Invited Talk

Metamodeling as a Concept

Dimitris Karagiannis

Department of Knowledge and Business Engineering
University of Vienna
Brünner Straße 72, A-1210 Vienna, Austria
dk@dke.univie.ac.at

Abstract

Metamodeling is a powerful concept in the area of computer science that is applied to solve a variety of tasks. This talk is going to provide a general introduction to the basic thoughts of metamodeling as well as a brief overview on existing application scenarios that can be basically distinguished in *design* and *integration*. An outlook on the combination of metamodels and ontologies for achieving semantic interoperability of graphical models concludes the talk.

Keywords: metamodeling, semantic integration, semantic interoperability

1. Introduction

Metamodels provide the means for the creation of graphical models in that they are models *"of a modelling language"* [Favre (2005)]. They define the available modeling constructs and describe their graphical representation (i.e. notation), as well as their structure and allowed combination (i.e. syntax). Metamodels can be arranged in layered metamodeling hierarchies whereas metamodels of higher layers describe the language elements of metamodels of lower layers. A meta^2-model, for instance, describes a series of metamodels.

Research work concerning metamodels can focus on three different aspects: their actual representation (e.g. logical rules, graphical representations, …), conceptional scenarios for solving concrete problems, and finally the actual implementation of these scenarios. In the following we focus on an elaboration of application scenarios.

2. Metamodels for Design and Integration

In the course of a literature survey [Karagiannis et al. (2006)] we identified two main usages of the metamodeling concept. On the one hand it is applied for *design* purposes and on the other hand to solve *integration* problems.

2.1 Design

Here the concept of metamodeling is applied for both the prescriptive definition of not yet existing as well as the descriptive modeling of already existing “objects” of interest. We distinguish two different scenarios in this context.

“Macro-level design” involves the creation of metamodels as design templates or reference frameworks in order to solve certain tasks in a specific domain. Metamodels for the formulation of business rules [Herbst (1996)] or the implementation of web-based systems [Nikolaidou et al. (2005)] can be given as examples here.

“Micro-level design” on the other hand is the use of the metamodeling concept for the realization of a kind of inheritance mechanism which implies the definition of the inner structure of data models or representation languages. This application can, for instance, be found in the definition of the knowledge representation language Telos [Mylopoulos et al. 1990].

2.2 Integration

We see *integration* as being the task of bringing together different existing “artefacts” of potentially various kinds. These artefacts are most often created corresponding to different modeling languages and, therefore, metamodels. This circumstance enforces the creation of a “translation layer” that allows for the mapping of language elements and, thus, model elements. Meta^2-models are necessary for this purpose which is, for instance, shown by [Nissen et al. (1999)] who describe an integration scenario in the area of requirements engineering.

3. Semantic Integration and Interoperability using Metamodels

Figure 1 on the next page depicts a recent approach for the improvement of metamodel-based integration and interoperability using ontologies. In a process that is called “lifting” or “ontology anchoring” elements of (meta-)models of any layer are linked to concepts of one or more ontologies which explicates the meaning of these elements. This allows for an automated processing of the (meta-)model semantics resulting in better integration and interoperability results. Due to lifting, for instance, elements of two process models can not only be related because they have been created using the same or related metamodel constructs (like e.g. “activity”) but because they involve the same actions (e.g. approving something) and/or the same resource (e.g. a proposal). For more information concerning this approach please refer to [Höfferer (2007)].

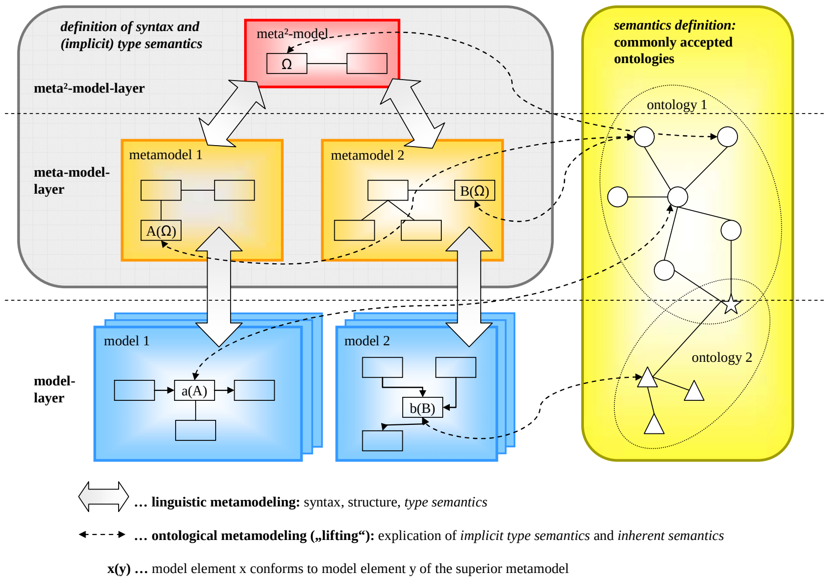
definition of syntax and (implicit) type semantics
meta²-model
Ω
meta²-model-layer
semantics definition: commonly accepted ontologies
ontology 1
metamodel 1
metamodel 2
meta-model-layer
A(Ω)
B(Ω)
model 1
model 2
model-layer
a(A)
b(B)
ontology 2
… linguistic metamodeling: syntax, structure, type semantics
… ontological metamodeling („lifting“): explication of implicit type semantics and inherent semantics
x(y) … model element x conforms to model element y of the superior metamodel

Figure 1. *Architecture for semantic integration and interoperability using metamodels and ontologies. [Höfferer (2007)]*

References

Favre, J. - M. (2005), Foundations of Meta-Pyramids: Languages vs. Metamodels - Episode II: Story of Thotus the Baboon, in J. Bézivin and R. Heckel (eds.): Language Engineering for Model-Driven Software Development, Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Dagstuhl, Germany.

Herbst, H. (1996). Business Rules in Systems Analysis: a Meta-Model and Repository System, Information Systems, 21 (2), 147-166.

Höfferer, P. (2007), Achieving Business Process Model Interoperability Using Metamodels and Ontologies, accepted for publication in Proceedings of the 15th European Conference on Information Systems (ECIS 2007)

Karagiannis, D., Höfferer, P. (2006), Metamodels in Action: An Overview, in J. Filipe, B. Shishkov and M. Helfert (eds.): ICSOFT 2006 - First International Conference on Software and Data Technologies, p. IS-27 - IS-36, Insticc Press, Setúbal.

Mylopoulos, J., Borgida, A., Jarke, M., Koubarakis, M. (1990), Telos: representing knowledge about information systems, ACM Trans. Inf. Syst., 8 (4), 325-362.

Nikolaidou, M. and Anagnostopoulos, D. (2005), A Systematic Approach for Configuring Web-Based Information Systems, Distributed and Parallel Databases, 17 (3), 267-290.

Nissen, H. W., Jarke, M. (1999), Repository Support for Multi-Perspective Requirements Engineering, Information Systems, 24 (2), 131-158.