CHRISTOPH RICHTER, HEIDRUN ALLERT, VASSILIY P. TCHOUMATCHENKO, IVAN H. FURNADZIEV, TANIA K. VASILEVA, DIMITRIS KOTZINOS, GIORGOS FLOURIS, VASSILIS CHRISTOPHIDES, JUHA LÖYTÖLÄINEN

5. A PRAGMATIC APPROACH TO COLLABORATIVE SEMANTIC MODELLING: THE VISUAL MODELLING (LANGUAGE) EDITOR

INTRODUCTION

Conceptual modelling has attracted a lot of research interest in recent years and various tools, such as Cmap[1], Compendium[2], FreeStyler[3], and Belvedere[4] have been developed to support the collaborative creation and work with various kinds of visual models. This interest is hardly surprising, given the prominent role of conceptual modelling across a large variety of professional and educational domains. As physical or digital artefacts, conceptual models provide important means for the explication, communication, and scrutinizing of each other's ideas and concepts. In order to guide the modelling process and to support mutual understanding among participants, semi-formal notations or modelling languages are used regularly both in professional, training as well as educational settings.

Recent research in the Learning Sciences provides important insights into the utility of conceptual modelling for learning, and current tools allow creating and working with these models more effectively (cf. Allert, Markkanen & Richter, 2006; Beguin, 2003; Suthers, Vatrapu, Medina, Joseph, & Dwyer, 2007). Nevertheless, we believe that, in order to unfold its full potential for learning and knowledge creation, we have to reconsider *models* as *epistemic artefacts* in the sense of investigative instruments and provide users with tools which not only allow them to create their own models but also put them in control of the semantics (i.e. the modelling languages) these models are build on. While predefined modelling languages provide scaffolds and can help to create common understanding, they might also impede collaboration and knowledge creation when they force users to stick to given perspectives and distract them from the issues and phenomena they are working on. Recent advances in semantic web technology provide new and more powerful means to support collaborative modelling (Braun, Schmidt & Walter, 2007; Domingue, 1998; Gangemi Presutti, Catenacci, Lehmann, & Nissim, 2007; Morita, Izumi, Fukuta & Yamaguchi, 2006; Sereno, Buckingham Shum, Motta, 2007) but, so far, respective applications have been overly complex to use and hardly in line with the pragmatic requirements of knowledge workers and students (cf. Hepp, 2007; Froehner, Nickles & Weiss,

A. Moen, A.I. Mørch, S. Paavola (eds.), Collaborative Knowledge Creation, 00–00.
© 20XX Sense Publishers. All rights reserved.

2004; Van Kleek et al., 2008). Towards that end, a high knowledge barrier, requiring substantial background knowledge to use the tools efficiently, an overemphasis on ontological considerations detached from other knowledge processes, combined with the use of static schemata as well as the enforcement of explicit and complete specifications beyond users' interest, appear to be the predominant problems that hamper the utilization of these technologies.

The KP-Lab Visual Modelling (Language) Editor (VM(L)E) provides a flexible and easy to use collaborative environment for the creation, use and evolution of conceptual models and their underlying languages in diverse domains of interest. The VM(L)E draws on the recent ideas for a pragmatic semantic web (McCool, 2005; Schoop, de Moor & Dietz, 2006) and considers modelling as an inherently epistemic activity that goes beyond the mere representation of what is already known and what can be agreed upon. The vision behind VM(L)E is to provide users with the possibility to create their own conceptual tools and thereby to advance pre-existing perspectives. In contrast to other tools, such as the Distributed Visual Language Environment, which require users to select a (self-defined) language beforehand (Hoppe, Gaßner, Mühlenbrock & Tewissen, 2000), the VM(L)E allows users to modify the underlying language throughout the modelling process.

Building on a brief introduction to its theoretical and empirical foundations, we outline the core motivating scenarios and high-level requirements underlying the design of VM(L)E. Against this background, design decisions and implementation is detailed and insights from field trials are reported.

COLLABORATIVE MODELLING – KNOWLEDGE PRACTICE & TOOL SUPPORT

Despite the significant interest in collaborative modelling for learning and knowledge building, it appears that research and development in computer-supported collaborative learning has been focused on the utilization of conceptual modelling (e.g. in the form of concept or argument mapping) for instructional purposes, while less attention has been paid to modelling as a means for knowledge creation. In the current discussion on conceptual modelling for learning, models are usually conceptualized as means for the explication and communication of knowledge, while the respective modelling languages provide methodological or instructional tools to foster and scaffold the modelling process. Models are thereby first and foremost characterized by their capability to represent a target system such as a certain phenomenon, a set of data, a theory, a domain of discourse or a product, in order to communicate, explain, or predict those phenomena of interest (cf. Frigg & Hartmann, 2006). Even though this approach appears to be appropriate when aiming to support the explication and communication of knowledge and *'to express the group's emerging consensus'* (Suthers et al., 2007), it undermines the epistemological value of models in that it restricts them to representations of what is already known.

Adopting a knowledge creation perspective on learning (cf. Paavola & Hakkarainen, 2005; Paavola, Engeström & Hakkarainen, this book), and building on the work of Knuuttila and Voutilainen (2003) and Knuuttila (2005) we understand the creation and manipulation of models as a genuinely epistemic activity that goes beyond the representation of a target system for communication purposes but aims to produce new insights and ideas. Drawing on the work of Morrison and Morgan (1999), Knuuttila argues that scientific models can be understood as *'investigative instruments'* or '*productive things*' which are partially independent of both the domain theory (or formal domain knowledge) and the world. Accordingly, a main purpose for the use of models is not to represent what is already known but, on the contrary, to come to terms with what is not yet known.

Conceptualizing models as epistemic artefacts, as proposed by Knuuttila (2005), has far reaching consequences for the understanding of models as well as modelling practices. First, models as manifestations of human agency are purposively created artefacts and not as an end in itself. Hence modelling should not be treated as an isolated activity but as an integral part of more overarching knowledge practices, such as scientific inquiry or product development for example. Second, models have, besides their conceptual, a material form and, therefore, are subject to the affordances and constraints of the medium used for modelling. These affordances and constraints are due to the technical as well as conceptual tools used for modelling. Modelling languages, here in the sense of conceptual tools, are crucial towards this end as they entail ontological commitments and make some aspects of domain more salient than others (e.g. Suthers, 2001). While fixed modelling languages are helpful to scaffold the modelling process and to establish a common understanding among participants, they easily become problematic from a knowledge creation perspective when they limit expressiveness or force participant to predefined perspectives. Third, models might become knowledge objects in their own right, and their creation and manipulation can result in new knowledge or even constitute new realities. The productive nature of models becomes especially apparent in such domains as health and engineering. Here, models are not just used to abstractly represent a target system but to actively design or intervene in the target system. For example, the reorganization of a business unit along newly-defined workflow models or the adoption of a new diagnostic scheme in a hospital reach beyond the realm of abstract representation but inherently affect the target systems they are supposed to model and have a direct bearing on reality.

Re-conceptualizing models as epistemic rather than as representational artefacts also poses new requirements for tools in support of collaborative modelling. The creation and use of models and their underlying languages should be as integrated as possible. Instead of treating modelling as a separate activity, collaborative modelling should be tightly integrated into the groups' work processes, allowing for easy access and reference to other resources used. Towards that end, tools for collaborative modelling should be an integral part of a respective learning and working environment. Rather than restricting users to a predefined set of modelling languages, they should be able to modify existing or create new languages whenever needed. To allow for an integrated work on models and modelling languages, users have to be able to move easily between both levels of abstraction without mixing them up. Furthermore, learners should be assisted in developing alternative models and in supporting the triangulation of different perspectives; these models can be based on the same or different modelling languages. Supporting long-term and boundary-crossing processes of knowledge creation affords the reuse, and evolution of the employed models and languages. Towards this end, users have to be aware of existing models and languages, to understand their specific purposes but also to adapt them to their local circumstances and own ideas. Allowing users to create and maintain their own modelling languages also requires powerful metaphors and easy-to-use tools to overcome the formalization barrier imposed by current tools. As concepts and their interrelations often become apparent, and crystallize only over a series of consecutive refinements and applications, learners should be supported in the systematic development and enrichment of models and their underlying languages. In order to trace the rationale of their evolution, means for comparing successive versions of models and languages have to be in place. Furthermore, whenever feasible, feedback should be provided to learners regarding possible consequences that a suggested change will have.

THE VISUAL MODELLING (LANGUAGE) EDITOR

The Visual Modelling (Language) Editor is part of the Knowledge Practices Environment, developed in the Knowledge-Practice Laboratory project (www.kp-lab.org). The Knowledge Practices Environment is a web-based collaborative environment offering various facilities for individuals and groups to interact with knowledge artefacts, knowledge process models as well as other users. The Knowledge Practices Environment aims at supporting students as well as practitioners in their working and learning activities. The environment provides users with flexible means to create, annotate, work on, and modify shared artefacts as well as to organize them visually (Lakkala et al., this book; Markkanen, Holi, Benmergui, Bauters, & Richter, 2008).

The Visual Modelling (Language) Editor provides an extension to the basic functionalities offered by the Knowledge Practices Environment and allows users to create, share, use, and update visual models as well as the underlying visual modelling languages as another type of shared artefacts. The aim of the Visual Modelling (Language) Editor is to provide users with an easy-to-use and customizable yet semantically powerful tool for collaborative modelling in diverse domains of interest. Exemplary application scenarios include, but are not limited to, the collaborative analysis and advancement of social practices, the modelling of problems, requirements, and options in design projects as well as the explication and analysis of logic models for evaluation and strategy development. In all these cases modelling is conceived as part of a more overarching activity whereby the model is meant to be an epistemic artefact for knowledge creation.

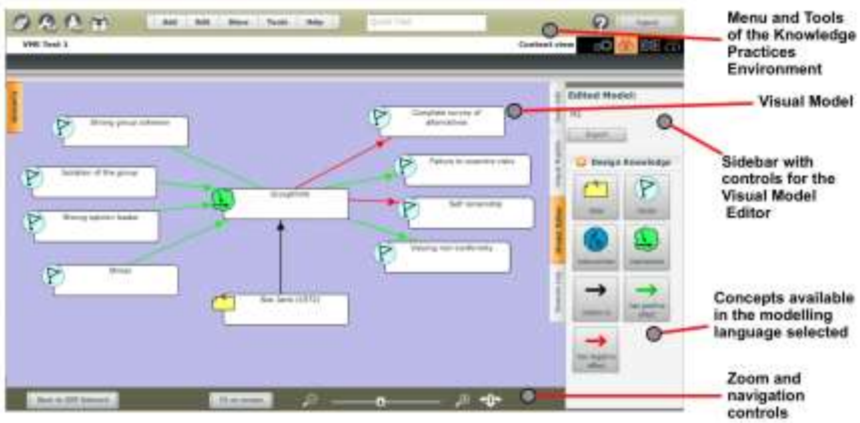

Figure 1. Graphical user interface of the Visual Model Editor

The Visual Model Editor comprises two core components: the Visual Model Editor, which allows users to create, compare and update visual models and the Visual Modelling Language Editor (VMLE), which provides users with the possibility to define and revise the underlying visual modelling languages and, hence, to specify the semantics of the models. Figures 1 and 2 depict the graphical user interfaces of the Visual Model Editor and Visual Modelling Language Editor.

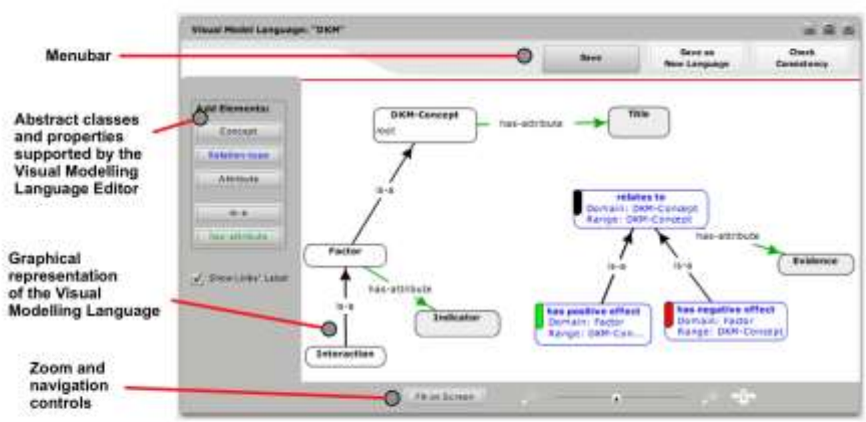

Figure 2. Graphical user interface of the Visual Modelling Language Editor.

The Visual Modelling (Language) Editor allows users to work collaboratively on visual models with explicitly defined semantics. The semantics are accessible to the user by means of the respective visual modelling languages. Providing access to the semantics of visual models and enabling users to revise and update these semantics while working on a particular model allows users to create their own conceptual tools. It also supports the use of visual models as epistemic artefacts since visual models can carry along the meaning their creators initially attributed to them.

The visual models as well as the visual modelling languages are represented as graphs to the user. While in principle other visual encodings would also be feasible, we decided to use graph-based visualizations, as they provide a very common metaphor familiar to many users and are extensively used in education as well as in professional domains. Furthermore, this type of visualization has a high degree of flexibility and can be easily handled by prevalent interaction techniques. To provide a better overview, even in the case of large-scale models, only the title of nodes and an icon representing the type of concept are displayed permanently. Additional information such as a description and concept specific attributes are displayed upon mouse rollover. Similarly the user can decide whether labels for the edges are to be displayed or not.

As shown in Figure 2, the visual modelling languages are depicted as graphs, with nodes symbolizing language concepts, relation types and attributes as well as edges representing is-a and has-attribute relations. The Visual Modelling Language Editor allows users to modify the language tree by adding/removing concepts and relation types (as instances of the metalanguage generic types), changing their attributes and defining constraints on the way concepts can be linked together in the visual models by specifying the properties' domain and range. Each model is based on a particular modelling language and is constructed from the concepts and relation types defined in this language. The metalanguage used to specify the visual modelling languages is based on a review of modelling languages and tools more commonly used in education as well as an analysis of the visual modelling languages that have been created within the KP-Lab project. Based on this analysis, and aiming to provide a tool also suitable for users with limited or no background in conceptual modelling, we decided to keep the metalanguage as simple as possible while being expressive enough to realize a broad array of visual languages used in education. An exception to this is that the Visual Modelling Language Editor allows users to specify attributes not only for concepts but also for relation types. This particular requirement arose from the analysis of the languages developed in KP-Lab project.

Another particular challenge, stemming from the attempt to allow users to work on the models and the underlying modelling languages simultaneously, is to find a proper mechanism to ensure the integrity between models and languages. This is due to the fact that visual models might evolve not only based on direct user-inflicted changes but also because of changes to the underlying modelling languages. Towards this end, various proposals, including the preview of effects on existing models as well as the semi-automatic update of models, have been discussed. In the current version a quite rudimentary solution has been implemented. To ensure consistency between a model and the modelling languages, the type of a node or edge is set to unknown in case the respective element has been deleted from the visual modelling language while the vertex or edge remains in the model. Although the semantics of this node or edge are not defined anymore, the user still has access to the respective information and can decide him- or herself whether to delete the node or edge or change its type according to the modified modelling language. A more advanced mechanism would also provide suggestions to the user, e.g., if there were alternative concepts that are semantically closest to the former one. In any case it appears important that the user stays in control of both the modelling language and the models and can trace the changes resulting from a modification of the language.

To support easy transition between modelling and other collaborative activities, the Visual Modelling (Language) Editor has been directly integrated into the Knowledge Practices Environment. Both visual models and visual modelling languages are represented as icons in the content view (a graphical display of the available artefacts) of the Knowledge Practices Environment and can be handled as any other content item. Once a visual model is opened it is displayed on a translucent layer 'on top' of the content view. Figure 4 shows an opened visual model with items on the content view visible in the 'background'. While the visual model elements can be identified as white rectangles with arrows between them, the content items are represented by the darker colour rectangles partly concealed by the model elements. As can also be seen from Figure 3, model elements can be directly linked with other artefacts at the group's disposal, hence providing an additional layer to structure shared resources.

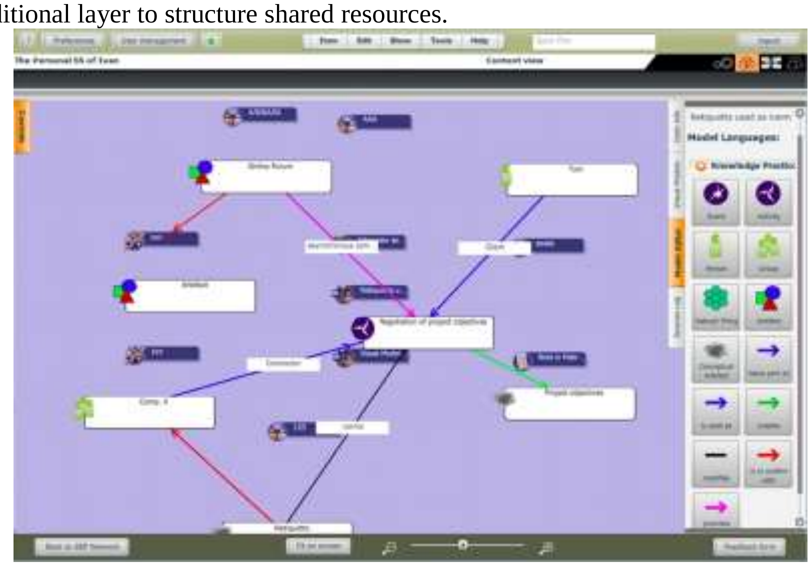

Figure 3. Screenshot of a visual model 'in front' of other resources available in the content view.

To support different modes of collaboration, the Visual Modelling (Language) Editor allows for both synchronous and asynchronous work. Therefore, changes are propagated in real time to all members of the group that are online but also logged by the system in order to trace back changes while someone has not been online when a model was modified. Commenting and chat functionalities provide additional support towards that end. Finally, the logs, as well as the models, can be exported in textual format for detailed analysis outside the Knowledge Practices Environment.

ARCHITECTURE AND TECHNICAL IMPLEMENTATION

The Knowledge Practice Environment is a distributed multi-tier web-based software system (Figure 4). The KPE front-end is implemented as a Rich Internet Application based on Adobe Flash and running inside a Web browser (a web browser with Adobe Flash support is also the only requirement for the client side, making it easy to use the KPE tools in any environment). The Flash client operation is supported by various 'front-end' web services, which constitute the middle tier of the web application. The supporting infrastructure for the 'front-end' web services (persistence, authentication, authorization, logging) is provided by the 'KPE Platform', which consists of several persistent storage spaces (e.g. databases) and web services.

The KPE user interface provides several 'views' (content, process, tailored and community views) and various 'tools' (note editor, chat, sketch pad, process tool, annotation tool, Visual Model Editor, Visual Modelling Language Editor and others). The tools are used to create content items, tasks and other KPE artefacts, while the views provide different perspectives and ways of organizing these artefacts. The content items and the other KPE artefacts are stored in repositories, provided by the KPE platform and, thus, are also shared by the rest of the tools in the platform.

In the KPE VML ontologies, the basic 'concept' class is a subclass of ContentItem class, form the KPE Trialogical Learning Ontology (TLO) – the core domain ontology used in the Knowledge Practices Environment. Similarly, the basic VML 'relationship' class is a subclass of the TLO:Relationship. This coupling with the TLO facilitates the integration of the VM(L)E tools in KPE and provides a unified view on the KPE artefacts. The visual languages, the visual models and their elements are all seen as content items by other tools in the Knowledge Practices Environment. This unification allows for interoperability among the different tools in the KPE; for example, for individual visual model elements to be annotated with the already existing annotation tool (the KPE Annotator) without the need of any external intervention.

At the middle tier, the VM(L)E tools rely on the Collaborative Semantic Modelling (CSM) front-end service for retrieving and storing the visual models and languages in the KPE Knowledge Repository. The translation of the visual model/languages graphs from/to RDF also takes place in the CSM service. The front-end services are implemented as HTTP based RPC-style SOAP services and deployed on Sun Glassfish Application Server. Another aspect of the VM(L)E operation is the collaborative editing, which requires reliable messaging for state synchronization and locking. This is handled by a dedicated synchronization service based on Adobe LiveCycle DS server. The presentation elements of the visual model/language graphs, like identification icons and line styles, are kept in the KPE Content Repository.

At the back-end tier of the KPE architecture, the Semantic Web Knowledge Middleware (SWKM[5]) serves as a gateway to the KPE Knowledge Repository (KR) and provides a suite of advanced knowledge management services that support actions both at the Visual Model and the Visual Modelling Language levels. These services include the ability to query and update the Knowledge Repository. SWKM supports advanced knowledge management functionalities that are superior to most of the existing knowledge management platforms, either generic ones like KAON (Gabel, Sure & Voelker, 2004) and Hozo (Kozaki, Sunagawa, Kitamura, & Mizoguchi, 2007) or specific ones targeting the e-learning arena like IMS Abstract Framework (Guangzuo, 2004) and ELF[6]. More on this can be found in Kotzinos, Flouris, Tzitzikas, Andreou, and Christophides (2008).

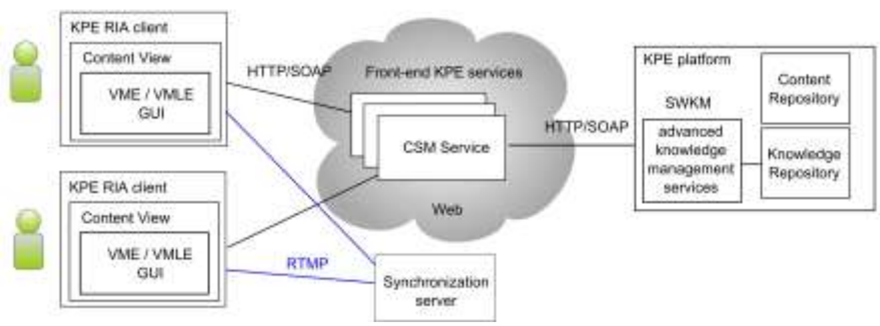

Figure 4. KPE / VM(L)E architecture.

FIELD TRIALS AND FINDINGS

Besides dedicated usability tests, the different releases of the Visual Model Editor have been used in several university courses carried out by the University of Helsinki as well as the University of Applied Sciences Upper Austria, throughout the winter-terms 2008/09 and 2009/10. In the following, we briefly describe the pedagogical scenario and findings obtained from the field trials carried out at the University of Applied Sciences Upper Austria in the bachelor programme 'Communication and Knowledge Media' and provide an example of how a teacher has used the Visual Modelling Language Editor to devise a Visual Modelling

Language she used to plan one of her courses at the Christian-Albrechts-Universität zu Kiel.

Students' Use of the Visual Model Editor

The compulsory cornerstone course 'eModeration' is aimed at fostering students' knowledge practices in solving complex design problems. Throughout their first semester students are asked to envision, develop, implement and evaluate a solution for a complex design problem in the fields of eCommunication and eModeration. To promote an inquiry-oriented and reflective design approach from the beginning, students have been asked to constantly explicate their understanding of the design space in the form of a conceptual model.

In the two iterations of this field trial that took place in winter terms 2008/09 and 2009/10, the students were introduced to the Visual Model Editor as well as the visual modelling language devised by the research team in close collaboration with the teacher.

In all, 35 students in 10 project teams took part in the first and 29 students in 9 teams took part in the second field trial. Both field trials lasted for about five months. The groups met face to face with the instructor alternately, every second week. Figure 5 shows a screenshot of the sample model created by the instructor to introduce the Visual Model Editor in the first iteration.

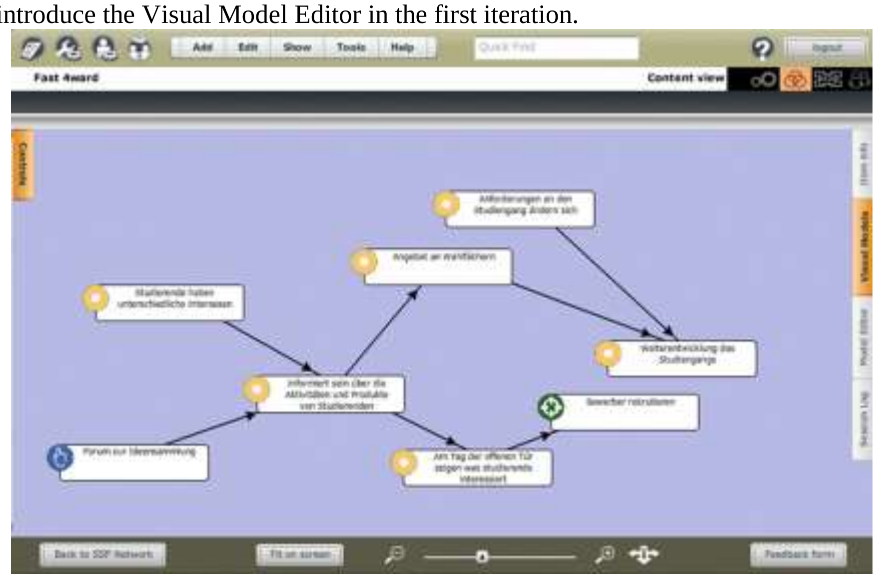

Figure 5. Sample model of the instructor.

As part of an accompanying research study, we have been interested in the students' appropriation and utilization of the Visual Model Editor while working on a complex design task. The primary aim of the study has been to better understand how students actually make use of the Visual Model Editor and the semantics of the languages provided as well as to inform the further development of the Visual Modelling (Language) Editor. The following observations and findings are based on an exploratory analysis of the models created by the project teams (screenshots had been taken on a weekly basis), the responses to a questionnaire administered to all students at the end of the first iteration, as well as interviews with representatives of four teams in the first and eight teams in the second iteration.

In both iterations the project teams responded quite differently to the modelling assignment, which is reflected, for example, in the number of additions and modifications made to the models over time, the overall number of models created as well as their structure. While there have been some groups that made only limited use of the Visual Model Editor or abandoned it after some first tryouts, the majority of groups worked on their models fairly systematically. The number of nodes created per group ranged from 18 to 107 in the first and form 9 to 36 in the second iteration.

Even though the Visual Modelling Languages had been introduced carefully to the students and scope notes for the different concepts are easily accessible via the Visual Model Editor, we found that the specified factor, which provided a kind of default concept in the language used in the first iteration, was used quite excessively by all teams (cp. Figure 6).

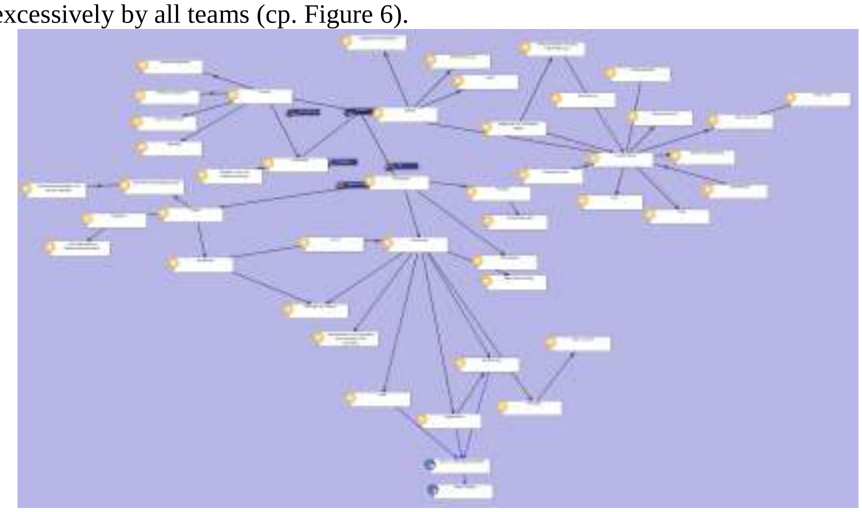

Figure 6. A model created in the beginning of a project. Even though the model is already quite complex, hardly any other concept than the specified factor (orange circle) is used.

Those groups who started to work on their models right from the beginning made hardly any use of the more specific concepts provided to depict their design space.

This behaviour only changed later, after the instructor provided additional guidance on how the different concepts could be used efficiently. In contrast, two teams that started to work on their models relatively late made more sophisticated use of the different concepts available right from the beginning. This finding is partly in conflict with the expectation that explicitly-defined concepts would scaffold students' elaboration of the design space. It might be that in the early phases of the design process the effort to explicitly classify ideas according to a predefined scheme does not outweigh the expected benefits, or even hinders a brainstorming-like collection of ideas. This assessment might change later on when the scope of the project becomes clearer and there is more need to structure and integrate the existing ideas. This interpretation is at least partly supported by the students' reports on how they created and used the models at the different stages of the project. In the second iteration, all teams made use of a broader set of concepts right from the beginning. This might be due to the teacher being more sensitized to this issue and providing better guidance from the beginning but also to the fact that the Visual Model Editor was introduced at a later point in the project, where the students already had collected a lot of information about the design space, which required some kind organization.

Some students in the first iteration mentioned that the predefined modelling language had been too restrictive and that they had difficulties mapping their ideas to the concepts provided. The examination of the models revealed that several teams had problems making proper use of the concepts provided: they mixed up resources and actions and/or, the understanding of the idea of typed nodes – for example, they introduced a concept specified in the Problem Analysis Language as a separate node. We found fewer such problems in the second iteration, providing some indication that the revised modelling language better matched the students' understanding. Besides these perceived limitations, we found at least one case n which a team actively introduced a new concept (the visual modelling Language Editor was not available at that time). In Figure 7 it can be seen that the team added a kind of prefix, in this case 'problem', to the title of several nodes in order to provide an additional 'typification'. These observations parallel those reported by Hoppe, Gaßner, Mühlenbrock and Tewissen,(2000), who found that the semantics of the concepts provided could not be taken for granted, but require a constructive explicit effort on the side of the users. The findings also underline the need for a possibility not only to edit the models but also the underlying modelling languages.

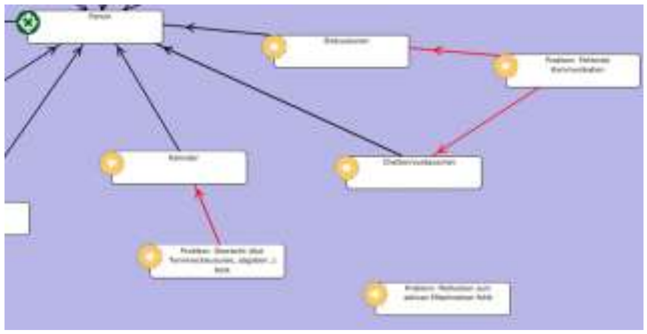

Figure 7. Partial screenshot depicting the extension of the modelling language by the prefix 'Problem' to title of the nodes on the bottom and the right side.

The comparison across teams in the first iteration revealed that those who used the Visual Model Editor more intensively often created more than one model in the project's lifetime. While, in some cases, the creation of multiple models was reportedly due to the fact that the current release of the Visual Model Editor does not allow changing the type of a node once it has been created, other groups created multiple models on purpose. Interviews with the students as well as examinations of the models revealed at least the following reasons for working with different models in parallel. One of the interviewee's reported that they had created multiple models in order to be able trace back their understanding at different stages and, hence, to provide some kind of history. In another case, the participants reported that they produced different models to depict different aspects of their project. A third group obviously used different models to elaborate and compare different project ideas, weighing the pros and cons of each proposal. In the second iteration we also found that about half of the teams created a new model in the course of the project rather than revising the existing one. Having a closer look at the conceptual similarity within and across models created by the teams, we found that the overlap coefficients between different versions of the same model have been very high (1.0 for all instances in the first iteration and between 0.91 and 1.0 in the second iteration), indicating that the teams added but did not delete elements. In contrast, overlapping coefficients across models created by a team were significantly lower, rarely reaching values above 0.5, indicating more significant changes in the contents of the models. One possible interpretation for the creation of multiple yet rather unrelated models is that students understood visual models as a means for documentation in the first place rather than as a cognitive tool in support of collaborative inquiry.

Finally, closer inspection of the visual models revealed that in some cases students obviously used the conceptual models not only as an epistemic artefact, depicting the design space, but also as a means to organize their collaborative work. For example, some models included open questions to be answered later on but also as kind of to-do-lists. This finding is also backed up by the students' reports, indicating that in several cases the models were also used to monitor and assess the work progress.

Even though these findings are preliminary, it appears that the adoption and utilization of semantically-rich conceptual models heavily depends on the direct added value for the user. The assessment of required efforts and expected benefits might also change, depending on the stage of the project as well as the actual task at hand. Consequently, tools to support collaborative modelling have to be quite flexible in order to accommodate the changing requirements that arise during the lifecycle of a project but also for the different strategies adopted by a particular group. Furthermore, the findings back up previous observations that the use of visual modelling language is a non-trivial task and that languages have to be designed carefully to provide meaningful scaffolds. Finally, the results point to the complexity of collaborative modelling as a real world practice that might not only fulfil an epistemic but also a social, pragmatic and even reflective purpose.

A Teacher's Use of the Visual Modelling Language Editor

In this section we describe a Visual Modelling Language that has been created by a teacher in order to plan and explicate the rationale behind one of her courses at the Christian-Albrechts-Universität zu Kiel. The teacher had been asked to expose her ideas on an upcoming course in order to attune and focus an accompanying research study. The teacher volunteered to use the Visual Modelling (Language) Editor for this purpose. Both the Visual Modelling Language and the Visual Model have been created during two meetings between the teacher and the researcher involved in planning of the research study. The teacher as well as the researcher had been familiar with visual modelling as well as the visual model editor and had collaborated on other projects before. Nevertheless, the primary aim of the meetings has been to describe the planned course rather than to specify a Visual Modelling Language.

Figure 8 shows a screenshot of the Visual Modelling Language Editor displaying the Modelling Language created by the teacher. The language includes eight concepts (ultimate goal, intermediate goal, intervention, outcome, context, input mechanism and wild card) and one relation type (influences).

Figure 8. Screenshot of the Visual Modelling Language created by the teacher.

Although the language is quite rudimentary, in that it does not specify any particular attributes or is-a relations, it provides the core concepts used to create what has been called logic-models in the field of programme evaluation (cf. Rogers, 2000). The teacher was familiar with logic models from previous works and, hence, the decision to build a respective language appears to be likely. Nevertheless, there are two things that appear to be noteworthy about how she actually implemented the language.

First, she introduced the concept ‘mechanism’ to denote ‘the process that is supposed to bring about a change’. While some authors in the field of programme evaluation, such as Rogers (2000), have made a strong point for the explication of the assumed mechanisms a programme is supposed to trigger, this is an often-neglected logic-modelling practice. Hence, the fact that the teacher deliberately included this concept into the language also means that she takes a certain perspective on logic modelling explicated in the language.

Second, the teacher included a concept called ‘wild card’ to denote ‘something not yet classified’. This concept provides a kind of placeholder and allows the user of the language to add elements to the model which appear somehow relevant to the model but do not yet seem to fit any of the other concepts. In contrast to the other concepts, the ‘wild card’ has a kind of pragmatic purpose in that it allows the user of the language to suspend decision on the correct classification of any idea immediately but lets them store the information and think about the appropriate classification later on. Even though this approach seem to be at odds with the idea of clearly-defined semantics, it perfectly reflects the fact that, in practice, users often have a hard time seeing immediately where an idea fits into the entire model. It also enhances the understanding that the ability to change the modelling language is highly necessary since, in this way, such practices would be limited or might even disappear.

Figure 9 shows a screenshot of the visual model the teacher created based on the language described above.

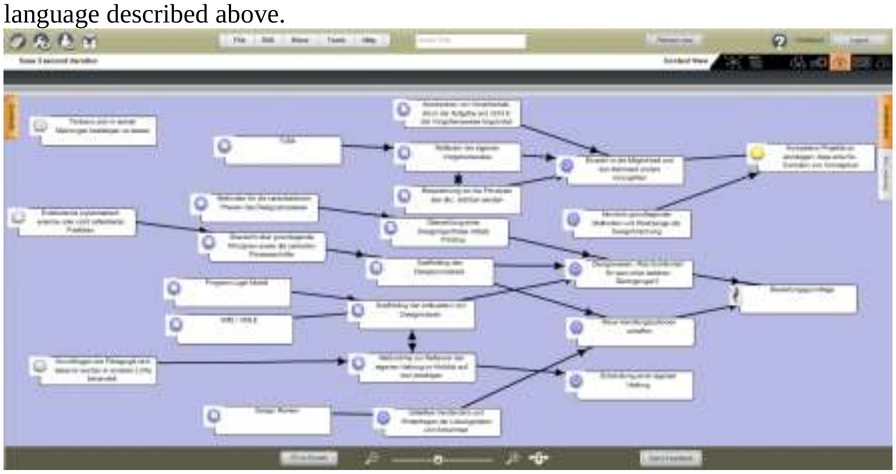

Figure 9. Screenshot of the Visual Modelling Language created by the teacher to depict the rationale for the planned course.

As the teacher has been familiar with visual modelling, as well as the visual modelling editor, it is difficult to make any generalizations from this example. Nevertheless, the example provides some more evidence that the metalanguage employed by the visual modelling language editor is indeed sufficient to specify language of interest for practitioners.

CONCLUSIONS AND FUTURE WORK

Collaborative modelling is a core knowledge practice across a huge variety of scientific and professional communities. Even though various researchers, instructional designers and developers investigate processes of collaborative modelling and explore new methods and technologies to foster these processes, the understanding of collaborative modelling as a knowledge practice is still in its infancy.

In this chapter we sketched briefly our understanding of modelling as an inherently epistemic activity going beyond the mere representation of what is already known and what can be agreed upon. Against this background we introduced the Visual Modelling (Language) Editor as an attempt to provide users with a flexible and easy-to-use but still semantically-powerful tool for the creation of visual models and their underlying modelling languages. The vision behind this tool is to provide users with the possibility of creating their own conceptual tools and, thereby, to advance pre-existing perspectives. Based on findings from the field trials with the Visual Mode Editor, it appears that the adoption and utilization of semantically-rich conceptual models to a large extent depends on the direct added value for the user, while at the same time modelling fulfils not only epistemic but also social and pragmatic purposes for the user. Additionally, experiments with the Visual Modelling Language Editor support the assumption that the chosen meta-language is suitable to define languages of practical value.

NOTES

[1] http://cmap.ihmc.us/
[2] http://compendium.open.ac.uk
[3] http://www.collide.info
[4] http://belvedere.sourceforge.net
[5] SWKM Website: http://139.91.183.30:9090/SWKM
[6] http://www.elframework.org/

REFERENCES

Allert, H., Markkanen, H. & Richter, C. (2006). Rethinking the Use of Ontologies in Learning. *2nd Workshop on Learner-Oriented Knowledge Management & KM-Oriented E-Learning (LOKMOL 2006)*, Crete, Greece, October 2006.

Beguin, P. (2003). Design as a mutual learning process between users and designers. *Interacting with Computers, 15*, 709–730.

Braun, S., Schmidt, A., & Walter, A. (2007). 'Ontology Maturing – A collaborative Web 2.0 Approach to Ontology Engineering', Workshop on Social and Collaborative Construction of Structured Knowledge at WWW, 2007. *Proceedings of the Workshop on Social and Collaborative Construction of Structured Knowledge (CKC 2007) at the 16th International World Wide Web Conference (WWW2007) Banff, Canada, May 8, 2007*, CEUR Workshop Proceedings vol. 273.

Domingue, J. (1998). Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web. *11th Knowledge Acquisition for Knowledge-Based Systems Workshop. Banff, Canada, 1998*.

Frigg, R. & Hartmann, S. (2006). Models in Science. In: E.N. Zalta (Ed.). *The Stanford Encyclopedia of Philosophy*. Retrieved from http://plato.stanford.edu/entries/models-science.

Froehner, T., Nickles, M. & Weiss, G. (2004). Towards modeling the social layer of emergent knowledge using open ontologies. *Proceedings of The ECAI 2004 Workshop on Agent-Mediated Knowledge Management (AMKM)*, 2004.

Gabel T., Sure Y. & Voelker J. (2004). KAON-Ontology Management Infrastructure. *SEKT informal deliverable, 3*(1).

Gangemi, A., Presutti V., Catenacci, C., Lehmann, J. & Nissim, M. (2007). CODO: an OWL metamodel for collaborative ontology design. *WWW2007, May 8–12, 2007, Banff, Canada*.

Guangzuo C. (2004). OntoEdu: Ontology based Education Grid System for e-learning. *GCCCE2004, HongKong, 2004*.

Hepp, M. (2007). Possible Ontologies: How Reality Constrains the Development of Relevant Ontologies. *IEEE Internet Computing, 11*(1), 90-96.

Hoppe, H.U., Gaßner, K., Mühlenbrock, M., & Tewissen, F. (2000). Distributed Visual Language Environments for Cooperation and Learning: Applications and Intelligent Support. *Group Decision and Negotiation, 9*, 205-220.

Knuuttila, T. (2005). *Models as Epistemic Artefacts: Toward a Non-Represenationalist Account of Scientific Representation.* PhD Thesis. University of Helsinki, Helsinki, Finland.

Knuuttila, T. & Voutilainen, A. (2003). A Parser as an Epistemic Artefact: A Material View on Models. *Philosophy of Science, 70*(5), 1484-1495.

Kotzinos D., Flouris G., Tzitzikas Y., Andreou D. & Christophides V. (2008) Supporting Evolution of Knowledge Artefacts in Web Based Learning Environments. In N. Karacapilidis (Ed.). *Solutions and Innovations in Web-Based Technologies for Augmented Learning: Improved Platforms, Tools and Applications* (chapter IX). London: IGI Global.

Kozaki K., Sunagawa E., Kitamura Y. & Mizoguchi R. (2007). A Framework for Cooperative Ontology Construction Based on Dependency Management of Modules. *Proceedings of the International Workshop on Emergent Semantics and Ontology Evolution, 12 Nov. 2007, Bexco, Busan Korea*.

Markkanen, H., Holi, M., Benmergui, L., Bauters, M. & Richter, C. (2008). The knowledge Practices Environment: A virtual environment for collaborative knowledge creation and work around shared artefacts. *World Conference on Educational Multimedia, Hypermedia and Telecommunications 2008*, AACE: Chesapeake, pp. 5035-5040.

McCool, R. (2005). Rethinking the Semantic Web, Part I. *IEEE Internet Computing, 9*(6), 86-87.

Morita, T., Izumi, N., Fukuta, N. & Yamaguchi, T. (2006). A Graphical RDF-Based Meta-Model Management Tool. *IEICE TRANS. INF. & SYST., 89*(4), 1368-1377.

Morrison, M. & Morgan, M. (19999: Models as Mediating Instruments. In M. Morgan & M. Morrison, (Eds.). *Models as Mediators (pp. 10-37).* Cambridge: Cambridge University Press

Paavola, S. & Hakkarainen, K. (2005). The Knowledge Creation Metaphor – An Emergent Epistemological Approach to Learning. *Science Education, 14*(6), 535-557.

Rogers, P.J. (2000). Program Theory: Not Whether Programs Work But How They Work. In: D.L. Stufflebeam, G.F. Madaus, & T. Kellaghan (Eds.). *Evaluation Models: Viewpoints on Educational and Human Services Evaluation (pp. 209-232).* 2nd ed.- Boston: Kluwer.

Schoop, M., de Moor, A. & Dietz, J.L.G. (2006). The Pragmatic Web: A Manifesto. *Communications of the ACM, 49*(5), 75-76.

Sereno, B., Buckingham Shum, S., & Motta, E. (2007). Formalization, User Strategy and Interaction Design: Users' Behaviour with Discourse Tagging Semantics. *Proceedings of the Workshop on Social and Collaborative Construction of Structured Knowledge (CKC 2007) at the 16th International World Wide Web Conference (WWW2007) Banff, Canada, May 8, 2007*, CEUR Workshop Proceedings vol. 273.

Suthers, D.D., Vatrapu, R., Medina, R., Joseph, S. & Dwyer, N. (2007). Conceptual Representations Enhance Knowledge Construction in Asynchronous Collaboration. *Proceedings of the Computer Supported Collaborative Learning (CSCL) Conference 2007, New Brunswick, NJ, USA, July 16 - July 21, 2006*, pp. 704-713.

Suthers, D.D. (2001). Collaborative Representations: Supporting Face to Face and Online Knowledge Building Discourse. *Proceedings of the 34th Hawaii International Conference on the System Sciences. Maui, 2001*: IEEE Computer Society.

Van Kleek, M., Bernstein, M., André, P., Perttunen, M., Karger, D. & Schraefel, M. (2008). Simplifying knowledge creation and access for end-users on the SW. *Proceedings of CHI 2008 Workshop on Semantic Web User Interfaces, Florence, Italy, 2008*.

AFFILIATIONS

Christoph Richter,
Christian-Albrechts-Universität zu Kiel,
Kiel,
Germany
Heidrun Allert,
Christian-Albrechts-Universität zu Kiel,
Kiel,
Germany
Vassiliy P. Tchoumatchenko,
Technical University of Sofia,

Sofia,
Bulgaria
Ivan H. Furnadziev,
Technical University of Sofia,
Sofia,
Bulgaria
Tania K. Vasileva,
Technical University of Sofia,
Sofia,
Bulgaria
Dimitris Kotzinos,
FORTH-ICS,
Heraklio,
Greece
Giorgos Flouris,
FORTH-ICS,
Heraklio,
Greece
Vassilis Christophides,
FORTH-ICS,
Heraklio,
Greece
Juha Löytöläinen
Helsinki Metropolia University of Applied Sciences,
Helsinki,
Finland