Theme:

Title: **How BcXML Handles Construction Semantics**

Author(s): Reinout van Rees, Frits Tolman, Reza Beheshti

Institution(s): Delft University of Technology, Civil Engineering Informatics
(Reinout van Rees also works for the Stabu Foundation)

E-mail(s): R.vanRees@citg.tudelft.nl, F.Tolman@citg.tudelft.nl, R.Beheshti@citg.tudelft.nl

Abstract: *The paper focuses on the development of a new Communication Technology for the Building-Construction industry, called bcXML. One problem that has to be solved before ICT really will help to increase the BC industry's competitiveness is the way construction semantics should be treated. Over the years several approaches have been researched, but none was able to provide a satisfactory solution. What is needed is a common (neutral) taxonomy with definitions of objects and properties with units and names in different languages and alphabets that can be used to create and translate meaningful XML-based messages, but that is not part of the actual standard. Such an approach would in particular serve the purpose of the fragmented European Building and Construction industry, and is prerequisite for the unification and industrialization of the European Building and Construction industry. The paper describes how eConstruct handled building and construction semantics.*

Keywords: *Building-Construction, XML, bcXML, Communication Technology, eConstruct*

Introduction

The objective of the European eConstruct project is (or better was) to develop, evaluate and demonstrate how the Next Generation Internet can be used to improve meaningful electronic communications in the European Building and Construction industry, supporting future eCommerce and eBusiness. More information on the eConstruct project and bcXML – the Building-Construction specific Communication Technology developed in the project - can be found on http://www.econstruct.org/.

As there have already been several introductions and overviews of the eConstruct project, this paper focuses as much as possible on the question: How does bcXML treat the construction semantics, i.e. the names and definitions of objects and properties of the 100.000 things that play a role in construction?

The nice thing about XML is that it supports the distinction between *content* and *mark-up*. Content is what you want to communicate and mark-up is how you want to present the content on paper and/or on the screen (several mark-ups can be used). For eConstruct this XML feature is interesting because in multi-lingual communication the mechanism can be used to display the content of a message concurrently in different (European) languages or, for instance, together in textual and graphical formats.

Improving electronic communication at large is the ultimate goal of bcXML. However this goal is still far away. In eConstruct we developed bcXML that provides us with the right level of semantics. Suddenly we can talk about Wall Elements, Hollow Concrete Floor Slabs and such and the Internet 'knows' what we are talking about. As a result, computers from now on have at least some understanding of building and construction.

LexiCon: the basis for all bcXML data

The Dutch STABU Foundation is undertaking the building of a taxonomy for the building and construction industry called *the LexiCon*. A lot of this development work was undertaken in the European CONCUR and eConstruct projects. The bcBuildingDefinitions taxonomy developed in eConstruct is a freely available version of the LexiCon build for (and in) the eConstruct project.

For several years ISO TC59/SC13 has been working on the standard ISO 12006 "Organization of information about construction works". This standard now consists of two parts: Part 2: Framework for classification of information, and Part 3: Framework for object-oriented information exchange. Part 2, prepared by WG2, has been sent to the ISO Central Secretariat for formal vote (FDIS), and Part 3, prepared by WG6, is published as a Publicly Available Specification (PAS).

Part 3 describes a formal information model for a framework for a common terminology about objects and their attributes that are of interest for the construction industry. The framework provides for:

- the definition of concepts;
- the definition of relations between concepts;
- the naming of concepts (with multi-lingual capacity).

In the LexiCon, concepts are distinguished into the following main categories:

- *Subjects*: the objects that result from building Activities, including spaces and products
- *Activities*: the activities that result in or modify Subjects
- *Collections*: a grouping of Subjects, Activities or Properties based on a certain criterion, mostly a function or role
- *Properties*: characteristics or features characterizing or qualifying a Subject or an Activity, either directly or indirectly through a Property collection
- *Measures*: the scale used for the value of a Property
- *Units*: the scaling part of a Measure.

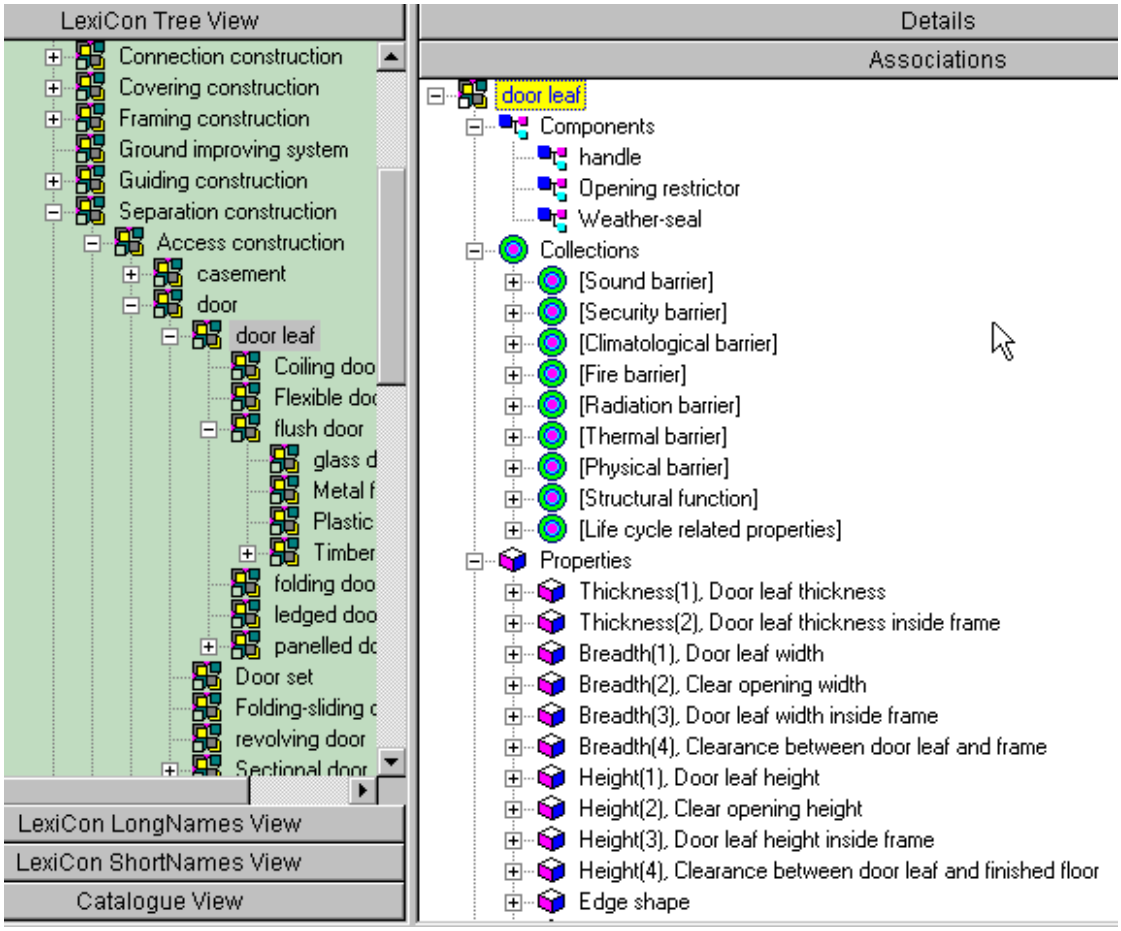

Figure 1: LexiCon hierarchy (left) and information available on "door leaf" (right)

XML taxonomy definition format

One of the results of eConstruct is the XML Taxonomy Definition (XTD). The XTD format is basically a stripped-down version of the LexiCon format. The main reason for a simpler and smaller format was to speed-up of the application development process.

What bcXML is able to communicate in its data is defined in the taxonomy. The XTD is a BC-independent meta-schema used in eConstruct for modelling content-aspects for eCommerce and eBusiness transactions.

This XTD bridges the level of XML Schema Definition (XSD) and the actual industry-sector specific schemas, like bcXML for the description of goods & services between demand and supply in both supply-chains and goods & services development life cycles. It allows for the explicit and semantic definition of objects, properties, units and their interrelationships.

This XTD meta-schema is seen to be complementary to and/or a potential valuable extension for the more transaction-oriented ebXML modelling work. If combined it allows the coherent and consistent development of schemas defining specification messages (demand and supply) within an ebXML envelope.

The XTD format is closely harmonised with the ISO/DPAS 12006-3 effort, which provides us with good interoperability with specification efforts like undertaken in the LexiCon, also allowing us to use the current LexiCon database.

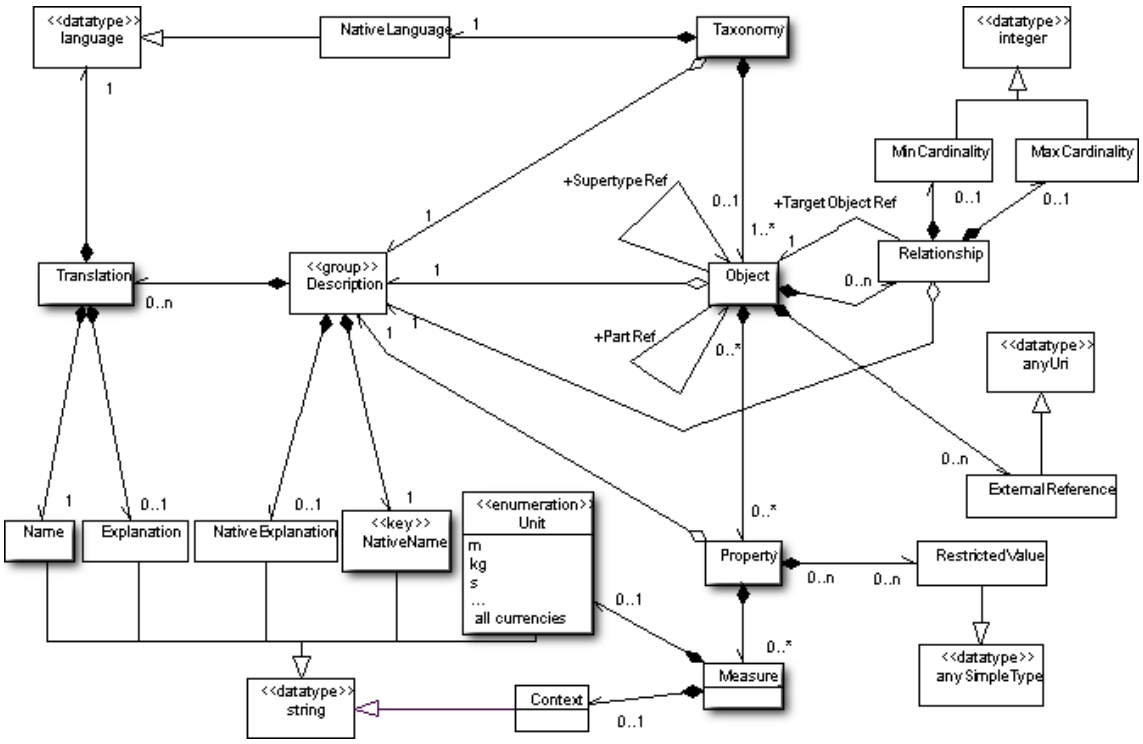

Figure 2: XML Taxonomy Definition (XTD) meta-schema (with most important items highlighted).

Taxonomy Concepts that are supported:

- Multi-lingual data.
- Objects.
- Object specialisation.
- Object decomposition (including maximum number of occurrences for different types).
- (Assigned) Properties.
- Units.
- Restricted Values (i.e. "allowed values").
- General Relationships
- External references.

Taxonomy Concepts not supported:

- No separate object-independent properties. Assigned Properties only.
- No complex (aggregate) properties (atomic properties only).
- No explicit placement of parts in wholes (other than via Properties).

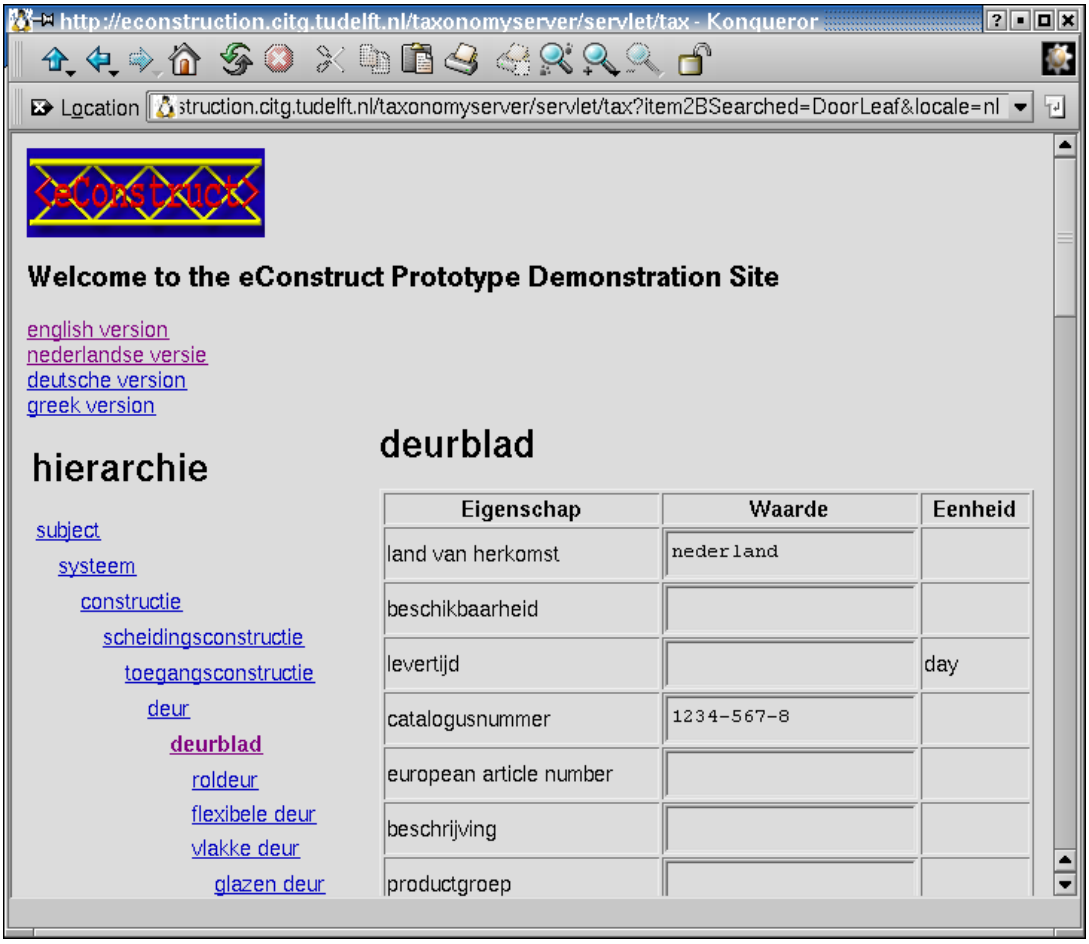

Figure 3: eConstruct taxonomyserver showing the object hierarchy and the "door leaf"-properties in Dutch

From taxonomy to data

The XTD is used to define the objects and their properties. **This is the sole purpose of this data format.** The actual data (requirement specifications, catalogues) is placed in an XML format modelled after the data in the XTD file. That format is purpose-built for easy communication according to the taxonomy in the XTD file. This two-level modelling process integrates well in the multi-level ebXML developments for the transactions.

This is achieved by this four-level method (technical xml format in parentheses):

1. The start is the XTD format definition (.xsd).
2. The taxonomy is expressed in this XTD format (.xml).). Within eConstruct this is called the bcBuildingDefinitions schema.
3. From this XTD-formatted taxonomy a purpose-built format definition is generated (.xsd). Within eConstruct this is called the bcBuildingSpecifications schema.
4. A requirements message or a catalogue containing the actual data is made, expressed in the format generated in step 3 (.xml).

What this means in practice is that on the one hand you've got a taxonomy format dealing with *"an object named door has got a property named fire-resistance"* and on the other hand a data format that allows for a simple and expressive *"a door with a fire-resistance of …."*. So, the taxonomy format and the data format are separate things in the eConstruct universe.

This is not a decision set in stone for follow-up projects to heed, but it makes for a lean, small, readable format without unnecessary complexity with the right amount of expression power to be able to create catalogues and requirement messages according to the definitions in a certain taxonomy. Once eConstruct made the switch to this easy format there was a noticeable increase in implementation speed.

The possibility to mix definitions from multiple taxonomies provides the needed flexibility. It is not difficult to see that only simple XSLT-processing is needed to visualise this nicely to an ordinary end-user. The key-factor for obtaining the required simplicity is the strict division between Definitions (all taxonomy related items) and Specifications by generating a purpose-built schema from the taxonomy data (which is fully automated). Specifiers and catalogue builders often do not even ever need to see the XTD format.

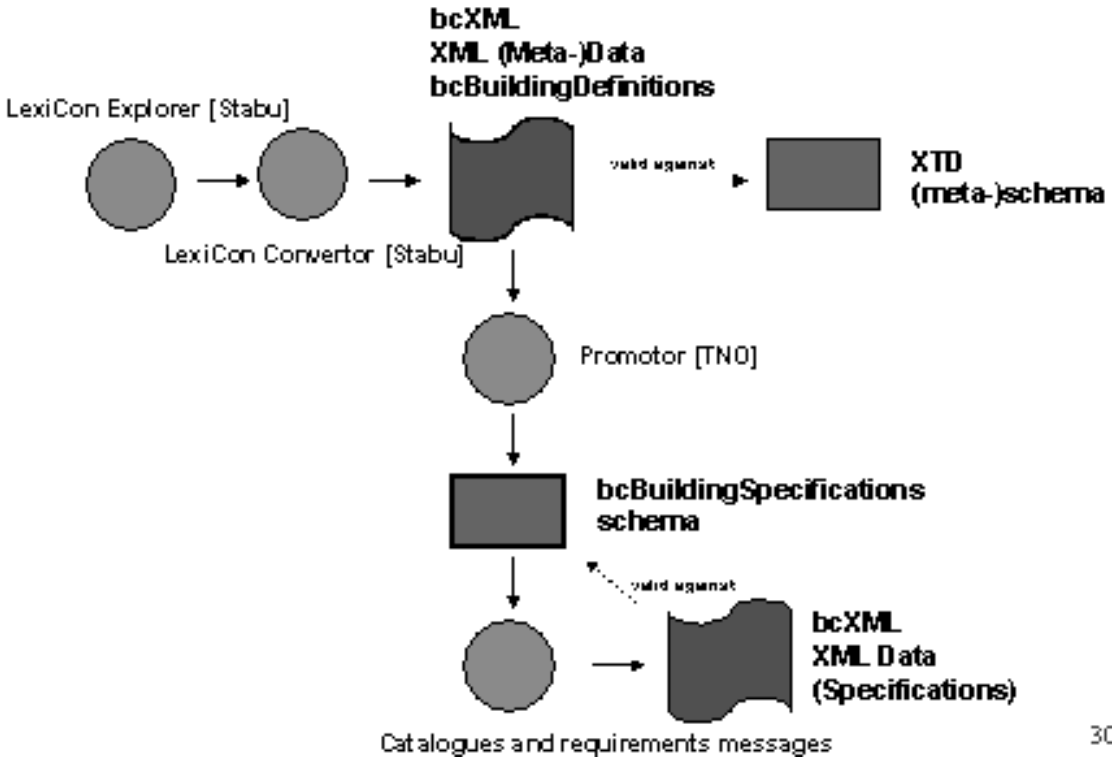

Figure 4: The taxonomy data (level 2) exported from the LexiCon and then promoted to the bcBuildingSpecifications schema (level 3).

To introduce the resulting data, the following two sections deal with generating visual information based on that data, as that is an excellent showcase for the possibilities allowed by this data format.

Context-dependent visualisation of components

Figure 5 shows the complete process. It begins with a bcXML data file. A stylesheet (using information retrieved from the taxonomy) adds multilingual information to the file (Norwegian information in this example).In the next stage, this data is transformed to SVG 2D-graphical using a stylesheet containing the visualisation information. For both the multi-lingual and the SVG data, the visualisation is shown in the figure.

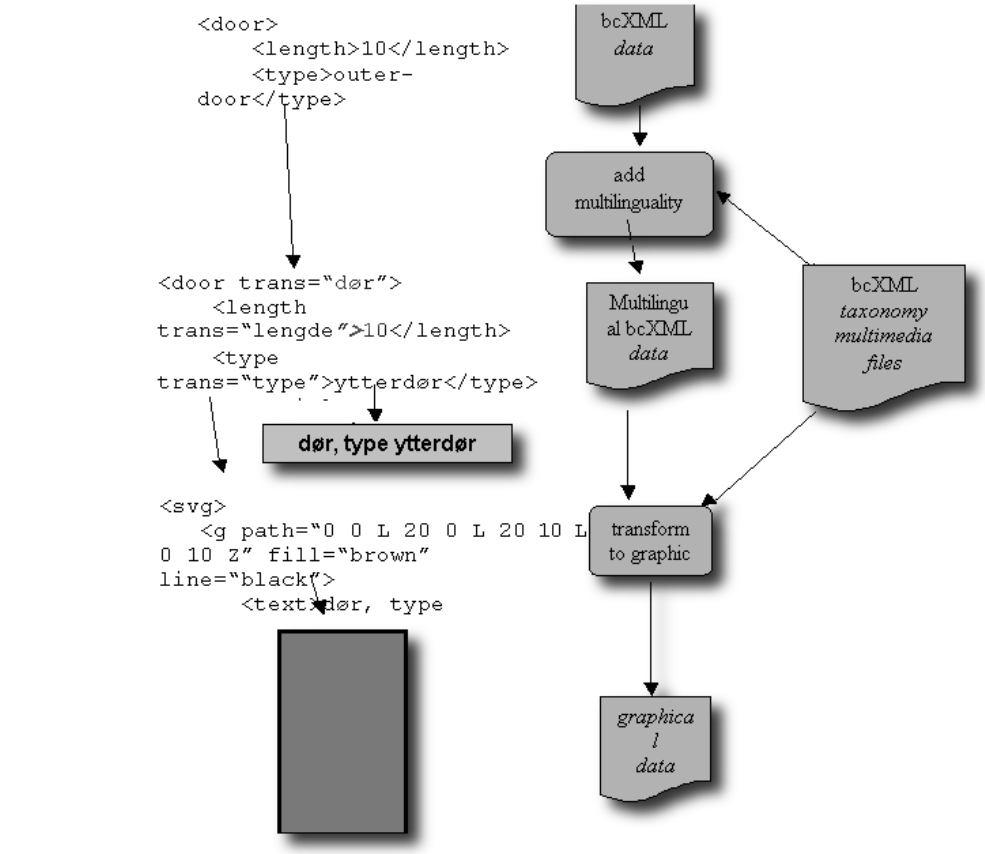

Figure 5: two ways of visualising the same bcXML data

The next two screenshots (Figure 6) further illustrate the use of contexts. A context can be, for instance, a certain language or a professional field. Shown are a Greek textual visualisation and a 2D image with some extra information that only pops up when you click on the door. Information hiding in order to remove clutter. You can choose to receive materials information, size information, information about the price, etc.

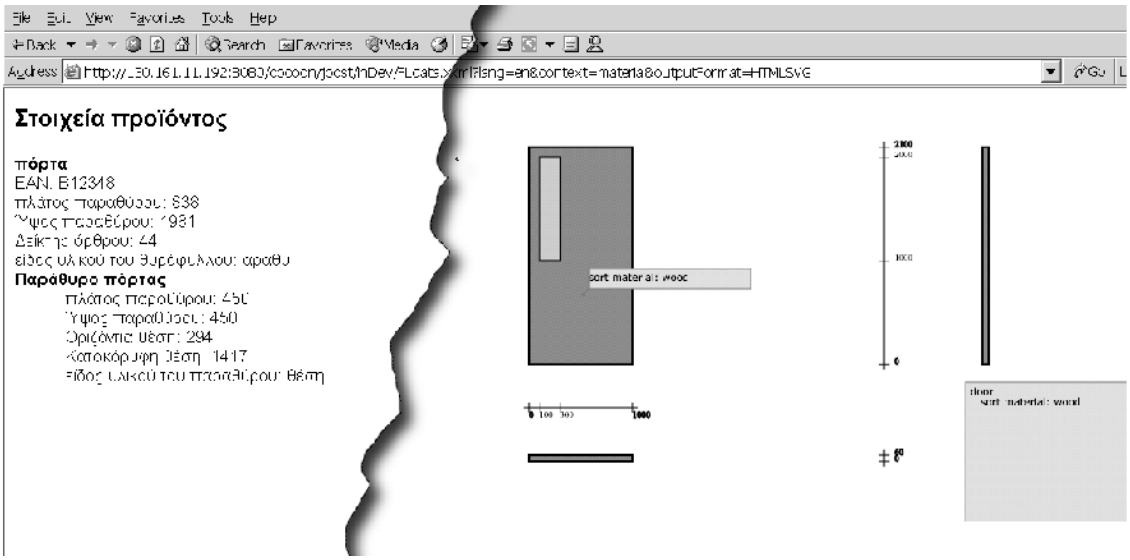

Figure 6: Greek textual information (left) and pop-up window with extra information (right)

From Components to Projects

The difference between a *component* visualisation and a *project* visualisation is that a component visualisation visualises just one item at the time, while a project visualisation displays multiple objects together, in their relationships.

What-if all parties involved can have access to a virtual world describing the project in detail? *Some detail though, not great detail, because the VR project world is used to support general information retrieval not detailed shape. Real CAD data is IFC's terrain!* Abstracted shapes and textures are in most cases sufficient and topology (i.e. how a beam is exactly connected to a column) can be hidden from most users.

The component VR information front-end described above is particularly suited for communications about specific types of components (no instances). One usually does not buy a particular door, but a particular *type* of door. For project information, however, more occurrence information is required.

Starting point for the development of a general solution was the idea of Tolman [Tolman et al. 2002] that the world can be viewed as a collection of four groups of real world objects that can be roughly grouped according to their geometry:

- point-like objects (e.g. ElectricityOutlet, WheelBarrow)
- line-like objects, (e.g. Column, Beam, AccessRoad)
- face-like objects and (e.g. Wall, Roof, SiteArea, Window)
- volume-like objects (e.g. Building, FloorSpace)

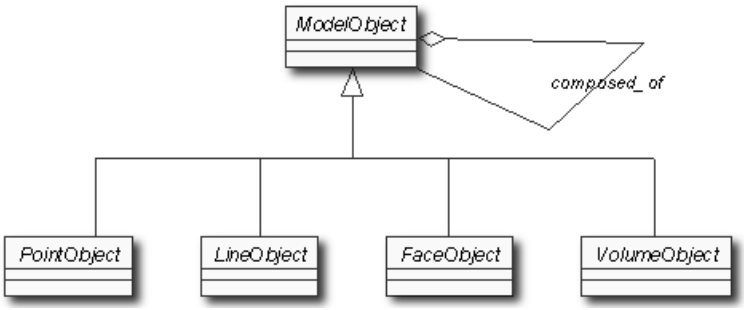

Figure 7: Simple UML model classifying objects

As shown in figure 7, PointObject, LineObject, FaceObject and VolumeObject are *abstract* classes. The basic idea is that all real world objects can be classified according to these four object types and thus can be defined as subtypes of these four *semantical* entities.

For example the class Wall inherits from FaceObject, the class Road inherits from LineObject, and the class Building inherits from VolumeObject. Note that the four basic objects are *not* shape objects but very abstracted 'things' that you can see in and around a construction project and thus also objects that can be entered into the bcTaxonomy and treated accordingly.

To support the communication of above information, a small taxonomy has been made containing those objects and properties. The mixing of the "regular" bcbuildingdefinitions taxonomy used by eConstruct and this small taxonomy allows us to add shape information to a bcXML file:

```
<Door>
  <height unit="m" value="2.10"/>
  <width unit="m" value="0.80"/>
  <Face xtdReuse="ForProperties">
    <x value="3.20"/>
    <y value="6.00"/>
  </Face>
</Door>
```

The *xtdReuse="ForProperties"* indicates that the *Face* isn't a component, but is only used to include it's properties (this is a bcXML mechanism to deal with multiple taxonomies).

The example shown here is a picture of a particular wall with a door, a window, a light switch and an electricity outlet. The window and door etc. are components that together form a part of a building. The bcXML file for this wall in the example and the formatting process are illustrated below.

As a first start, we take an almost empty bcXML file containing just “wall”, but fleshed out with more information. Using “external” data from the small taxonomy introduced in the beginning of this chapter, an x-length and y-length (width and height) are added to the wall: Width and height are missing from the taxonomy because eConstruct concentrated itself on door data.

```
<?xml version="1.0" encoding="UTF-8"?>
<bcBuildingSpecifications xmlns="http://www.bcXML.org/2002/bcXML">
  <ItemSet>
    <Wall>
      <External xmlns="http://www.tudelft.nl/shapemodel">
           <Face xtdReuse="ForProperties">
             <x-length>
               <SingleValue>6.0</SingleValue>
             </x-length>
             <y-length>
               <SingleValue>3.0</SingleValue>
             </y-length>
           </Face>
      </External>
    </Wall>
  </ItemSet>
</bcBuildingSpecifications>
```

As a last code example, an object is added to the “wall”, namely a “door leaf”. Putting such an object (technically) within the xml-tag “<wall>” means that the “door leaf” is a component (or part) of the “wall”. Normally it would be visualised in the lower left-hand corner of the wall, but a transformation puts the door leaf 4 meters more to the right:

```
<?xml version="1.0" encoding="UTF-8"?>
<bcBuildingSpecifications xmlns="http://www.bcXML.org/2002/bcXML">
  <ItemSet>
    <Wall>
      <External xmlns="http://www.tudelft.nl/shapemodel">
           […  same as above …]
      </External>
      <DoorLeaf>
           <doorLeafThickness>
             <SingleValue>0.05</SingleValue>
           </doorLeafThickness>
           <doorLeafWidth>
             <SingleValue>0.80</SingleValue>
           </doorLeafWidth>
           <doorLeafHeight>
             <SingleValue>2.20</SingleValue>
           </doorLeafHeight>
      </DoorLeaf>
    </Wall>
  </ItemSet>
</bcBuildingSpecifications>
```

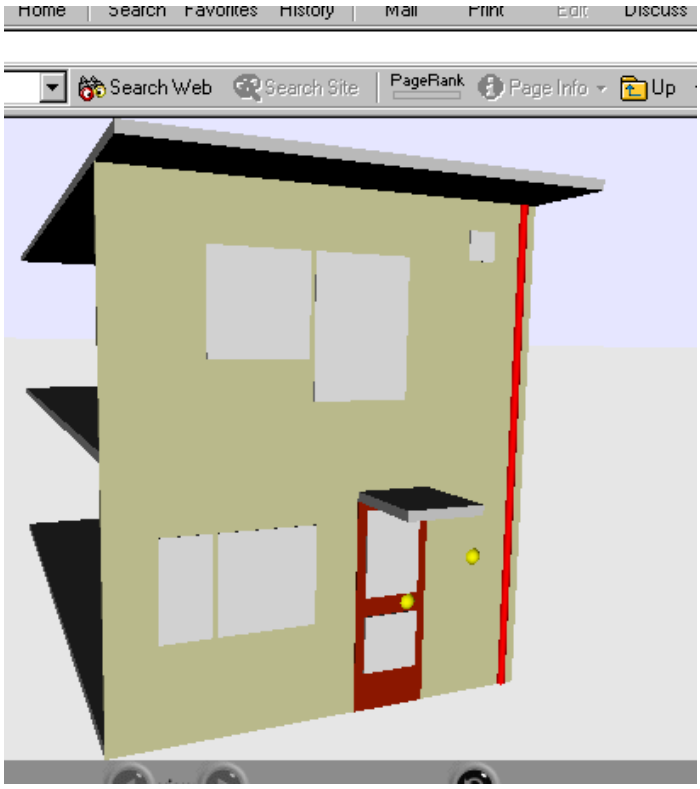

Figure 8: Above example data further expanded (modelled after an existing Dutch house)

This data file still retains all possibilities inherent to bcXML, like multilingual textual information and also other visualisations, for instance to 2D.

The fact that bcXML can also be used in project related communications is very important. Quite often small teams of designers and engineers are working on the design of details like floors with openings for various systems.

Discussion and recommendation

1. BcXML and related components can probably provide a sound basis for communication improvement for large-scale Building and Construction projects at European level.
2. BcXML is probably able to provide the European BC industry with the required low-cost but powerful communication technology, provided however that the work on BC taxonomies will be continued. It is strongly recommended that a number of bcXML compliant taxonomies (for Building and Civil Engineering), each containing a basic set of object definitions, be made available in all the languages and alphabets of the current and future member states.
3. The European Commission can take a closer look at the eConstruct results and try to get consensus about the above recommendation as this recommendation calls for a new type of EU-project involving governmental representatives of each of the (future) member states. As bcXML based electronic communication is a prerequisite to the unification of the BC industry, it is recommended that the Commission organising such a project (Currently an initiative is underway under the CEN umbrella, also the LexiCon model and the XTD are now being harmonised into one model.)
4. We estimate that 50 man-year is needed to provide the LexiCon with the content needed to serve the complete building and construction industry, a task that definitely requires full support of the European Commission as well as the local governments.

Acknowledgements

Further dissemination of eConstruct (IST 1999-10303) results is carried out within the cluster project ICCI (2001-33022). http://cic.vtt.fi/projects/icci/