Issues to discuss – Kaj A. Jørgensen

In addition to memo 1 from 2012-02-27, some further issues are raised in the following regarding the revision of ISO 12006-2.

Previous discussions indicate that there are a number of fundamental issues related to the scope of the standard, which apparently are interpreted differently. Hence, it seems necessary to establish an agreement about a common understanding. This memo is an attempt to give inspiration for seeking such an understanding. Included is also a description of selected basic concepts.

Use of classification in Construction Projects

A fundamental characteristic of classification tables is that they give an easy understandable and accessible overview over their content. Hence, provides tables can support users in selection among presented alternatives. The use of classification tables is very much related to finding and inserting externally represented data into construction projects and, as illustrated in Figure 1, different tables are needed in different life phases of a construction project. For example, in the very early phases, a function table could give inspiration about what functions should be required or provided by the construction entity.

Figure 1 – Use of classification tables in construction projects
This overall view of the use of classification tables is independent of what tools and technologies are applied to project but, in practice, users can be supported in many ways by use of dedicated software. Such software can support the use of classification tables in an automatic or semi-automatic mode, which is not possible in a pure document based approach.

Therefore, in order to make efficient and effective use of classification tables, it is important to use software, which can provide an integrated support throughout all phases. The basis for this is that relationships are established between tables in a classification system (illustrated in Figure 2). In practice, this means that codes, referring to one table, can support selection of codes in another table. This also implies that the first selected code is very important and underlines that modelling is a valuable basis.

Figure 2 – Relationships between classification tables can support efficient and effective work

Besides the use of classification to insert external represented data into construction projects, it must be stated that classification tables can be created for many other reasons. In general, the overall purpose of developing a classification table is to structure a selected set of concepts in order to give a good overview and to enable easy navigation.

Design (Modelling) and Specification

As indicated above, the use of classification tables is much related to design phase of construction entities and its parts. As also argued, an efficient and effective use of classification tables is based on utilisation of appropriate software. This means that current and future development of classification systems must take new trends in software technologies into consideration. Today, the use of modelling software has gained momentum and there is no doubt that this development will be continued and further amplified. It must be underlined that the concept “model” is used with a general understanding and not specifically computer based models. Any kind of description is in principle a model.

The most promising type of modelling software is categorised as object based, which means that the primary content of the models is a set of software objects representing the commonly understood parts of construction entities, e.g. walls, windows, lamps, switches, sinks, pipes, and fittings. For these objects, it is often primarily the 3D geometric shape that is represented in the software and, naturally, most modelling is about generating such model objects and putting them together in order to visualise the overall architecture of the construction entity. The general image of modelling is that it is rather constrained regarding the support for detailing.

By use of modelling software, new model objects are selected based on in-built libraries of object types. At first, such objects represent major sub-objects of construction entities and, according to general design theory, the model components are subsequently detailed along
three dimensions: *structure*, *specification*, and *specialisation*. Modelling of the structure is to sub-divide objects into sub-objects, ultimately down to objects representing products. Initially, the internal structure is given by the library types and the level of detail can be very different in the software products. Likewise, the objects instantiated from library types are born with a pre-defined set of attributes and some of them have values assigned. Next, specification modelling is about adding further attributes to model objects and to determine values of attributes, e.g. colour, material and fire resistance. Finally, specialisation is, from a given type, to select a sub-type, e.g. from a generic door to select a sliding door. Much development focus regarding modelling software and model based software is about support for model detailing and this means that an increasing number of tasks of the traditional specification process will be integrated in the modelling process or in use of dedicated software, which can utilise models with various degree of detail. Classification tables can support modelling in three ways: 1) when identifying and selecting types of new model objects, 2) when identifying sub-types, i.e. specialisation and 3) when performing specification modelling of the instantiated model objects.

As explained, model objects are created from object type libraries and the organisation of these libraries varies over software products. So, for users, it would be beneficial, if these libraries were based on existing classification tables, ideally international standardised tables. By implementing a library of object types in software, the available types can be presented by generation of different structures based on the values of attributes. *At least some of these structures should conform to standardised classification tables.* In a typical design approach, the initial object type is rather *general* and later on this type may be changed to more *special* types by selecting a sub-type of the existing type in the previously used classification table. Specialisation is, however, not suitable supported by current modelling software.

Use of classification tables in specification modelling is to assign codes, referring to entries of classification tables, to attributes of model objects. Current modelling software is not ideal regarding this. Very often, this must be performed as a manual and time consuming task. Alternatively, by the use of separate software like analysis or simulation software, much specification can be generated implicitly. However, as previously stated, if tables in a classification system are related to each other, the manual operations can be much better supported. In order to use such an approach, it is an advantage that the library types have an attribute, which refer to a specific entry in the corresponding classification table.

*Classification tables must enable support for modelling tasks by efficient and effective solutions in relationship the use of modelling software. Key issues are about the degree of detail, which include further structuring, specification of models, and specialisation.*

**Model Object Types and Specification Datasets**

As stated, modelling includes selection of external data to be inserted into models. Figure 3 give an overall range of datasets, which comply with general design and construction methodologies. For any given construction entity, the figure shows a set of typical modelling steps (the top row) and important overall results of the work in these steps (the bottom row). The rest of the figure (the middle row) includes 1) primary model object types (middle row left column, including spaces, components/parts, products and construction activities) and 2) typical examples of datasets (remaining part of middle row) to be used for specification of the model objects. It is important to make this distinction because model object types are the basis ("templates") for creating/instantiation of model objects, the data objects/containers, while the datasets are the basis for specification, i.e. selected data from these sets are added to the model objects as attributes. This gives a clearer view of the individual datasets and makes the development of classification systems much easier.
It must be underlined that components/parts, products and activities are by far not the only model object types but, to simplify, these are considered the most primary and other object types are subordinate and regarded mostly as implementation solutions. Some of those subordinate objects are handled automatic by modelling software, e.g. objects for description of geometry, location and relationships.

As shown, the identified datasets are not necessarily related to specific modelling steps but are overlapping multiple steps and illustrate modelling progression according to general design theory. In practice, many project approaches may be different regarding sequencing, degree of detail, resource allocation, etc.

Efficient and effective handling of these data is a key issue of modelling and, consequently, it is important to provide support of this. A beneficial support method is to develop classification tables for both each of the model object types and each of the specification datasets indicated in Figure 3. Such tables can then be used in the modelling steps as previously described. For the model object types, the standard ISO/PAS 16739 can be considered as a foundation. This standard is about the buildingSMART data model Industry Foundation Classes (IFC), which include a widely accepted taxonomy of object types, e.g. walls, columns, distribution elements, sensors, products and tasks. As stated and illustrated in Figure 2, it is important that classification tables are interrelated and also here, IFC can provide a foundation, i.e. mappings from the IFC object types to the tables for the specification datasets will support efficient and effective selection of specifications.

The key specification datasets for spaces are regarding user activities, functions, safety, accessibility, etc. Often, user activities and space functions are mixed but it is important to underline that they should be separated. Specification datasets for components/parts,
products and construction activities are many and some examples are shown in the figure. \textit{Forms} and \textit{structures} are important in the early phases and can support the specification of model objects with focus on the primary architecture and the overall organisation of the bearing system as well as the building services. Components/parts may belong to different \textit{systems} and may be specified by \textit{materials}, \textit{fire resistance}, \textit{environmental impact}, etc. \textit{Technical composites} primarily support the subdivision detailing process down to the degree of detail, necessary for construction to be carried out. In addition, some of the technical composites may be replaced by or linked to pre-fabricated \textit{products}. In relationship with parts, products and construction activities, datasets about \textit{disciplines/skills}, \textit{resources} should be available for detailed specification. As also shown in Figure 3, other classification and attributes/properties may be added continuously over many steps.

\textbf{Analysis and Synthesis}

Two basic characterisations of models are important as illustrated by the following Figure 4. There is a difference between models created based on \textit{analysis} and models based on \textit{synthesis}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure4.png}
\caption{Two kinds of models}
\end{figure}

Very often, when discussing issues about models, it is not made clear which kind of model is considered or if, perhaps, both kinds of models are covered. Usually, there is a big difference between talking about a model, which is created as an abstraction on the basis of \textit{real world objects} (analysis), and the other type of model, which in contrast is aimed at being used as the foundation for realisation of \textit{real world artefacts}. Think, for instance, on various definitions of the room concept. Many times, a room is defined as a space delineated by walls and slabs (the analysis view) but, in the early steps of a programming process, it should be possible to identify rooms solely from its user activities or functions (the synthesis view) and long before any delineation is defined.

\textit{Obviously, the standard must primarily address the synthetic view on construction as this relates to the major parts of the design and construction work. Some activities later in the life time may of cause also relate to the analytic view, e.g. in operations an maintenance.}

\textbf{The Object Concept}

As already indicated, the concept \textit{object} also has multiple meaning and a basic distinction could also be explained on the basis of Figure 3. There is a big difference between objects in
the real world and objects in the model world. In order to simplify, model world objects are in the following termed data objects.

A further distinction and clarification: real world objects have properties while data objects have attributes. Both real world objects and data objects may have a structure of internal objects. Attributes of data objects may have a usual data type like the first five examples in the following Figure 5 but also special attributes as references and collections (more precisely anchors of collections). This is illustrated in Figure 5 as arrows with one and two heads respectively. Observe that the data object illustration shows both a system generated object identifier (the ID attribute) and a user defined identifier (the Name attribute).

![Figure 5 – Data object with different kinds of attributes](image)

In the analytic view, a data object is usually created as a description of (model of) one or more objects in the real world and the attributes of data objects will most often correspond to properties of the real world objects. Normally, this description has a specific purpose and, hence, a deliberate simplification is made (abstraction), when the data object is created. In contrast, a data object can be created by synthesis and, e.g. as part of a design process, it usually defines something about (model of) a future real world artefact. Data objects have a representation and are captured on a medium, i.e. most often in paper based documents or computer based objects. Although data objects on paper have a physical existence (representation), they must still be considered model objects.

The definition of the object concept must include the more special meanings also.

**Class and Type**

It is also very important to distinguish between individual objects and classes or types of objects. A class or type is a defined as a common description of a set of individual objects (real world objects or data objects). Classes or types, however, belong to the model world and each representation will occur as a data object. The following Figure 6 illustrates at the left hand side the analysis view and at the right hand side the synthesis view.
In order to make a clear distinction in the following between the class concept and the type concept, it is defined that classes relate to the analysis view and that types relate to the synthesis view. More precisely, classes are identified and created from existing objects (may also include data objects) and types are identified and created synthetically as the basis for generating objects (instances), i.e. data objects. Subsequently, such generated data objects may be used as the basis for construction/manufacturing of real world artefacts (realisation). To exemplify: groups of products in a catalogue are considered classes while groups of product models in a library of a modelling software application are considered types.

**Composition**

Another distinction about models is whether they describe individual buildings or they describe buildings in general. Most often, the backbone in descriptions or documentations of individual buildings is the *building structure*, i.e. a *whole-part structure*, where the building is subdivided into components/parts, which again are subdivided into other components/parts etc. down to an appropriate level. This is termed *composition* and such a structure can be formed in many ways depending on the purpose. In different building life phases, the structure and the need for details may be different; so, a suitable description as the basis for construction may not be ideal for e.g. the operations and maintenance. In this phase, for instance, detailed description of many basic components like kernels of foundations, walls and slabs may be of minor importance while more detailed descriptions about coverings and building service components may be of greater importance.

*General regulations* about how descriptions and specifications of buildings should be formed will naturally also include provisions for the structure and the sequence in which description parts should occur. The purpose of creating such regulations should be to standardise building specifications and thereby to make it easier to share and compare building specifications between partners.
Sometimes, such regulations are characterised as classifications but, seen from a theoretical point of view, they are not. Although they are of a general nature and they use names, which may be argued to represent classes, the overall characterisation is that they are about the whole-part structure of buildings. *Classification is something else and more than just identification of classes.*

Figure 7 – Simplified view of model object relationships according to IFC

Compositional structures can be generated on the basis of relationships between model objects (see Figure 7) and it must be remembered that modelling software are supposed to create such relationships. If other kinds of software are used, e.g. traditional database software, relationship can easily be generated based on the geometry. IFC offers two kinds of *decomposition* relationships: *aggregation* of objects of different types and *nesting* of objects of the same type. In addition, the relationship "*contains in spatial structure*" makes it possible to link space objects with building construction objects and vice versa. If this is not enough, other kinds of data can be included by referencing via the *association* relationship – external references.

Because each model object has exact *coordinates*, the exact position in the building or the site is available.

**Classification**

Discussions also indicate that there seems to be multiple interpretations of the fundamental concept *classification*. First of all, there is a difference between 'development of a
classification system (set of classification tables)' and 'the activity to classify'. To classify
requires that a classification table is developed and 'to characterise something', e.g. to
assign a value to an attribute/property is not necessarily the same as 'to classify'.

Classification is an abstraction mechanism by which classes/types can be arranged in a
hierarchy, termed taxonomy. The most general classes/types are at the higher levels (root
levels) and the most special classes/types are at the lower levels. This means that, at any
node, the sub-classes/types must be specialisations of the super-class/types and, in
contrast, any super-class/type is a generalisation of its sub-classes/types. Each sub-
class/type is said to inherit the attributes of the super-class/type and, in addition, each sub-
class/type must have its own attributes. Classification is the foundation for the paradigm
object-orientation, which has a general scope but most extensively has been used in
software development.

Composition, as described above, is also an abstraction mechanism, but clearly the two
abstraction mechanisms are very different. Classification and composition are sometimes
classified as orthogonal to each other. Classification may be very useful in modelling as
the basis for identification and creation of objects and, when objects are created, the
composition structure can be created. In this way, both abstraction mechanisms will be
used in models.

It is necessary to establish an agreement about the purpose of classification, also related a
modern perspective (BIM perspective). The orientation towards working with computer
based models of construction entities leads to new work forms and changes rather
substantially the need for classification. In principle, the basic tasks are two: a) to create
model objects and, from that point in time, b) to detail the existing objects by making
updates on the objects.

For a selected set of objects, multiple classifications can be developed and it is therefore
necessary to select a classification criterion to determine the nodes of the taxonomy. Hence,
different classification criteria result in different taxonomies of the same components. If
each node in the hierarchy can express a class/type according to only one criterion, the
classification is clean and if multiple criteria are used, the classification is mixed. In this
case, only one criterion should be used on each level of the taxonomy. A criterion must be
selected due to a purpose, so not all classifications (included clean classifications) may be
useful or relevant for a selected purpose.