# Using ‘project types’ in IFC R2.0

DRAFT 1 – 14.07.2000

‘Project types’ or ‘construction types’ are a way to identify the **type** an object in data exchange. Instead of a detailed description of the object, only the type of the object has to be exchanged between programs.

## *Benefits of the ‘project type’ approach*

There are two benefits to this approach, it is simple and it reflects the nature of data exchange between different application types. The simplicity comes from the fact that as a minimum only the type tag has to be exchanged between programs to identify the object. In addition to the tag also geometry, quantities and relationship information is exchanged, but the tag is the main means to **identifying** the object.

This method also reflects the way applications are able to communicate with each other. Different types of programs deal with the same objects (walls, doors, windows etc.) but their view of these objects is very different. For each application the type of an object means something different; an architectural design application is interested in the exact geometry of e.g. a wall, but a cost estimating application is interested in the composition of the same wall into materials and labour. In the same manner a thermal simulation application is interested in the thermal properties of that wall. Passing a detailed ‘view’ of an object to another program does not help much, because the receiving application is dealing with a different view of the object anyway.

## *Agreeing on the type tags*

When the detailed description of the objects is not communicated through the IFC file there has to be another way to agree what construction types are used in each project. (Because this is done on a project basis the types are called ‘project types’).

It would be beneficial if there existed a globally agreed list of construction types that could be used in projects. Since this is not the case and since it is very difficult to make even national or regional lists of this nature the project type has to be just a simple string. In the most optimistic scenario big construction companies or design teams can agree on a set of construction types that are used in more than one project. However, it does not make sense to hardcode such lists into specifications or design applications and consequently the agreement about the types has to be captured somewhere else.

The IFC model and the BLIS views of the IFC model have to rely on the existence of such agreed lists of project types, but they can’t define such lists themselves.

Some objects do not use the project type approach. For spaces it is possible to make a fairly generic list of space function types. If a more detailed description of a space type is needed this has to be done on a project level. HVAC and electrical fixtures have extensive databases from where a suitable component can be selected. These databases are not universal, but they are also not defined on a project basis. Furniture, fixtures, build-ins and all kinds of equipment are also not typed on a project basis.



Author: Jiri Hietanen (jiri.Hietanen@qpartner.com)

Although the identification tags for these are not defined on a project basis the tag still is a simple string or other generic variable. From the viewpoint of data exchange this is the same as having a project type, semantically calling it a project type could cause confusion.

## *Additional identification*

When an object is identified the first thing to look for is the object type (e.g. IfcWall). The object type defines the list of project types that is applicable for that object. The second thing to look for is the actual project type (e.g. exterior_wall_type-1). These two are the main identification information for most building elements.

In addition to this the object can have other kinds of information attached that can be used for identification and for validating the project type. A project type for a window contains the dimensions of that window, from the geometry the receiving application can check that the windows actually have the size they should have. A wall type can be defined as being e.g. an external wall, there is also a attribute on walls that defines if a wall is internal or external. These two pieces of information have to match. As the model becomes more complicated over time this kind of identification information becomes richer and richer, but still the type information is the one that defines the object.

On one hand this information is redundant and should be avoided, but on the other hand it is valuable information for 'debugging' the building design. The more we are automating e.g. the process of quantity takeoff the more important it becomes to make sure that the model that is being estimated is a valid model.

## *IFC R2.0 object listing*

### IfcWall

| Data | Location | Comments |
|---|---|---|
| Project type | Pset_WallCommon.Reference | |
| Internal / external | Pset_WallCommon.ExternalWall | |
| Fire rating | Pset_WallCommon.FireRating | |
| Height | *From geometry* | Standard geometry, extruded solid profile Xdim |
| Thickness | *From geometry* | Standard geometry, extruded solid profile Ydim |



Author: Jiri Hietanen (jiri.Hietanen@qpartner.com)

## IfcDoor

| Data | Location | Comments |
|---|---|---|
| Project type | Pset_DoorCommon.Reference | |
| Internal / external | Pset_DoorCommon.IsExterior | |
| Fire rating | Pset_DoorCommon.FireRating | |
| Hardware group | Pset_HardwareGroup.Reference | The hardware group is not dependent on the project type. |
| Height | *From geometry* | Standard geometry of containing opening element, extruded solid profile Ydim |
| Width | *From geometry* | Standard geometry of containing opening element, extruded solid profile Xdim |

## IfcWindow

| Data | Location | Comments |
|---|---|---|
| Project type | Pset_WindowCommon.Reference | |
| Internal / external | Pset_WindowCommon.IsExterior | |
| Fire rating | Pset_WindowCommon.FireRating | |
| Hardware group | Pset_HardwareGroup.Reference | The hardware group is not dependent on the project type. |
| Height | *From geometry* | Standard geometry of containing opening element, extruded solid profile Ydim |
| Width | *From geometry* | Standard geometry of containing opening element, extruded solid profile Xdim |

## IfcColumn

| Data | Location | Comments |
|---|---|---|
| Project type | Pset_ColumnCommon.Reference | |
| Profile shape | *From geometry* | Profile shape for extruded solid |
| Xdim | *From geometry* | Standard geometry, extruded solid profile Xdim |
| Ydim | *From geometry* | Standard geometry, extruded solid profile Ydim |
| Radius | *From geometry* | Standard geometry, extruded solid profile radius |

## IfcBeam

| Data | Location | Comments |
|---|---|---|
| Project type | Pset_BeamCommon.Reference | |
| Profile shape | *From geometry* | Profile shape for extruded solid |
| Height | *From geometry* | Standard geometry, extruded solid profile Xdim |
| Width | *From geometry* | Standard geometry, extruded solid profile Ydim |



Author: Jiri Hietanen (jiri.Hietanen@qpartner.com)

## IfcSlab

| Data | Location | Comments |
|---|---|---|
| Project type | Pset_SlabCommon.Reference | |
| Slab type | IfcSlab. PredefinedType | Floor / Roof |
| Fire rating | Pset_SlabCommon.FireRating | |
| Thickness | *From geometry* | Standard geometry, extrusion depth of profile |

## IfcRoof

| Data | Location | Comments |
|---|---|---|
| Project type | Pset_RoofCommon.Reference | |
| Fire rating | Pset_RoofCommon.FireRating | |



Author: Jiri Hietanen (jiri.Hietanen@qpartner.com)