Ontology Design for Visual Modeling Languages

Vassiliy Platonovitch Tchoumatchenko and Tania Krumova Vasileva

Abstract – **Conceptual modeling constitutes an important knowledge practice in quite many professional and scientific communities but also in various educational settings. The Visual Model Editor (VME) provides a collaborative tool for creation, use and evolution of visual models and their underlying languages. A paper outlines the visual modeling languages currently implemented in the VME. Their common characteristics are identified in order to facilitate modeling and ontology design is discussed. An example of using visual modeling languages and their primitives in VME is also given.**

Keywords – **Collaborative semantic modeling, Ontology, Visual models**

I. Introduction

The Web is evolving from a huge information and communication space into a massive knowledge and service repository. One of the enablers of the above change is ontology, commonly referred to as the conceptualization of a domain [1]. Ontology provides a sound semantic ground of machine-understandable description of digital content. It is ubiquitous in information systems [2] by annotating documents with meta-data, improving the performance of information retrieval and reasoning, and making data between different applications interoperable [3-4]. In addition, ontology-type semantic description of behaviors and services allow software agents in a multi-agent system to better coordinate themselves.

Ontology is one of the fundamental cornerstones of the semantic Web. The pervasive use of ontologies in information sharing and knowledge management calls for efficient and effective approaches to ontology development. Making ontologies operational in the context of the Web and other large distributed systems requires a considerable amount of research effort towards developing methodologies and technologies for constructing and maintaining domain-specific ontologies in a dynamic environment [5].

People have been using computer systems for decades to model the things in the "real world" which they study. Conceptual modeling constitutes an important knowledge practice in quite many professional and scientific communities but also in various educational settings. A conceptual model is a formal model in which every entity being modeled in the real world has a transparent and one-to-one correspondence to an object in the model. Conceptual model declares all the object classes in the problem domain and their attributes, including integrity constraints on attributes that store values and built- in queries on those that compute their values on-the-fly. Visual model declares one or more ways in which objects of each class can be formatted for display to the user [6].

Recent advances in semantic web technology provide new and more powerful means to support collaborative modeling activities by allowing the users to create, share, and advance their own models and modeling languages. Nevertheless, most of the existing tools do not take into account the pragmatic needs of students and knowledge workers.

The Visual Modeling (Language) Editor currently under development within the framework of the KP-Lab Project [7] provides a collaborative tool for the creation, use and evolution of visual models and their underlying languages. The aim is to provide users with an easy to use and customizable but yet semantically powerful tool for collaborative modeling in diverse domains of interest. The Visual Modeling (Language) Editor is an extension to the basic functionalities of Knowledge Practices Environment (KPE). KPE is a web-based collaborative working and learning environment offering various facilities for creating and interacting with knowledge artifacts and knowledge process models as well as for collaborating with other users.

The Visual Modeling (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 modeling languages.

The paper outlines three visual modeling languages (VMLs) used in the initial Visual Model Editor (VME) and Visual Modeling Language Editor (VMLE) tools implementation. The common characteristics of these languages are identified and the ontology design is considered.

II. Visual Modeling Languages

Three visual modeling languages (VMLs) are addressed in the initial VME implementation: Problem Analysis Language (PAL), Knowledge Practices Descriptive Framework language (KPDF) and Concept Maps language (CMAP). We will provide short overview of these languages so as to identify common characteristics in order to facilitate the modeling.

A. Problem Analysis Language

A problem analysis language is meant to support students in exploring the problem space of their task at hand as well as possible design solutions while working on a joint project.

V. Tchoumatchenko is with the Department of Electronics and Electronics Technologies, Faculty of Electronic Engineering and Technologies, Technical University - Sofia, 8 Kliment Ohridski blvd., 1000 Sofia, Bulgaria, e-mail: vpt@tu-sofia.bg

T. Vasileva is with the Department of Electronics and Electronics Technologies, Faculty of Electronic Engineering and Technologies, Technical University - Sofia, 8 Kliment Ohridski blvd., 1000 Sofia, Bulgaria, e-mail: tkv@tu-sofia.bg

The Problem Analysis Language is described in terms of concepts and attributes.

Figure 1 provides an overview of the core concepts and attributes. The concept *link* (and it's sub-classes *affects*, *casual influence* and *inverse casual influence*) is further defined as relationship between two Factors. All *links* can have attributes *title*, *description* and *evidence*. The attributes *title*, *description* and *operationalization* are defined for *specified factors* (and sub-classes). All attributes have free-text values.

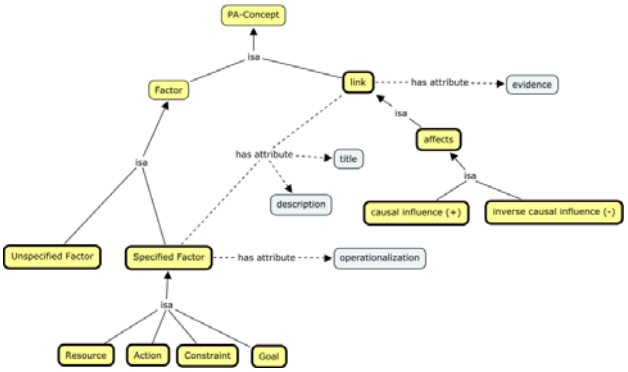

FIGURE 1. PAL CONCEPTS AND ATTRIBUTES

Figure 2 shows an example of a PAL visual model. It shows how the notational system can be used to describe a problem space, including available resources, possible actions, existing constraints as well as goals.

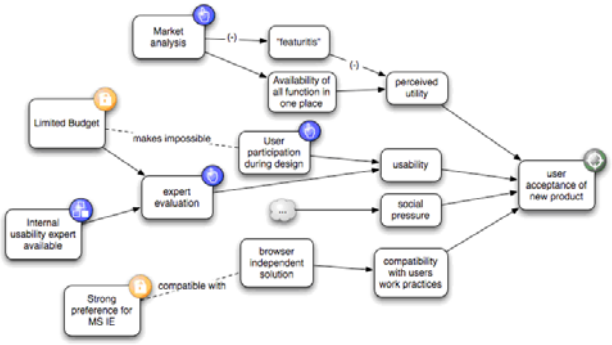

FIGURE 2. EXAMPLE OF PAL VISUAL MODEL

Note that the texts *makes impossible* and *compatible with* are the values of the title attribute for the respective links.

B. Knowledge Practices Descriptive Framework Language

A descriptive framework for knowledge practices is focused on overt behavior and interaction of the persons involved and aims to capture the complex interrelations of multiple activity systems. The descriptive framework proposed here is based on some basic design decisions which are:

Event-oriented modeling: Even though collective activities could be modeled from an actor or document centered perspective the descriptive framework outlined here follows an event-oriented approach, whereby activities are understood as a special kind of event including actors, tools as well as objects of activity to be transformed by the activity. In contrast to a state-oriented approach, emphasize is on the processes instead of outcomes.

Role-based modeling: In addition, a role-based approach for modeling activities was chose n, in order to account for the context-dependency of the roles filled by persons and objects within a particular activity. In contrast to other approaches the role-concept is not only applied to persons but also to objects both physical as well as conceptual [8].

The Knowledge Practices Descriptive Framework Language (KPDF) defines a hierarchy of concepts with specific attributes. The possible relationships between the concepts are shown on Figure 3. The relationships *uses as*, *takes parts* as have attributes, which are instances of the concepts *MA Role* and *Actor role* respectively. The relationship *is in conflict with* has a textual attribute *description*.

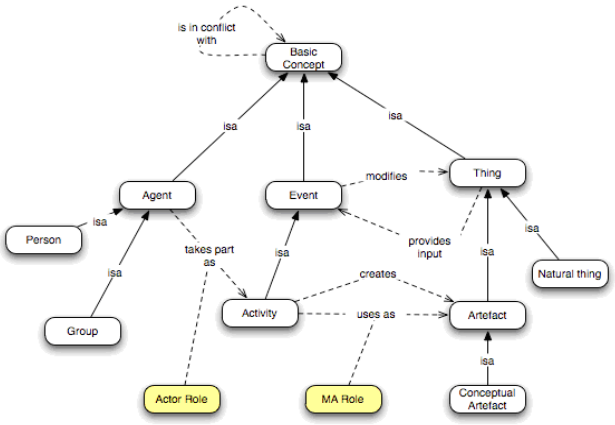

FIGURE 3. KPDF CONCEPTS AND RELATIONSHIPS

Fig.4 describes an interview situation as a set of actions. In this case Karl, as the interviewer, is asking questions on prior learning experiences. In order to do so he is using some guiding questions he thought of before the interview as well as a tape-recorder to record the interview. On the other hand, Ann as the interviewee is responding to Karl's questions and explores her learning experiences, whereby her answers in turn trigger new questions by Karl. Both Karl's questions and Ann's answers are stored as an audio file. This real world situation could be modeled with visual model based on KPDF language as shown in figure below.

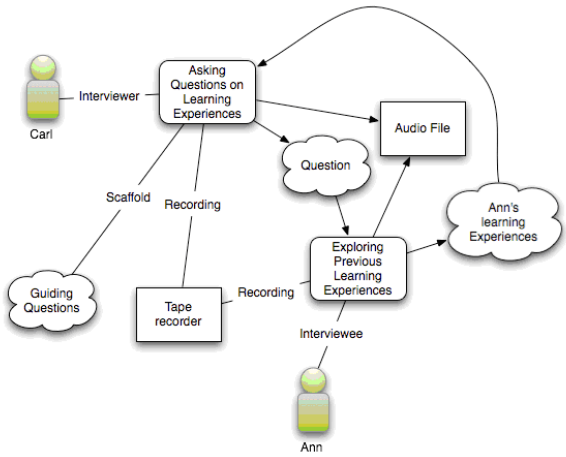

FIGURE 4. EXAMPLE OF KPDF VISUAL MODEL.

In the example, the Agent "Carl" is in a *takes part as* relationship with the *Activity* "Asking Question on Learning Experiences". This relationship has an attribute *role*, whose value is instance of *Actor-role* concept with title "Interviewer".

The *Conceptual artifact* "Guided questions" is in *uses as* relationship with the *Activity* "Asking Question on Learning Experiences". This relationship has an attribute *role*, whose value is instance of *MA-role* concept with title "Scaffold".

C. Concept Maps Language

The CMAP language (CMAP) has simpler class hierarchy compared to PAL and KPDF. It consists of a single "node element" – Concept/Phenomenon and several relationships (see Table 1). Only the Concept/ Phenomenon has attributes title and description with textual values. The relationships have no attributes.

TABLE 1. CMAP LANGUAGE ELEMENTS

Element type	**CMAP Element Name**
Node	Concept/Phenomenon
Relationship	Relation
	Definition of Purpose/Function of Feature of Consists of Type of
	Consequence/Outcome of Affects/Regulates Prerequisite of Influence of
	Situation related to Context of/Used in Environment of
	Activity related to Actor in Object of Mean/tool of
	Chronological stage of Iterative stage of

For each visual modeling language, used in the VME, a domain specific ontology is devised.

III. ONTOLOGY FOR VISUAL MODELING LANGUAGES

Ontologies are aimed to provide knowledge about specific domains that are understandable by both developers and computers. In particular, ontologies enumerate domain concepts and relationships among the concepts. They may also explicitly define properties, functions, constraints, and axioms.

The analysis of the PAL, KPDF and CMAP languages shows that they are based on concepts – visually presented as graph vertices and relationships – presented as graph edges. In all three languages the concepts have string-valued attributes like title and description.

In the PAL and KPDF languages the relationships also have attributes, which might be string-values or object-valued (see figures 5 and 6). This particular requirement was the single most problematic modelling issue, we encountered while implementing the VME/ VMLE tools.

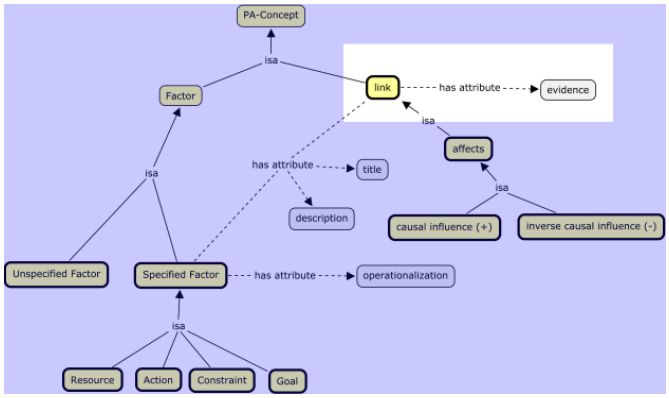

FIGURE 5. EXAMPLE OF RELATIONSHIP WITH STRING-VALUED ATTRIBUTE IN PAL.

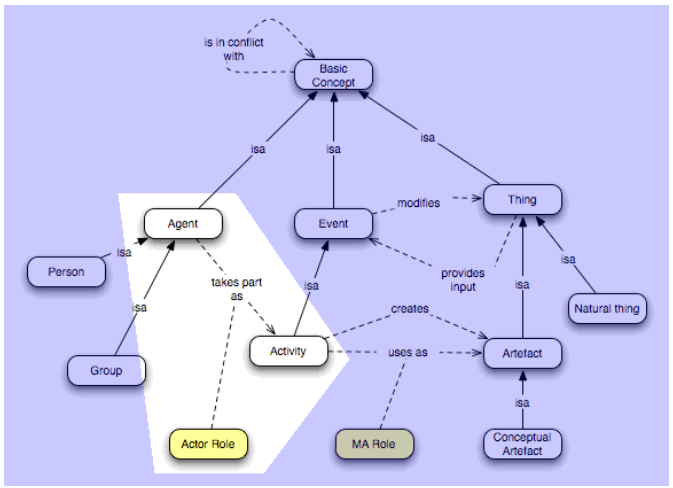

FIGURE 6. EXAMPLE OF RELATIONSHIP WITH OBJECT-VALUED ATTRIBUTE IN KPDF.

Ontology representation is the most fundamental issue in ontology development. In addition to making ontologies understandable by computers and humans, an ontology representation language should also provide representation adequacy and inference efficiency. The standardization of ontology representation languages (e.g., RDF, RDFS [9]) has taken big strides in the past few years. The above languages have mainly adopted a frame-based knowledge representation paradigm.

In the visual languages, currently used in the KPE, we were able to identify two basic hierarchies – "concept" and "relationship". They were modeled as RDF classes and shown as vertices and edges in the visual graph representation. Specific attributes, e.g. title, description, of the concepts and relationships are modeled as RDF properties. In the KPE VML ontologies, the basic "concept" class is a subclass of ContentItem class form the KPE Trialogical Learning Ontology (TLO). Similarly, the basic VML "relationship" class is a subclass of the TLO:

Relationship. This coupling with the TLO facilitates the integration of the VME/VMLE tools in KPE and provides a unified view on the KPE artifacts. The visual languages, the visual models and their elements are all seen as content items by the non-VME/VMLE tools. This unification allows, for example, annotating individual visual model elements with the already existing KPE Annotator, without changing the tool and making it "VM aware".

The PAL and KPDF language has similar modeling problems because they define relationships with attributes. The adopted solution was to model both concepts and relationships as RDF classes and the attributes as RDF properties. Figure 7 depicts the class diagram of the "concept" hierarchy in one of the KPE VMLs.

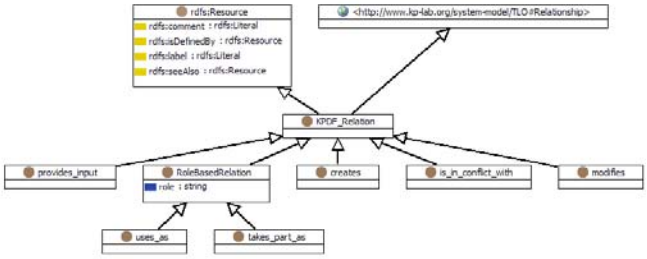

FIGURE 7. CLASSES AND PROPERTIES IN THE KFDP ONTOLOGY

Compared to PAL and KPDF, the CMAP modeling does not creates any new problems.

IV. EXAMPLE OF USING VISUAL MODELLING LANGUAGES

Figure 8 shows a screenshot of the current prototype of the Visual Model Editor, integrated in KPE. The VME supports multiple languages model. User can choose one of the visual modeling languages. The Model Editor tab contains palette with the elements of the corresponding visual modeling language(s). The user adds nodes or links (concepts or relationships) to the model, by selecting them from the palette.

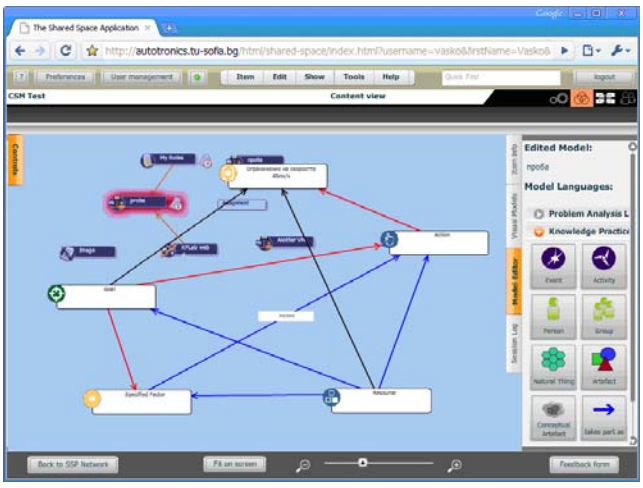

FIGURE 8. VME WITH MULTIPLE LANGUAGES MODEL

V. CONCLUSION

Collaborative modeling is a core knowledge practice across a huge variety of scientific and professional communities. The developed Visual Model Editor provide users with a flexible and easy to use but still semantically powerful tool for the creation of visual models and their underlying modeling languages. It allows collaborative work on visual models with explicitly defined semantics. The semantics are accessible to the user by means of the respective visual modeling languages.

A short overview of visual modeling languages currently implemented in Visual Model Editor is provided. The common characteristics are identified in order to facilitate the modeling. An ontology design is discussed.

The work reported in this paper is still work in progress. At present the prototype is used in two university courses to test its usability and pedagogical utility under real world conditions. The results of these field trials will feed into a new version of the prototype.

VI. ACKNOWLEDGEMENT

The reported work is developed within the Knowledge-Practices Laboratory (KP-Lab) project funded by the EU 6th R&D Framework program. We thank all our colleagues and partners involved in the design, implementation, and evaluation of the Visual Model Editor.

REFERENCES

[1] T.R. Gruber, A translation approach to portable ontologies, Knowledge Acquisition 5 (1993) 199–220.
[2] N.F. Noy and M.A. Musen, Ontology versioning in an ontology management framework, Intelligent Systems, IEEE, v. 19, pp. 6–13, 2004.
[3] Z. Duo, L. Juan-Zi and X. Bin, Web Service Annotation Using Ontology Mapping, presented at Service-Oriented System Engineering, 2005. SOSE 2005. IEEE International Workshop (2005).
[4] L. Khan, D. McLeod and E. Hovy, Retrieval effectiveness of an ontology-based model for information selection, The VLDB Journal, v.13 pp. 71–85, 2004
[5] Zhou L., Ontology learning: state of the art and open issues, *Information Technology and Management*, Vol. 8, No. 3. (2007), pp. 241-252.
[6] Gary F. Simons, Conceptual modeling versus visual modeling: a technological key to building consensus, Consensus ex Machina Joint International Conference of the Association for Literary and Linguistic Computing and the Association for Computing and the Humanities, Paris, 19-23 April 1994
http://www.sil.org/CELLAR/ach94/ach94.html
[7] www. kp-lab.org
[8] Kozaki, K., Sunagawa, E., Kitamura, Y., Mizoguchi, R. (2006). Fundamental Consideration of Role Concepts for Ontology Evaluation, 4th International EON Workshop, May 22nd, 2006, Edinburgh.
[9] D. Brickley and R. Guha, Resource description framework (RDF) schema specification 1.0, vol. 2000: W3C recommendation (2000).