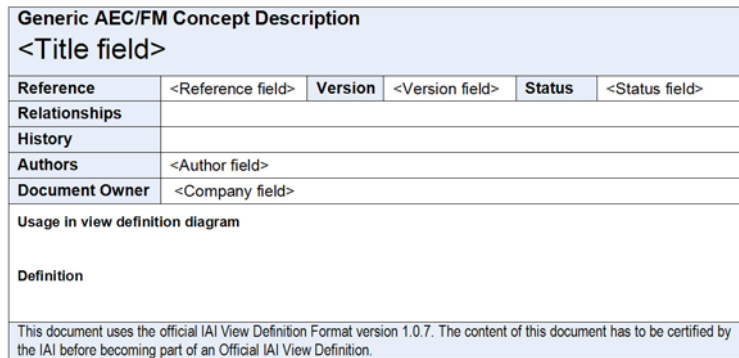


Figure 14 Template for generic concept definition<sup>10</sup>

The purpose of the generic concept definition is to document the idea of the concept independent from any data exchange format.

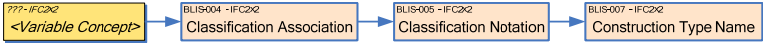
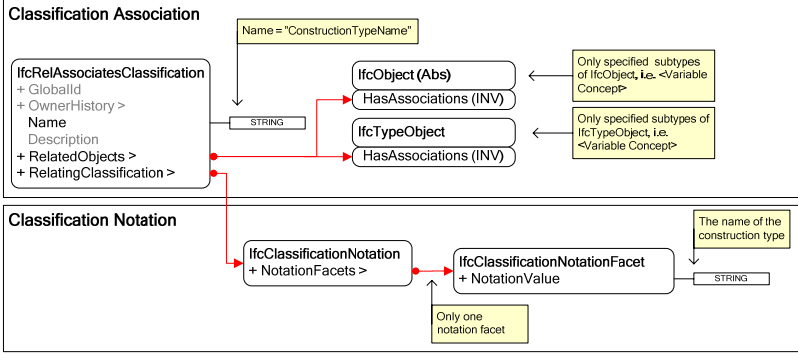
Field	Description
<Title >	The name of the concept
Reference	The reference number of the concept
Version	The sequential version number of the concept
Status	The status of the concept. Sample, Draft, Proposal, Candidate, Official or Deprecated
History	The history of the concept, e.g. if the concept has been created by harmonizing existing definitions.
Document Owner	The person or organization responsible for maintaining the document. Should contain some contact information, e.g. email address.
Description	<p>The “Usage in view definition diagram” defines the place of the concept in diagrams. Example:</p> <pre> graph LR     A["??? &lt;Variable Concept&gt;"] --&gt; B["??? &lt;Properties&gt;"]     B --&gt; C["BLIS-001 Construction Type"]     C --&gt; D["BLIS-002 Construction Type Name"]             </pre> <p>The “Definition” provides all necessary information for understanding the concept. If the definition is longer than one page it would probably be a good idea to break up the concept into several concepts.</p>

<sup>10</sup> GenericConceptDescription.dot

IFC Release Specific Concept Description (<IFC Release field>)					
<Title field>					
Reference	<Reference field>	Version	<Version field>	Status	<Status field>
Relationships					
History					
Authors	<Author field>				
Document Owner	<Company field>				
Usage in view definition diagram					
Instantiation diagram					
Implementation agreements					
This document uses the official IAI View Definition Format version 1.0.7. The content of this document has to be certified by the IAI before becoming part of an Official IAI View Definition.					

Figure 15 Template for IFC release specific concept definition<sup>11</sup>

The purpose of the IFC binding definition of a concept is to document how the concept must be implemented using a specific IFC release.

Field	Description
<IFC Release>	The IFC release for which the binding is defined
<Tile >	The name of the concept
Reference	The reference number of the concept
Relationships	Relationships to other definitions <ul style="list-style-type: none"> <li>• Implements : the generic concept implemented by the IFC binding</li> <li>• Extends : the adapter concept a concept is based on</li> </ul>
Version	The sequential version number of the concept
Status	The status of the concept. Sample, Draft, Proposal, Candidate, Official or Deprecated
History	Any history specific to the IFC binding
Document Owner	The person or organization responsible for maintaining the document. Should contain some contact information, e.g. email address.
Description	<p>The “Usage in view definition diagram” defines the place of the concept in diagrams. Example:</p>  <p>The “instantiation diagram” shows the IFC objects that need to be instantiated for the concept and any requirements on the attribute values. Instantiation diagrams may be freely commented and may contain clarifying drawings. Instantiation diagrams from other concepts may be inserted and further specified. Example:</p> 

<sup>11</sup> IfcReleaseSpecificConceptDescription.dot

Field	Description
	<p>There is no official format for instance diagrams, but the use of the notation in the example above is encouraged.</p> <p>The “Implementation agreements” are the official agreements made regarding the implementation. Most of them will already be shown in the instantiation diagram, but this is the official list used in certification. Example:</p> <ul style="list-style-type: none"> <li>• The Name attribute of IfcRelAssociatesClassification must have the value “ConstructionTypeName”. This value is not case sensitive.</li> <li>• Only subtypes of IfcObject and IfcTypeObject, which are specified by the view to have a construction type name, may be used.</li> <li>• The number of IfcClassificationNotationFacet objects is restricted to one.</li> <li>• The name of the construction type is captured by the Notation-Value attribute of IfcClassificationNotationFacet</li> </ul>

In addition the IFC binding definition of a concept may contain any number of optional definitions. Such definitions are not part of the official definition, but can make it easier to understand and implement the definition or help in creating certification test cases. Examples:

- EXPRESS-G diagram
- EXPRESS sub schema
- UML diagram
- Sample files

---

## Diagrams

The official format for diagrams and configurations is defined by a XML schema<sup>12</sup>. A Microsoft Visio template<sup>13</sup> is provided but diagrams and configurations may be created with any software or system. The Visio template can be used for reading and writing the official XML format.

The XML format for diagrams supports three different styles, which may be combined into the same XML dataset

- **Definition:** the concepts used in a diagram and their relationships in the context of that diagram.
- **Configuration:** The status of the concepts (ON/OFF) and diagram specific comments for concepts.
- **Layout:** The position, visibility and other layout related settings of concepts in a diagram. The layout is typically specific to an application, e.g. the MS Visio template uses a ‘Visio layout’. Layouts are not part of the official format, only the ability to define layouts.

This division makes it possible to create several configurations and layouts for the same definition.

---

<sup>12</sup> ViewDefinition.xsd

<sup>13</sup> View Diagram.vss

Since the official format for diagrams is an XML representation there is no official page size or orientation. Large diagrams have to be kept on one ‘page’, i.e. it is not allowed to split a diagram over several pages. For accommodating large diagrams the Visio template has function for hiding sections of the diagram. Such settings can be saved in a separate layout.

A separate View Diagram is created for each Variable Concept in the view.

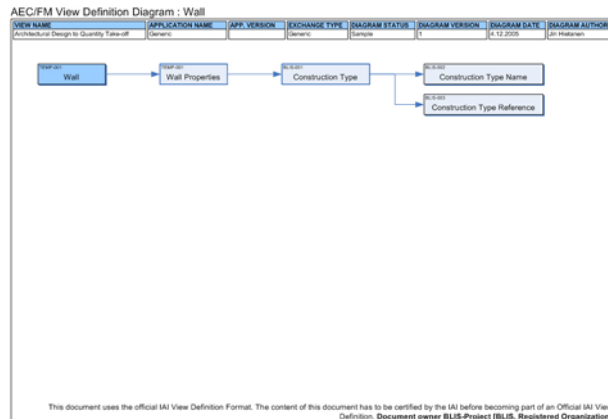


Figure 16 Template for generic view diagram

Field	Description
Diagram name	The name of the diagram is the name of the variable concept of the diagram. The name is shown in the title.
View Name	The name of the IFC Model View Definition
Application name (optional)	The name of the software application for which the diagram is made.
Application version (optional)	The version of the software application for which the diagram is made.
Exchange type	Generic, Import, Export or Roundtrip
Diagram status	Sample, Draft, Proposal, Candidate, Official or Deprecated
Diagram version	The sequential version number of the diagram
Diagram date	The data the version of the diagram was completed
Diagram authors	The authors of the diagram
Document Owner	The person or organization responsible for maintaining the diagram. Should contain some contact information, e.g. email address.

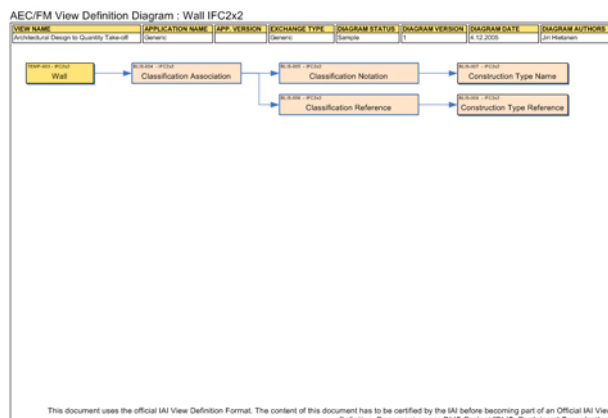


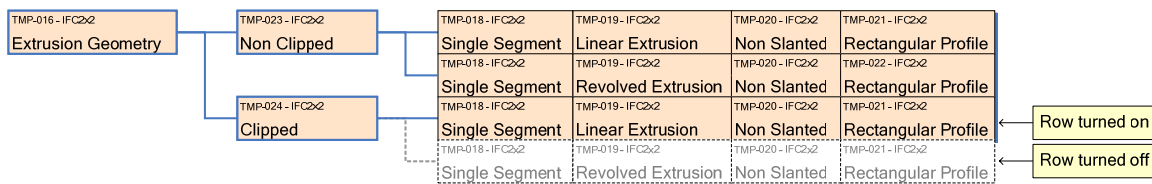
Figure 17 Template for IFC release specific view diagram

Field	Description
Diagram name	The name of the diagram is the name of the variable concept of the diagram. The name is shown in the title.
IFC Release	The IFC release the diagram is defining the binding for. The IFC release is shown in the title.
View Name	The name of the IFC Model View Definition
Application name (optional)	The name of the software application for which the diagram is made.
Application version (optional)	The version of the software application for which the diagram is made.
Exchange type	Generic, Import, Export or Roundtrip
Diagram status	Sample, Draft, Proposal, Candidate, Official or Deprecated
Diagram version	The sequential version number of the diagram
Diagram date	The data the version of the diagram was completed
Diagram authors	The authors of the diagram
Document Owner	The person or organization responsible for maintaining the diagram. Should contain some contact information, e.g. email address.

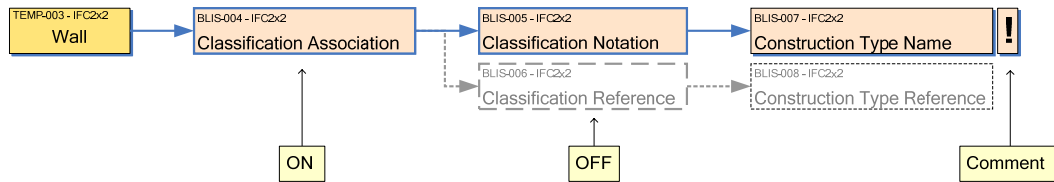


A diagram defines which concepts are used in a view and the relationships between those concepts. Static, group and adapter concepts may be placed on either side of the variable concept. There is no predefined meaning for the left and right side of the variable concept. Connectors in the diagram always point from left to right. Circular connections between concepts are not allowed and each concept may only be connected to one ‘parent concept’.

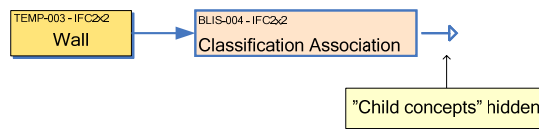
A concept may be marked ‘mandatory’ if the whole IFC Model View Definition doesn’t work or make sense if that concept is not supported. For example thermal analysis is not possible if spaces don’t have geometry suitable for this purpose. Mandatory should be used sparingly, only when absolutely necessary. Software not supporting all mandatory concepts of a view cannot pass certification for that view.



In some cases concepts need to be joined together to say “in the context of this diagram these concepts go together”. This is done by creating a table in which each row represents a set of joined concepts. In a table each row is either turned on or off.



Diagrams may be configured using two mechanisms; turning concepts off and adding comments to the concepts. In a configuration it is not allowed to delete concepts from the diagram. Turning a concept off is used for reducing the scope. If a concept is turned off it means that the concept is irrelevant or not supported in the context of the diagram. Commenting is used for being more specific about the scope that remains. In addition diagrams may contain any text or graphical elements, but such elements are not part of the official definition and will not be captured in the official XML format.



Large diagrams may be placed on one page by hiding concepts. Hiding a concept does not mean that it is turned off. Hiding is used purely for layout purposes.