www.itcon.org - Journal of Information Technology in Construction - ISSN 1874-4753

SPECIFYING COMPLEX VISUALIZATION CONFIGURATIONS WITH HIERARCHICALLY NESTED MAPPING RULE SETS

SUBMITTED: November 2014
REVISED: December 2014
PUBLISHED: January 2015 at http://www.itcon.org/2015/3
GUEST EDITORS: Mahdavi A. & Martens B.

Helga Tauscher
TU Dresden, Institute of Construction Informatics;
helga.tauscher@tu-dresden.de

Raimar J. Scherer
TU Dresden, Institute of Construction Informatics;
raimar.scherer@tu-dresden.de

SUMMARY*: A generic visualization framework should allow for the specification of arbitrary visualizations to be generated from Building Information Models. It has been shown before, how simple visualizations can be produced with mapping rule sets and how two simple visualizations can be combined into a more complex visualization using three different combination methods. Now these approaches are extended to arbitrary complex visualizations by nesting the combination methods hierarchically. Systematic analysis yields nine different nesting cases, which are evaluated and illustrated by use case examples.*

KEYWORDS*:* BIM, visualization, model mapping, hierarchical nesting

REFERENCE: *Helga Tauscher, Raimar J. Scherer (2015). Specifying complex visualization configurations with hierarchically nested mapping rule sets. Journal of Information Technology in Construction (ITcon), Special Issue: ECPPM 2014, Vol. 20, pg. 40-50, http://www.itcon.org/2015/3*

1. INTRODUCTION

1.1 Building information modelling and its implications for visual representations

Building information modelling, the semantically explicite modelling of information related to construction projects, is primarily targeted towards the interoperability of computers. To achieve defined semantics, information was factored out of its previous close connection with its visual representation during the development of BIM.

But visual representations have to be generated reproducibly and reliably to fulfil their communicational means. Furthermore, the growing amount of information contained in BIM requires new methods of visual representation, beyond traditional visualization methods. Domain experts should be involved in the creation of visualizations, but do currently not have the possibility due to technical obstacles.

In parallel to the development of BIM, new media have opened up new perspectives for visual representations. Also, the efficient and repeated creation of multiple visual representations from the same information is only made possible by the separation of information and visual representation.

1.2 Previous work on presentational issues in CAD and BIM

Previous research on presentational issues with BIM dates back to the application of digital media technologies to pre-BIM CAD data. Visual presentation has always been of special relevance in the design related parts of the construction field, therefor applications for the presentation of architectural design are one research focus, such as the work of Abdelhameed (2005) or Balakrishnan et al. (2006).

During the development of BIM the role and potential of visual representations where recognized and acknowledged. Liebich (1993) for instance introduces the concept of "monitors" to encapsulate certain visualization modes. Eastman et al. (2011) lists information visualization beyond 4D as one of the trends in BIM.

However, visualization is rarely treated as a distinct topic in the context of BIM, but appears only as a side issue in other research or focusses on specific concrete task specific visualizations, such as visualization of the information flow in construction projects (Otjacques et al. 2006) or colour schemes for 4D construction progress visualization (Chang et al. 2009).

1.3 Visualization specifications for BIM - a generic approach

As opposed to the development and improvement of concrete task-specific visualization techniques the authors pursue the topic on a meta level, by investigating the relation between the information model and its manifold visual representations conceptually. The goal of this approach is a model of the visualization process, which can be instantiated to represent different concrete visualization techniques. A description of the intended visualization serves as configuration for the generic model.

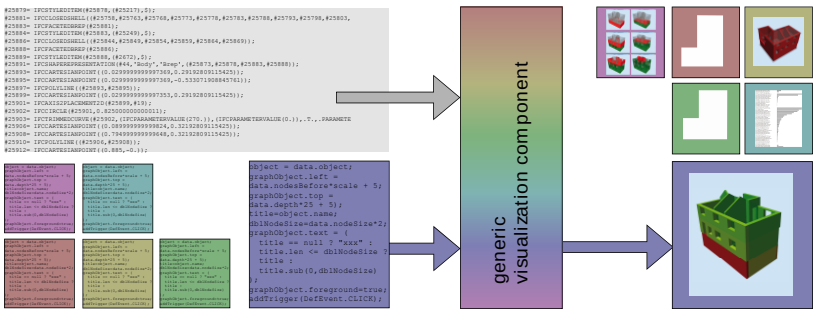


FIG. 1: Generic visualization component

An implementation as a generic framework for BIM visualization servers first of all as a proof-of-concept prototype, and second as the base for tool support, in order to foster the study of specific visualization issues. The framework generates visual representations not in a hardcoded way, but from a formal visualization description in addition to the building information (Fig. 1). In the process the visualization description or specification serves as configuration for the generic visualization component. The current implementation of the

framework is already able to generate simple visualization models from domain models using mapping rules (Tauscher & Scherer 2012).

1.4 Question and working hypothesis: a structure for complex visualization specifications

Our first approach towards this vision identified the general architecture of a respective visualization mapping framework based on the visualization pipeline (Haber & McNabb 1979), which describes the visualization process as a transformation from raw input data to the final image. The transformation is carried out in multiple steps - filter, map and render - creating intermediate transformation results from the input data in each step.

The visualization specification configures each of these steps: Which subset of the information model to select for visual representation, how to map the data to visualization objects and finally how to present the visualization model on the display area. Mapping rules can capture the specifics of the central mapping step for a certain visualization, and implicitly also for the first step, selecting the input data for mapping rule application.

As the amount of explicitly modelled information is constantly growing - for instance by combining multiple models into informational richer multi model assemblies (Fuchs et al. 2011), sophisticated and complex visualization methods are needed. Therefor the visualization description as well as the framework should support arbitrary complex visualization setups, which exceed the limitations of simple mapping rules. We have to answer the question: Which structure for visualization specifications is necessary to allow for the generation of every desired visual representation?

We hypothesize, that this can be achieved using a recursive approach, where simple combination strategies are applied repeatedly to create complex visualizations from simpler mapping rule sets. In order to validate this, we first propose a set of combination strategies for visualizations. We then analyse a couple of well-known visualizations, seeing whether they would fit into the proposed classification. Further we defined a manageable scope of two nesting levels, and checked whether all specifications in that scope would yield sensible visualizations.

2. COMBINATION METHODS FOR MAPPING RULE SETS

We first show, how known visualizations for elementary models can be integrated to produce more complex visualizations from multi models consisting of the respective elementary models using three different combination methods: Blending, Embedding, Interaction. The combination methods where derived by combining two visualization pipelines at the different stages of the pipeline model. The analysis in (Tauscher et al. 2011) showed, that these methods involve different main HCI usages and can be correlated with the dimensions of the visualization model space.

Blending combines information from the two different models into one visualization model and renders them into a single area. Information from the two input model is mapped to different objects in the shared visualization model or to different visualization properties of a shared visualization object. Blending represents information in a limited area at a specific point in time. The HCI resource involved most in the reception of blended visualizations is the recipient's cognition. A typical visual property used for blending is the color parameter, yielding chorophlet maps. We have also experimented with mapping to the size parameter (Tauscher & Scherer 2011).

Embedding maps information from the two different models onto two visualization models which are then combined by sharing the rendering space. One of the models may act as the positioning frame, while the other provides the details for embedded areas. The HCI resource engaged most by this visualization is display space, as information is distributed in space. This concept is called "small multiples" (Roberts 2007), "worlds within worlds" (Feiner & Beshers 1990) or "facets" (Wilkinson 2005).

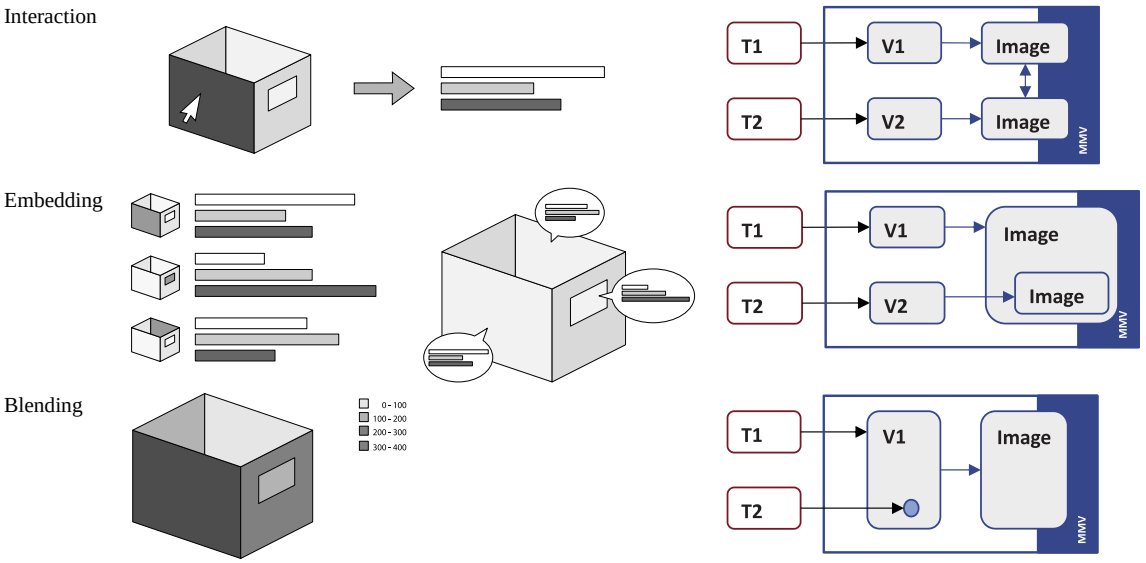
Interaction also maps information from the two different models into two visualization models which do not, as opposed to Embedding, share their rendering space. Instead the visualizations are rendered in different points in time. The points in time may be defined either by a schedule or by user events. Similar to Embedding the distribution in time may be guided by information from one of the two models. Obviously the HCI resource engaged most with this method is reception time. Research for this type of integrated visualization is carried out in the area of coordinated multiple views (Roberts 2007), where also the formal description of update operations is studied (Boukhelifa & Rodgers 2003).

These different combination methods tackle different points in the visualization pipeline. The integration points in the pipeline correspond to a different strength of the integration. Blending integrates at an early point in the visualization pipeline, by creating a common visualization model from different input data sets. Nesting integrates at a later point, creating different visualization models and rendering these in separate areas of the image. Interaction finally integrates even later; separately rendered images are connected by synchronized changes.

Implementation-wise, each successive combination method, beginning with the simplest one - Blending, adds a layer of complexity to the concepts implemented in the previous one: The Embedding mode introduces coordinate transformations and the Interaction mode requires a basic event system.

Table 1 shows a simplified application example for the different integration methods and a diagrammatic illustration of the integration in the visualization pipeline.

TABLE 1: Visualization combination methods



Previously we have analysed the correlation of these integration methods and the compositional characteristics of the multi model (Tauscher et al. 2011). Different types of elementary model pairs were contrasted with different methods to join single basic visualizations into a more complex visualization. We are now generalizing this approach to arbitrary parts of building information models, which do not necessarily have to be organized as multi models. This way the combination methods can be applied on different levels of detail in separating content for the visualization parts. For instance we can also think in smaller units than elementary models or group multiple elementary models into bigger units of information. Regarding the implementation in the framework this separation of content is done by selecting the input data for the respective visualization part, hence the subset of the information model which mapping rule set is to be applied to.

Breaking something down into smaller pieces can be done with similar methods as combining the smaller pieces into a bigger unit. Thus, complex visualizations can be broken down into smaller units by using the same three different methods as for the integration of visualization parts: Blending, Embedding, Interaction/Animation.

So far we described integration methods to combine two known visualization methods, expressed as mapping rule sets, into a more complex multi model visualization. However, we did not address the combination of more than two elementary or partial models. In general, the integration methods can easily be extended to accommodate more than two children. But there are two flaws with this naive approach: first, the visualization is limited to one single integration method and second, as the amount combined elementary models increases, it becomes difficult to maintain the visualization description.

We are now addressing this gap and extend the visualization framework and description structure to more complex cases. In the next section we will lift the integration methods to a higher level by nesting their application; that means by applying them recursively. We refer to and draw inspiration from research in the area of UI specifications, which does however not focus on visualizations (Coutaz 1987; Markopoulos 1997; Cai et al. 2000; Greer 2007).

3. HIERARCHICAL STRUCTURES

3.1 Handling complex issues with hierarchic structures

Hierarchical structures are a proven way to handle complex issues. By organizing visualizations with hierarchical principles, the description of complex visual representations can be carried out by everyone who is familiar with the structure of the source information and the domain specific visualization / presentation conventions.

Psychological studies have confirmed the limited capacity of human cognition of around seven entities in immediate memory. They concluded that to process more complex and larger amounts of information groups of entities have to be recoded into chunks (Miller 1955). These insights have been incorporated into research on user interfaces and have led to the pragmatic introduction of hierarchic modularity into the theoretical Model-View-Controller (MVC) pattern. The concept of hierarchic modularity is orthogonal to the monolithic layers of the MVC model and splits these layers into smaller, more manageable chunks (Coutaz 1987).

Using a hierarchical structure for the visualization specification we are repeatedly breaking a complex visualization into smaller more manageable parts using the combination methods described above. A visual representation is then constructed by recursively applying these modes to chunks of information, thus nesting visual representation parts and creating a hierarchical structure.

By including methods for the application of hierarchical nesting strategies in the framework and the visualization specification, users of the framework are encouraged to organize their visualization descriptions in a hierarchical way.

3.2 Tree representations

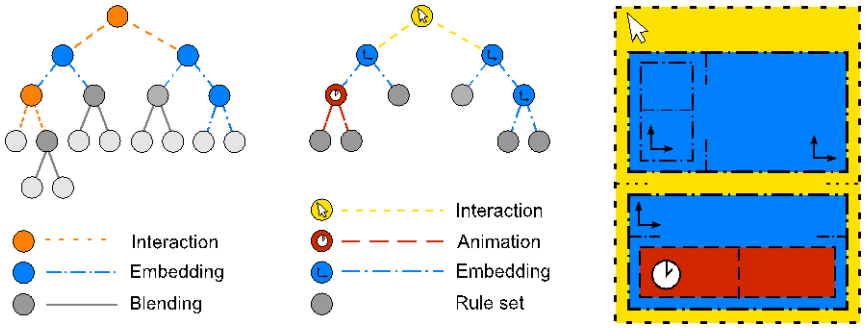


FIG. 2: Hierarchical nesting of the combination methods

Fig. 2 illustrates the application of the combination methods in a hierarchical manner. The hierarchical application yields a tree, with each sub tree representing a valid, simpler visualization. The example tree combines only two visualization parts on each level, yielding a binary tree. This is not a general restriction, but only made for the clarity of the example.

The left part of the figure shows the direct combination of the integration methods described before: On the upper level the example consists of two sub trees combined by Interaction, hence the visualization represented by the first sub tree is shown initially and replaced by the visualization represented by the second sub tree in case of a specified event or at a specified time. Each of these sub trees is further constructed by combining two separate visualizations using the Embedding method. That means, that the two visualization parts are rendered in different parts of the visualization space.

As opposed to Interaction and Embedding, integration by Blending does not create multiple visualization models. Instead the information from multiple input model parts is distributed to different properties of the same model. Because visualization properties cannot be further subdivided, as a consequence, Blending can only be the last integration. Blending does not take part in the hierarchical structure in the same way as the other methods. In the tree representation, Blending thus only appears as integration of leaf nodes. The final leaf nodes of the tree representation symbolize the mapping rule sets creating visualization objects.

3.3 Modifications to the initial integration methods

The middle part of Fig. 2 shows a modified version of the visualization structure. Sub trees consisting only of Rule sets integrated by the Blending method have been collapsed into single leaf nodes, according to the specific role of Blending in the visualization structure. Distribution of information to different visualization properties of the same object will be described inside a rule set instead of as an explicit combination.

Further, the Interaction integration method was divided into integration methods of two different subtypes: Animation and Interaction. Although both use time as the main HCI resource and are very similar from a technical point of view, differing only in the kind of trigger used to define the point in time for changes in the visualization model, the subsumption under a common label leads to constant confusion.

These two modifications to the initial model are supposed to increase the intuitive comprehensibility of the visualization structure model. Finally, we introduce the following symbols and line types for the integration methods to allow for a compact notation of visualization structures in the following sections:

- an arrow shaped pointer icon and dotted lines for the combination by interaction
- an iconic clock symbol and dashed lines for the combination by animation
- an iconic two-dimension coordinate system and dash-dot lines for the embedding method

3.4 Tree map notation

On the right side, the same tree is finally shown using a different type of representation: a tree map. Tree maps (Shneiderman 1992) are a means to lay out hierarchic structures in an area, subsequently dividing a given space according to the hierarchic structure. The original purpose of tree-maps was to encode values attached to tree nodes into the size property of the tree map areas, e.g. to visually represent the disk space used by an hierarchically organized file system.

The application of tree maps to hierarchical visualization structures allows to not only display the structure of the visualization, but also the content of the visualization leaf nodes in terms of the result of the exemplary application of the mapping rule set represented by the respective leaf node. Combined with the previously defined symbols and line types to encode the type of nesting, the tree-map representation provides a compact and intuitively understandable notation. It allows to represent animated and interactive visualizations in a static medium such as print.

4. NESTING CASES WITH DOMAIN SPECIFIC EXAMPLES

4.1 Evaluation strategy

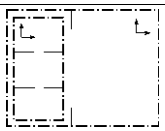
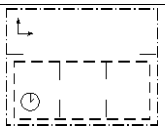
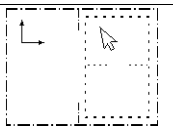
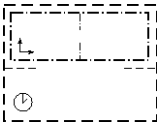
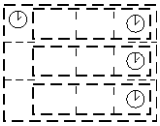
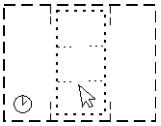
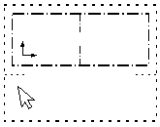
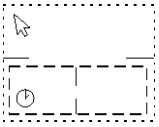
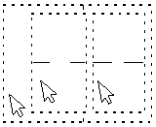
In this section we are evaluating the different scenarios emerging from this approach by nesting each mode into each other mode systematically. Simple examples representing these combinations reveal the potential of the approach.

The systematic analysis was carried out by nesting two combination methods in each case, running through all possible ordered pairs of combination methods. Since blending can only be applied to the leaf nodes it was left out of the matrix. Instead animation and interaction were treated as two different cases, expecting them to result in substantially different visualizations. The resulting combinations are shown in Table 2, with the outer combination method in rows and the inner combination method in columns.

Each nesting case consists of a tree with two levels. The arity of each node in the example cases is kept minimal while still producing useful visualizations. In particular, not all lower nodes do have children at all, but in general only one has children.

In the reminder of this section we are showing and explaining the cases in more detail. When converting the example visualizations for each case to a format suitable for print, animation and interaction had to be replaced by appropriate compensatory representations. Animation steps and interaction results are therefore laid out in the image area. Although not to be confused with the Embedding method, the resulting similar appearance confirms the interchangeability of the combination methods.

TABLE 2. Matrix of nesting cases, outer combination method in rows, inner combination method in columns

	embedding	animation	interaction
embedding			
animation			
interaction			

The examples use different visualization configurations, as specified by the combination method application matrix, but all examples are derived from the same building information model input. The building information is available as a multi model with 3D object data, a linked milestone schedule and progress information incurred during construction management. These elementary models are connected using a link model as proposed by Fuchs et al. (2011).

4.2 Case examples

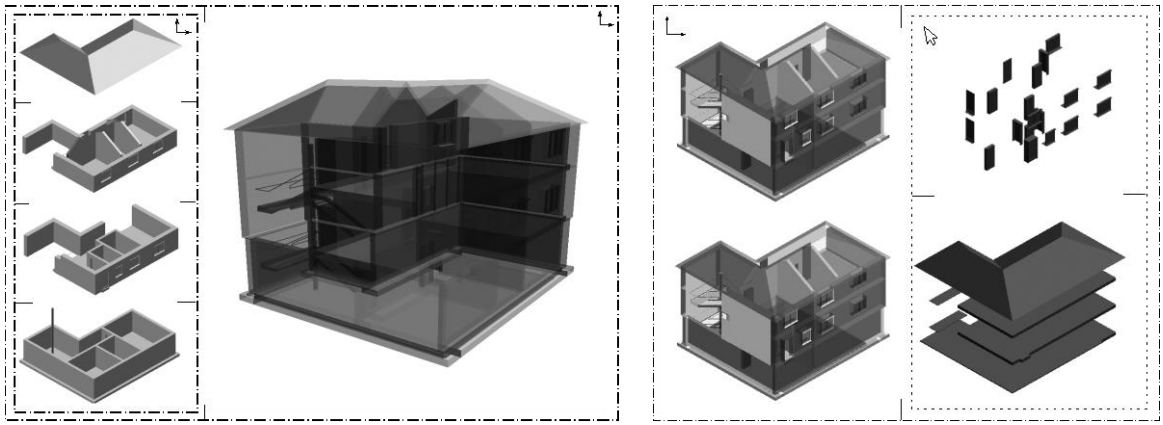


FIG. 3: Nesting case Embedding/Embedding (left) and Embedding/Interaction (right)

Embedding/Embedding: The first nesting case, shown in Fig. 3 on the left, features a classic architectural visualization method for the inner embedding: An axonometric exploded view of the building storeys and their floor plans. The outer embedding adds a perspective view of the full building with 3D-navigation features.

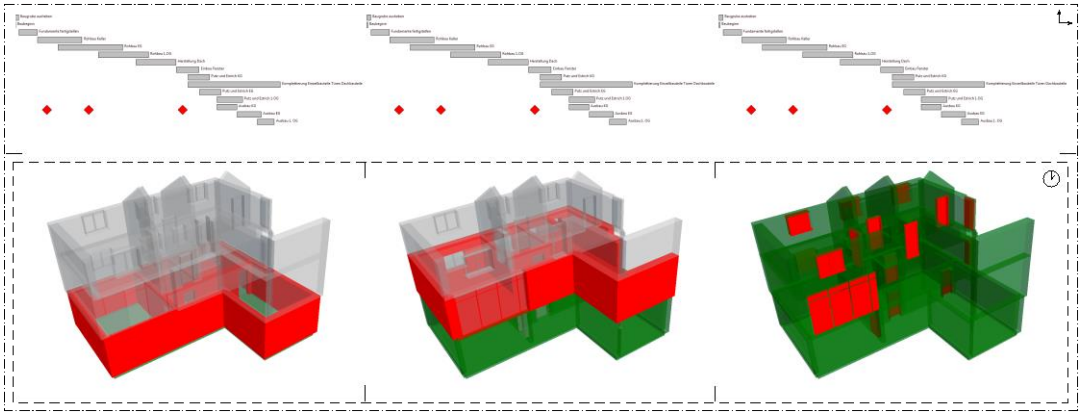


FIG. 4: Nesting case Embedding/Animation

Embedding/Animation: In the example for this case the visualization is divided into two parts, which are combined via Embedding. While one of the two parts stays fixed, the other part hosts an animation. The fixed top part consists of a Gantt chart with milestones and the bottom part consists of a nested animation showing a 3D view for the state of the construction progress at each of the milestones. The resulting visualization is shown in Fig. 4.

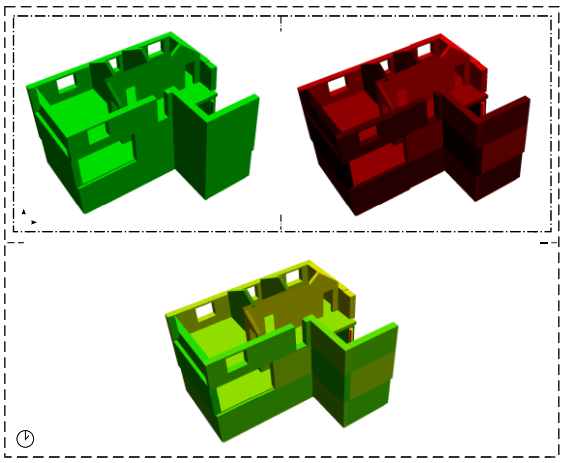


FIG. 5: Nesting case Animation/Embedding, similar to Interaction/Embedding

Embedding/Interaction: In this case only one part of the visualization is sensible to interaction changes. This sensible part is combined with a fixed part with the Embedding method. Note that the interaction combination refers to the part of the visualization which is changed as a result of the interaction and not to part of the visualization which is triggering the event. Fig. 3 (right) shows a respective example: One part of the visualization stays fixed, while the other part reacts to a query and hides all elements except the result set.

Animation/Embedding: For this case the outer combination method, an animation, contains at least one part with a split view, where multiple visualization parts are combined by Embedding. In contrast, the other parts of the animation are not split. The use case for this animation, shown in Fig. 5, compares the planned and the actually reported state of construction progress. The split view shows both sides of the comparison separately while the single view shows an overlay of the two.

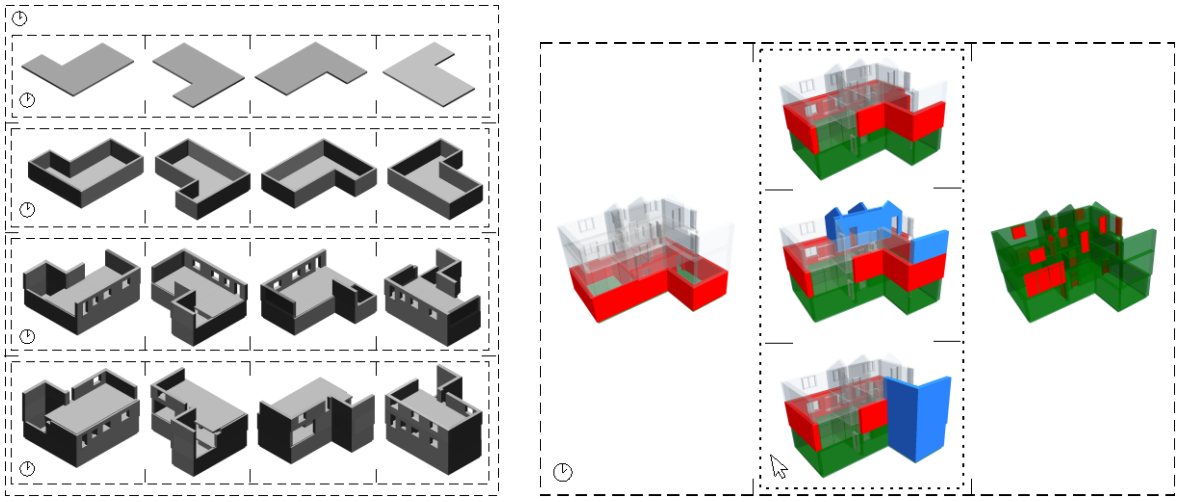


FIG. 6: Nesting case Animation/Animation (left) and Animation/Interaction (right)

Animation/Animation: In this case to separate animations are nested into each other. The outer animation of the example in Fig. 6 (left) covers different states of the building progress while the inner animation executes a 360° rotation of the whole scene for each step of the outer animation, in order to show all details of the 3D building.

Animation/Interaction: This case implies that only part of an animation is reacting to interaction requests. Hence the user or other external triggers would have to hit a certain time slot in the animation. The example in Fig. 6 (right) shows an animation of the building progress, with an additional highlighting in the third frame. The usefulness of this example is restricted. However, that does not render the nesting case illegitimate.

Interaction/Embedding: This nesting case produces subdivided visualization area where the subdivision is changed by user interaction. For instance an area is split into two parts to show several versions. This is similar to the Animation/Embedding case shown in Figure 5. While the animated case is of restricted usefulness, this nesting case is more common. Splits of the visualization area should generally be introduced intentionally, as the result of a user interaction. Splitting during an animation may be hard to understand and could easily confuse the recipient.

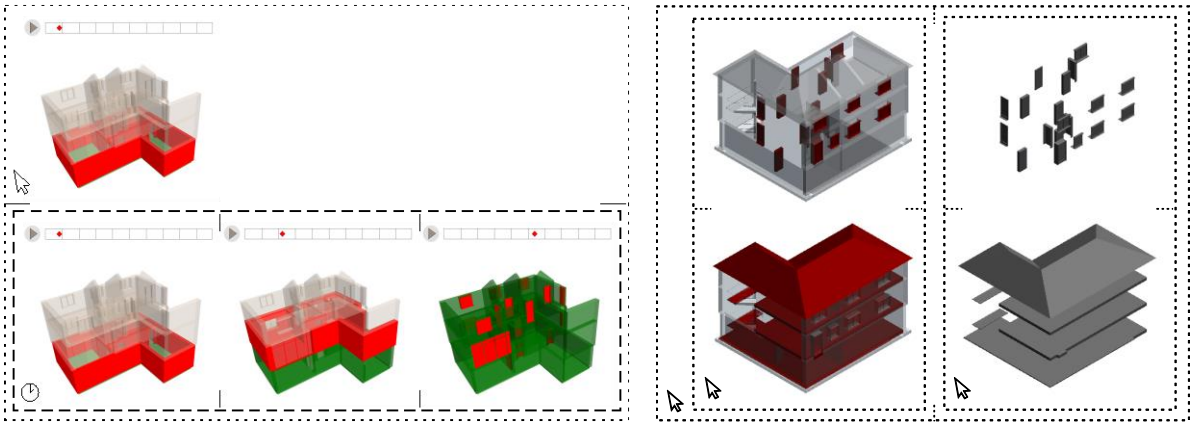


FIG. 7: Nesting case Interaction/Animation (left) and Interaction/Interaction (right)

Interaction/Animation: When the combination methods are nested in this way, then Interaction activates the nested animation or changes the parameters of the nested animation. This case is a common useful nesting case. An example is shown in Fig. 7 on the left side.

Interaction/Interaction: This nesting case makes sense when the inner interaction depends on the outer interaction having taken place before. In the example the outer interaction changes the way how the inner interaction works, it triggers different modes of highlighting changes. Fig. 7 (right) shows the results of applying the same inner interaction, a filter query, to the visualization after having chosen different highlighting modes in the outer Interaction combination. On the left side highlighting is done by marking the objects with a different colour, while on the right side highlighting is done by hiding all irrelevant objects.

4.3 Observations

While creating and examining these cases, we made the following observations in addition to the intended verification of the nesting approach.

The cases Embedding/Embedding, Animation/Animation and Interaction/Interaction have the special property that the outer and inner combination methods used for the nesting case are of the same kind. In these cases the nesting becomes a pure matter of giving structure to the visualization description. It is not absolutely necessary to use the combination methods twice. Instead, the nested solution could be expanded or resolved to a single application of the combination method with more elementary visualizations combined.

The example used for the case Animation/Animation is an exception where the same combination method is applied to all child nodes resulting in a tabular visualization structure. This case needs special coverage in the visualization structure to prevent high redundancy in the visualization description. Note also that this particular nature of the nesting results in a symmetric parent-child-relationship, which is invertible, a property not given in the other cases.

5. CONCLUSIONS

5.1 Summary

First, we clarified the requirements of the structure sought after. Second we described methods to combine two elementary model parts at a time. Third, we proposed a hierarchical nesting structure to extend the combination methods even further. Finally we analysed nesting cases systematically, supporting each case of two different combination methods with a construction related example.

The main contribution of this paper is a concept for the construction of complex visualization setups based on the nested application of combination methods. The combination methods allow for the construction of combined visualizations from simpler ones which are defined with mapping rule sets.

We further adapted the classification of the integration methods used for the specification: We have ruled out the Blending method as unreasonable on nesting levels other than the lowest level and we have separated the time based integration methods animation and interaction.

We have shown, that a hierarchical structure with nested mapping rule sets based on the basic combination methods Blending, Embedding and Interaction/Animation is a viable way to create arbitrary complex visualization specifications for a generic visualization application. Examples were produced as a proof of concept. The examples cover the whole space of nesting cases and thus allow for a systematic analysis.

It could be shown that the nesting cases do not produce unreasonable visualizations. However, two of the cases do not seem obviously useful in the first place and may not be needed often. However, all cases have to be supported in order to not restrict the possibilities of the visualization specification.

5.2 Outlook

We also wanted to show, that all possible visualizations can be produced with this approach, but this claim is only supported by the fact, that we did not find counterexamples so far. Because we systematically applied all combination methods, any edge cases should have occurred in the process. A set theoretic analysis could lead to further clarification.

From the exploration we concluded a suggestion for the further development and implementation of a special type of nested combinations, where the nesting is of a tabular form.

In a next step we can now integrate the nesting model into the visualization framework and create a domain specific language building upon the nesting model.

As another potential application of the model, visualizations described in terms of the proposed nesting structure could be assessed regarding the balance of information distribution in the different dimensions of the visualization space and thus the demands of different HCI resources for a given specific visualization could be analysed based on the description.

6. REFERENCES

Abdelhameed W. (2005). Digital-media impact on the representation capability of architects, *Proceedings of the 9th Iberoamerican Congress of Digital Graphics (Sigradi)*, Lima, Peru, pp 483–489.

Balakrishnan B., Kalisperis L.N., Muramoto K. and Otto G.H. (2006). Multimodel virtual reality environment for architectural design (re)presentation, *Proceedings of the 11th International Conference on Computer Aided Architectural Design Research in Asia (CAADRIA)*, Kumamoto, Japan, pp 513–519.

Boukhelifa N. and Rodgers P.J. (2003). A model and software system for coordinated and multiple views in exploratory visualization. *Information Visualization*, Vol. 2, No. 4, 258–269.

Cai J., Kapila R. and Pal G. (2000). HMVC: The layered pattern for developing strong client tiers, http://www.javaworld.com/javaworld/jw-07-2000/jw-0721-hmvc.html

Chang H.-S., Chih-Chung K. and Po-Han C. (2009). Systematic procedure of determining an ideal color scheme on 4d models. *Advanced Engineering Informatics*, Vol. 23, No. 4, 463–473.

Coutaz J. (1987). PAC, an object oriented model for dialog design, *Interact'87* (B.S. H.-J. Bullinger, editor), Elsevier Science Publishers.

Eastman C., Teichholz P., Sacks R. and Liston K. (2011). *BIM handbook. a guide to building information modeling. for owners, managers, designers, engineers, and contractors*, 2nd edition. Wiley.

Feiner S. and Beshers C. (1990). Visualizing n-dimensional virtual worlds with n-vision, *I3D '90 Proceedings of the 1990 Symposium on Interactive 3D Graphics*,

Fuchs S., Kadolsky M. and Scherer R.J. (2011). Formal description of a generic multi-model, *WETICE - 20th International Conference on Collaboration Technologies and Infrastructures*, Paris, France.

Greer D. (2007). Interactive application architecture patterns, http://aspiringcraftsman.com/2007/08/25/interactive-application-architecture/

Haber R.B. and McNabb D.A. (1979). Visualization idioms: A conceptual model for scientific visualization systems, *Visualization in Scientific Computing* (G.M. Nielson, B. Shriver, & L.J. Rosenblum, editors), IEE Computer Science Press, Los Alamitos, CA.

Liebich T. (1993). *Wissensbasierter architekturentwurf. von den modellen des entwurfs zu einer intelligenten computerunterstützung*. PhD thesis, Bauhausuniversität Weimar; VDG, Weimar

Markopoulos P. (1997). *A compositional model for the formal specification of user interface software*. PhD thesis, Queen Mary; Westfield College University of London, London

Miller G.A. (1955). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, Vol. 101, No. 2, 343–352.

Otjacques B., Noirhomme M. and Feltz F. (2006). Taxonomy of the visualization techniques of project related interactions. *ITcon*, Vol. 11, 587–605. [online] URL: *http://www.itcon.org/2006/41*

Roberts J.C. (2007). State of the art: Coordinated & multiple views in exploratory visualization, *Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization*, Zurich, Switzerland.

Shneiderman B. (1992). Tree visualization with tree-maps. 2-d space-filling approach. *ACM Transactions on Graphics*, Vol. 11, No. 1, 92–99.

Tauscher H. and Scherer R.J. (2011). Area cartograms in building product model visualization: A case study on the presentation of non-spatial object properties in spatial context with anamorphic maps, *Respecting Fragile Places (ECAADe 29)*, Ljubljana, Slovenia.

Tauscher H. and Scherer R.J. (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., Voigt M. and Scherer R.J. (2011). Integrating visual presentations of construction multi models. Visualization design space exploration, *ConVR 2011. Proceedings of the 11th International Conference on Construction Applications of Virtual Reality 2011* (H.-J. Bargstädt & K. Ailland, editors), Weimar, Germany.

Wilkinson L. (2005). *The grammar of graphics*. Springer.