Use cases for configurable building information model visualization

Helga Tauscher¹, Raimar J. Scherer²
¹,² TU Dresden, Institute for construction informatics
¹,² http://tu-dresden.de/bau/cib
¹,² {helga.tauscher|raimar.scherer}@tu-dresden.de

We present a new approach to reconnect building information models and their visual representations involving architects and engineers as domain experts in the process of the visualization specification. To this end a generic visualization framework and a visualization description language are introduced. Four use case areas for the application of the approach were identified and are illustrated with corresponding examples.

Keywords: BIM, visualization, DSL

CONTEXT AND OBJECTIVE
Due to the advances in building information modelling (BIM), the information itself and its visual representation are falling apart. Information and its visual representation where implicitly contained in a cohesive form in drawings and documents before, however now information is modelled explicitly and thus separated from its visual representation. As the visual representation is factored out of the information models it has to be generated on the fly. The exact arrangement, definition and design of this process are coded into specific software applications and are rarely accessible to the user.

However domain expert users of architectural and civil engineering applications are not only experts regarding their tasks, but also regarding the information and the visual representations needed for these tasks. Historically architects and engineers have developed a repository of specialized visual representations, visual signs and symbols and of methods to encode their content visually. Creating visual representations is not only a means to an end but an essential way of engaging with their subject and material.

Thus it is necessary to open up options to architects and engineers to work with the visual representations, to shape, configure and devise them, instead of just consuming preconfigured visual representations. The role of visualizations in the context of BIM was pointed out by several authors before, e.g. Liebich (1993), Katranuschkov (2000), Wender (2009). However, there is no approach tackling the problem in a way involving architects and engineers as participants in the process.

USE CASE AREAS FOR A GENERIC VISUALIZATION FRAMEWORK

Visualization framework
Towards the goal to open up more possibilities to work with visual representations for domain expert users the authors pursue the objective of a generic visualization description and framework. Instead of producing preconfigured visualizations hard-coded in the application, the visualization framework is capable of constructing arbitrary visualizations. The
configuration of the desired visualization to be constructed is defined by means of a visualization description, which is fed into the framework in addition to the BIM data in question (Figure 1).

The technical details of the visualization framework are outside of the scope of this contribution. We are referring to a first discussion in Tauscher & Scherer 2012. In this paper we will give a short introduction to a possible domain specific language (DSL) for the visualization description, which also outlines the scope of the framework. We will then define four use case areas, where visualization specifications might be useful. For each use case areas we constructed an application examples. These In section 3 the DSL is used to present these application examples and demonstrate the functionality of the framework in context.

**Visualization specification**

The DSL we are using here is only one possible way to specify the visualizations to be generated by the framework. It contains the following lexical elements:

- **rule**: A mapping rule defines the relation between elements of the data and elements of the visualization. The visualization elements are specified in terms of their type and the values of their properties. In order to set these properties the mapping rule can access properties of the related data element.

- **condition**: Conditions define the scope of mapping rules. They act as filters, narrowing the input data to a subset, which the respective mapping rule has to be applied to. They are specified in terms of the type of the data elements the rule applies to and in terms of additional conditions the data objects have to fulfil.

- **initial/update**: Mapping rules may have two different roles: On the one hand (in the "initial" role) they define, how to create the visualization elements. On the other hand (in the "update" role) they may define how to change already created objects. Changes may be triggered by events or by the advance of time.

- **graph/data**: These keywords are used to access the properties of data and visualization elements inside of the mapping rule. During the mapping rule application, the respective data and visualization elements are made available under these names.

- **space/part**: The visualization space may be subdivided into spatial parts. The "part" keyword is used to group and relate rules to the respective spatial parts.
In terms of further syntactical rules, we are using whitespace indentation as known from Python instead of brackets to identify blocks belonging together. This produces compact code. Further we are using common mathematical operators for arithmetic and equality operations as well as curly braces for anonymous functions. These functions can be applied to each element in a set using the *map* keyword.

The framework also provides statistical functions aggregating values over sets of data elements and the possibility to calculate and specify global values outside of the local scope of certain rules. We are not using these possibilities in the exemplary DSL nor in the examples. Instead we are strictly limiting the calculations to the values reachable from the current data object. Fixed global configuration values are indicated by using capital letters.

To further compress and simplify the DSL, keywords may be left out under certain conditions. The *rule* keyword may be left out if there is only one single rule, such that there is no need to mark and enumerate multiple rules. Similarly the *initial* keyword may be left out if there are no updates and no conditions to differentiate from the mapping itself.

**Use case Areas**

Certainly useful visualizations are tied to specific use cases, and it could be argued that for this reason the generation of visual representations belongs exactly where it is - inside of specialized applications. However, we have identified four areas, where it seems to be especially helpful to grant further options to control, specify and manipulate visual representations, hence to provide extended access to visualization methods to engineers and architects as domain experts. These areas are: communication, education, exploration and experimentation.

*Communication:* A substantial part of the communication happening in architectural and construction projects is carried out using non-textual media or textual media with references to non-textual parts. With the widespread use of BIM these non-textual parts are replaced by the exchange of models, such that a common view of the subject is not guaranteed anymore. The exchange of visualization specifications together with the models could fix this.

*Education:* When teaching and learning BIM principles, a visual representation of the otherwise intangible and abstract concepts is especially important. Classes, objects, properties and relations in a model are hard to grasp. Visual representations on the other hand are more accessible to the human mind and understanding how the abstract concepts map to visual representations helps to grasp the abstract concepts.

*Exploration:* In the process of planning and execution of construction projects different stakeholders often rely on preliminary work of other project partners. Thus it often becomes necessary to review incoming building information models, to get a general idea of the content of a building information model or to track changes in particular parts of the models. These activities may be supported by customized views on the model, which can be reused later on.

*Experimentation:* During the development of new analytical or numerical methods, for instance to assess risk during construction, to validate schedules or to shape the outer shell of a building, there are neither the software tools nor established visual representations available for exactly this task. A framework to generate the necessary visualizations would support the creative work in these cases.

These use cases have commonalities as well as substantial differences regarding the objective and goal. For communication purposes it is relevant to present the result of an analysis, no matter whether the analysis was carried out automatically or manually. On the other end of the spectrum in the case of exploration the result is not achieved beforehand, but the visualization serves the process of gaining insight in the underlying information. The educational use case combines both uses of visualization: when teachers demonstrate facts about the subject, they are using visualization in the communicational way, while learners trying to understand the subject use
visualizations in the exploratory way. The education use case differs from the communication and exploration use cases in the complexity of the information and the resulting visualization: education is usually carried out with simpler material, while in the other cases visualizations unfold their full potential when applied to more complex material. Experimental visualizations may aim at all three previous application areas. They only differ in the fact, that they use a new or unusual way of encoding visual information.

APPLICATION EXAMPLES FOR THE USE CASE AREAS
To demonstrate how users would benefit from the possibility to specify a visualization description, this section will introduce four example visualizations and present them together with their visualization description. The general methods and the exemplary model employed for the examples are similar, but the utilization and focus of each example is different. We are assigning the examples to the use cases, although this assignment is not exclusive. Each example could also occur in the context of one of the other use cases. The assignment between each use case and a typical example is made solely in order to illustrate the different application areas of the proposed visualization framework and description language.

Communication
This example has a clear communicative character, it states a message and supports the dialog about a subject. A solution fulfilling basic communication requirements is already specified in the BIM Collaboration Format (BCF). BCF is a lean format, intended to be used for the exchange of information during the communication referring to a building model or parts of it. The specification contains a so called "viewpoint" section allowing for the specification of camera parameters and the designation of one or more objects of interest, intended to highlight them in the visualization. The example presented here adds a simple colour scale to this setup.

The underlying model features a small construction project - a single family house with two storeys and basement. An exemplary realistic use case is the communication about a delay and an underlying obstruction. A load-bearing wall on the ground floor is not finished in time, such that work on the slab above ground level could not be finished in time as well and the construction of walls in the next level is obstructed. The visualization helps to track back the reasons for the delay and to facilitate communication about the issue.

The following code defines the visualization. Figure 2 shows the resulting visualization.

```plaintext
view:
  projection: isometric
  direction: (-1,-1,-1)
rule: ResLink<IfcObject> > Polyeder
condition:
  data.key.obj type IfcBuildingElement
initial:
  graph.vertices = data.key.geo.
  graph.normals = data.key.geo.normals
  graph.indizes = data.key.geo.indizes
  rep = data.link.report
  sch = data.link.schedule
  graph.color = scale(rgba)
  rep.date < sch.start: (0,255,0,150)
  rep.date > sch.end:     (255,0,0,150)
  else:                     (255,255,0,150)
rule: ResLink<IfcObject> > Polyeder
condition:
```
First, we define an initial camera view by specifying the projection mode (isometric) and the camera direction (view from top-north-east to the origin). The zoom property of the view defaults to "scale to extend" and does not have to be specified. In general, the visualization frameworks provide sensible defaults wherever possible to allow the user to concentrate on the essential parts of the specification. The remaining examples leave out view definition sections for their 3D parts completely, defaulting to a perspective projection.

Then a mapping rule set is defined, consisting of two rules. The first rule applies to all objects of type IfcBuildingElement and creates transparent 3D objects with a color scale according to whether the reported construction progress confirms to the schedule or not. Every object reported to be completed before the planned period has even started is colored green. Objects completed during the scheduled period are colored yellow and those completed with significant delay are colored red. Note that depending on the granularity of the underlying schedule, objects with delay may also receive yellow color.

The second rule applies under the condition that the object is of type IfcBuildingElement and in addition it has a specified unique ID. This rule is a partial definition only, because it only sets the color value to opaque blue without defining any geometry. Because the rule applies to a subset of the objects matched by the first rule, the geometry definition does not have to be repeated. Instead only the color value from the first rule is overridden.

The third rule again overrides the color value of the first rule, rendering the object in question invisible if it would otherwise hide the object in focus (highlighted by the second rule). The geometric calculations necessary to check for this spatial constraint could be coded directly in the condition, but they would quickly exhaust or exceed the computing resources as the size of the model grows. This is why the framework allows for the inclusion of external filter libraries to query the model efficiently. The specification refers to a fictional filter library called "spatialFilter". Provided that an efficient implementation existed, the runtime system could even update the visualization on view direction changes.

**Education**

This example is at the simple end of the spectrum, it should emphasize or confirm something already known about the information in a visual way. We are presenting an example with a more sophisticated color scale and an interaction component to engage with the visualization and explore details of the color scale.

The example used is a very simple building, constructed artificially for educational purposes. It consists solely of five building elements, four columns with two different geometric profiles and a roof. The example is designed with this high level of simplicity in order to teach the fundamentals of Industry Foundation Classes (IFC). Students are able to work with the STEP file in a text editor in parallel to a visual representation, without being overwhelmed. This way they are enabled to establish a clear relationship between the textual representation of the model and the concepts encoded in the model.

The following code defines the visualization. Figure 3 shows the result of the mapping process.
The visualization consists of a 3D view with default view settings as stated in explanation of the previous example. The 3D view is embedded in a small application featuring a text entry field for filter queries. Depending on the setting of the checkbox labelled "(Un)highlight" either an HIGHLIGHT or an UNHIGHLIGHT event is triggered from the outside into the visualization.

Similar to the communication example, a first rule maps the geometric properties of the data to the shape of the visualization object and sets a default color. Two further rules override the color value. Unlike before, these rules are not applied during the initial mapping pass, but instead they are applied during runtime triggered by events named HIGHLIGHT and UNHIGHLIGHT. This kind of deferred change of objects in the visualization allows for dynamic highlighting of multi object selections.

The visualization specification can be refined by adding a color scale that differentiates the type of the object.

```python
graph.color = rgba(128, 128, 128, 150)
```

This scale ranges from yellow over green and cyan to a greenish blue and could be extended for further types. Then, during unhighlighting the original color of the object has to be restored according to the same scale.

The example in the next section shows how two views can be interlinked. The educational example could benefit from a similar setup, by adding and connecting a text representation of the STEP file and highlighting respective lines upon object selection. Alternatively the display of the GUID and/or line number could facilitate lookup of the line in a text viewer or editor.

**Exploration**

This example is more ambiguous, the visual representation does not intentionally and clearly communicate a result or message, but rather encourages the examination of the information. We are presenting a visualization combining different views of the information contained in a complex data set interactively.

The underlying data follows the multi model paradigm - a method to couple multiple elementary, previously unconnected models (Fuchs 2011). By adding additional relations, multi models substantially increase the complexity of the information contained in the models, allowing for more advanced methods to analyze the data, to gain insights into and in general to work with the information, but at the same time increasing the difficulty to intuitively understand the data.

The concrete multi model in this example consists of two elementary models: one model contains 3D object information according to the IFC standard, and the other one contains a work specification with description of work and cost information conforming to the GAEB standard. The communication example in section 3.2 did already use multi model information with an IFC model and schedule informa-
The former example accessed the multi model information grouped by IFC model elements, treating schedule information as an attachment to these key elements. The current example on the contrary accesses the multi model on link level and groups the links twice - once according to the IFC model and once according to the GAEB model.

The following code produces the visualization in figure 4.

```plaintext
part: IfcObject > Polyeder
collection:
data.obj type IfcBuildingElement
initial:
    graph.vertices = data.geometry.
    -> vertices
    graph.normals = data.geometry.
    -> normals
    graph.indizes = data.geometry.
    -> indizes
    graph.color = rgba(128,128,128,150)
    graph.click = HIGHLIGHT
update(HIGHLIGHT):
    graph.color = rgba(200,0,0,0)
update(UNHIGHLIGHT):
    graph.color = rgba(128,128,128,150)
part:
rule: GaebTgItem > Rectangle
initial:
    graph.height = 15
    graph.left = 400
    graph.top = index*20
    graph.width = data.IT
    graph.click = HIGHLIGHT
rule: GaebTgItem > Label
initial:
    graph.text = data.outlineText
    graph.left = 0
    graph.top = index*20
```

The visualization could be extended by highlighting only parts of the bars in the chart - only that part that corresponds to the costs of the selected objects. Accordingly also the highlighting of 3D objects in reaction to selections in the bar chart could be scaled in color strength or brightness to indicate the proportion of costs of that single building element selected in the bar chart.

**Experimentation**

The final example uses hierarchical edge bundles (HEBs), a novel visualization method from outside of the architectural or construction domain and verifies its applicability for domain specific information and tasks.

This example is very similar to the one in example 2, because the experimental study is also targeting an exploratory visualization against the background of the steadily increasing amount and complexity of information.
The following code produces the visualization in figure 5.

```javascript
space:
  part: IfcObject > Polyeder
    ...
  part: ResolvedLink > Bezier
    tree = { n ->
      y: (n.before + n.size/2) * SCALE
      x: n.depth * DIST + OFFSET
    }
    BS = 1-BUNDLING
    params = { t ->
      ax: BS*(t.first.x-t.last.x)/t.size,
      ay: BS*(t.last.y-t.first.y)/t.size,
      cx: BS*t.first.x,
      cy: BS*t.first.y
    }
    bundling = { node ->
      x: BUNDLING*node.x + cx + i*ax
      y: BUNDLING*node.y + cy + i*ay
    }
    tr = data.hierarchicalIfc.anchestor.
    map(tree)
    graph.points =
    tr.map(bundling, params(tr))
    tr = data.hierarchicalGaeb.anchestor
    .inverse.map(tree)
    graph.points =
    tr.apply(bundling, params(tr))
    graph.color = rgba(50,50,50,0)
  part:
rule: GaebTgItem > Rectangle
  ...
rule: GaebTgItem > Label
  ...
```

Again, similar to the previous example, space separation with the default layout algorithm is used. This time three different areas are constructed. The left area and the right area are similar to the ones in the previous example, 3D objects representing building element geometry on the left side and a bar chart representing cost values on the right. The mappings for these two areas are marked with ellipsis and are not specified in detail again.

The third mapping for the central area is different and contains the experimental application of hierarchical edge bundles. Each link connecting a building element with a certain work specification is represented by a bezier curve. These curves are bundled according to the algorithm proposed by Holton (2006), yielding a visual representation where links in the same part of the building structure or in the same part of the work specification structure are lying closer to each other than those belonging to different structures.

Details regarding the application of the original HEB algorithm to the construction domain can be found in Tauscher & Scherer (2013). In that paper we have also proposed several extensions to the exam-
ple visualization using dynamic color scales or highlighting according to the linked proportions selected in another part of the visualization.

As in the previous example objects in all parts of the visualization may trigger as well as receive highlighting events. This part of the specification is left out in the listing above.

CONCLUSION AND DISCUSSION

We have introduced four use case areas for the application of a new approach to reconnect building information and its visual representation by specifying the visualization generation explicitly. For each use case area a concrete example was shown, with a characteristic visualization and the corresponding visualization specification. Apart from the main contribution of this paper - the identification, classification and illustration of use cases for the approach - the examples where analysed regarding further development and further requirements for the framework and DSL.

Each example can be further developed and extended: The communication example can benefit from the integration of an efficient spatial filter library and the addition of runtime evaluation according to view properties. The education example can be extended to contain more explorative elements with embedded textual GUID annotations or an additional text representation of the whole model. Dynamic color scales and refined highlighting could make the exploration example even more useful. Also the experimentation example would benefit from dynamic color scales for the curve bundles.

Some of the extensions can be achieved using the possibilities already present in the current framework concept, but other extensions pose extra requirements on the framework and DSL: Filter conditions are currently evaluated outside of the framework, the results are then used to fire events in the framework. Dynamic query evaluation would allow for queries to be evaluated during runtime inside the framework. Update actions could be further parametrized to evaluate external code beyond querying, e.g. for dynamic color scale application. View parameters could be made available during runtime for dynamic queries and scales.

Further, the following problems where excluded from the example specifications, but have to be solved in order to reveal the potential of the approach in practical applications: The DSL should allow for the integrated specification of data access using external filter and query languages. This includes data preprocessing such as aggregation or grouping and consistent access across the single parts in multi part visualizations allowing to dispatch events between these parts. Reusable and modular visualization specifications should be facilitated through an include mechanism. The language constructs of the DSL have to be chosen carefully, taking the potential previous programming language knowledge of architects and engineers into account. The advantages and disadvantages of a standalone DSL tailored to the user group versus an embedded DSL with a more general allowance of language constructs from the host language have to be weighted.

ACKNOWLEDGEMENT

The building models where partly developed during the BMBF projects mefisto and eworkBau. The funding of the Federal Ministry of Education and Research is gratefully acknowledged.

REFERENCES

Fuchs, S, Kadolsky, M and Scherer, RJ 2011 'Formal Description of a Generic Multi-Model; WETICE - 20th International Conference on Collaboration Technologies and Infrastructures, Paris, France


Tauscher, H and Scherer, RJ 2012 'Towards a configurable nD-viewer for building information models.
A generic model for the description of visualization methods, Proc. 9th European Conference on Product and Process Modelling (ECPPM), Reykjavik, Iceland

Tauscher, H and Scherer, RJ 2013 ‘Utilizing hierarchical edge bundles in multi model visualization,’ eg-ice 2013. 20th International Workshop: Intelligent Computing in Engineering, Vienna, Austria