Fişa suspiciunii de plagiat / Sheet of plagiarism's suspicion	Indexat la: 00164/00

Opera suspicionată (OS)	Opera autentică (OA)
Suspicious work	Authentic work

OS	STOICA, Eduard, A. and BRUMAR, Bogdan, A. Advanced modelling for e-learning platforms. In: *Proc. of the 11th WSEAS International Conference on Computers. Agios Nikolaos, Crete Island, Greece*. July 26-28, 2007. p.163-167.
OA	KARAGIANNIS, D. and KÜHN, H. Metamodelling platforms. In: Bauknecht, K.; Min Tjoa, A and Quirshmayer, G. (Eds.) *Proc. of the Third International Conference EC-Web 2002 – Dexa 2002*, Aix-en-Provence, France. Sept.2-6, 2002. LNCS 2455. Berlin, Heidelberg: Springer-Verlag, p.182.

Incidența minimă a suspiciunii / Minimum incidence of suspicion	
p.163:Abstract:01 - p.163:Abstract:10	p.182:Abstract:01 - p.182.182:Abstract:13
p.163:02s – p.162:18s	p.183(2):33 - p.184(3):03
p.163:Fig.1	p.184(3): Fig.2
p.163:01d – p.164:10s	p.184(3):04 – p.185(5):02
p.164:11s – p.164:44s	p.185(4):33 – p.186(5):15
p.165:40s - p.165:18d	p.187(7):24 – p.187(7):00

Fişa întocmită pentru includerea suspiciunii în Indexul Operelor Plagiate în România de la
Sheet drawn up for including the suspicion in the Index of Plagiarized Works in Romania at
www.plagiate.ro

Notă:	p.72:00 semnifică textul de la pag.72 până la finele paginii.	**Notes**:	p.72:00 means the text of page 72 till the end of the page.

Metamodelling Platforms

Invited Paper

Appeared in:

Karagiannis, D.; Kühn, H.: Metamodelling Platforms. In: Bauknecht, K.; Min Tjoa, A.; Quirchmayer, G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France, September 2-6, 2002, LNCS 2455, Springer-Verlag, Berlin, Heidelberg, p. 182.

Dimitris Karagiannis
University of Vienna
Institute for Computer Science and
Business Informatics
Department Knowledge Engineering
Brünner Str. 72
A – 1210 Vienna
Austria
dk@dke.univie.ac.at
http://www.dke.univie.ac.at

Harald Kühn
University of Vienna
Institute for Computer Science and
Business Informatics
Department Knowledge Engineering
Brünner Str. 72
A – 1210 Vienna
Austria
hkuehn@dke.univie.ac.at
http://www.dke.univie.ac.at

Full version of: Karagiannis, D.; Kühn, H.: Metamodelling Platforms. LNCS 2455, Springer-Verlag, 2002, p. 182.

Metamodelling Platforms

Dimitris Karagiannis and Harald Kühn

University of Vienna, Department Knowledge Engineering, Brünnerstr. 72,
A-1210 Vienna, Austria
{dk, hkuehn}@dke.univie.ac.at

Baumann + Popa
Stoica + Baumann

Abstract. The state-of-the-art in the area of modelling of organisations is based on fixed metamodels. Due to rapid changing business requirements the complexity in developing applications which deliver business solutions is continually growing. To manage this complexity, environments providing flexible metamodelling capabilities instead of fixed metamodels has shown to be helpful. The main characteristic of such environments is that the formalism of modelling - the metamodel - can be freely defined and therefore be adapted to the problem under consideration. This paper gives an introduction into metamodelling concepts and presents a generic architecture for metamodelling platforms. Three best practice examples from industry projects applying metamodelling concepts in the area of business process modelling for e-business, e-learning, and knowledge management are presented. Finally, an outlook to future developments and research directions in the area of metamodelling is given.

1 Introduction

Baumann + Popa

Due to rapid changing business requirements such as faster time to market, shorter product lifecycles, increased interdependencies between business partners, and tighter integration of the underlying information systems, the complexity in developing applications which deliver business solutions is continually growing. Therefore, the elements of an enterprise are managed more and more model-based.

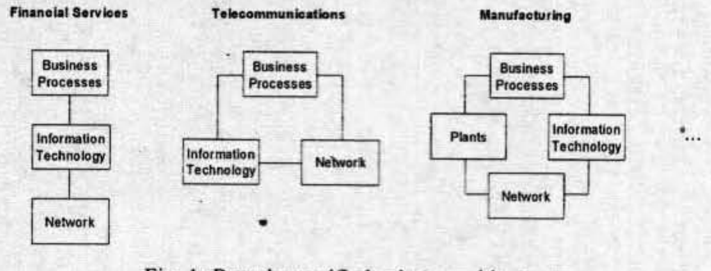

Fig. 1: Branch-specific business architectures

The *state-of-the-art* in the area of modelling of organisations is based on *fixed metamodels*. Product models are created by using product modelling environments, process models are created in business process modelling tools and organisational models are realised in personnel management tools. Web service models link these business models to information technology. They are created by using standardised languages and common ontologies. Information technology is modelled in tools supporting notions such as workflow or object-orientation. The models of the company's strategy, goals and the appropriate measurements are described and monitored by using tools supporting management concepts such as Balanced Scorecard.

Bremer + Popp

Additionally, business architectures depend highly on the branches under consideration. E.g. as the network is a supporting element for doing business in financial services or manufacturing, in the telecommunication industries the network is the essential part of the business model (see figure 1). Branch specific solutions can be seen for example in the enterprise resource planning market, where all major manufacturers offer solutions for different lines of businesses. This causes additional requirements for modelling platforms, such as *integration* mechanisms for different views and aspects under consideration. Other major requirements to an enterprise modelling platform are *flexibility, adaptability, and openness*, to integrate models based on different modelling paradigms such as decision support models, descriptive models, or predictive models. These requirements have to be fulfilled by environments providing flexible metamodelling capabilities. The main characteristic of such environments is that the formalism of modelling - the metamodel - can be freely defined. Platforms based on metamodelling concepts should support the following topics:

1. Engineering the business models & their web services
2. Designing and realizing the corresponding information technology
3. Evaluating the used corporation resources and assets

This raises research issues on how to design, manage, distribute and use flexible metamodels on a syntactic as well as on a semantic level and how to integrate, run and maintain a metamodelling platform in a corporation's environment.

The remainder of the paper is organised as follows. Chapter 2 gives an introduction to general metamodelling concepts. In chapter 3 technologies for metamodelling are presented. In chapter 4 examples of metamodelling in the areas of business process modelling for e-business, e-learning, and knowledge management, are given. Finally, chapter 5 gives an outlook to future developments and research directions.

2 Metamodelling Concepts

Strahl + Bremer

Modelling methods consists of two components: a modelling technique, which is divided in a modelling language and a modelling procedure, and mechanisms & algorithms (shorten: mechanisms) working on the models described by the modelling language (see figure 2). The *modelling language* contains the elements, with which a model can be described. A modelling language itself is described by its syntax, semantics, and notation. The *modelling procedure* describes the steps applying the modelling language to create results, i.e. models. In this paper we define a metamodel as a model of a modelling language. Applying language theory for levelling languages, the result is a hierarchy of languages, meta-languages etc. The hierarchy of the corresponding models, metamodels etc. is described in section 2.1. Section 2.2 gives a short overview of the definition of syntax, semantics and notation of modelling languages and section 2.3 describes different roles in metamodelling.

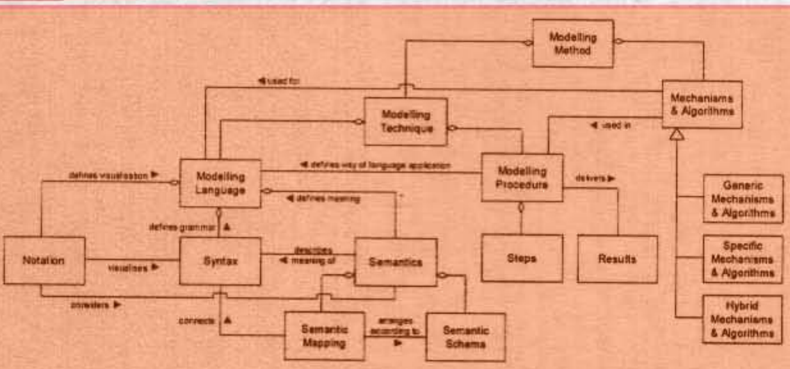

Fig. 2: Components of modelling methods

2.1 Modelling Hierarchy

The creation of a metamodel is also done by using a modelling language. This modelling language is called the *metamodelling language*. The model defining the metamodelling language is the meta-metamodel or *meta²-model* [8, 23].

Strecker + Baumann

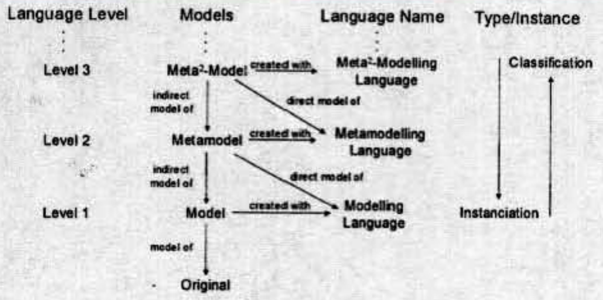

Fig. 3: Metamodelling based on language levels

Building language levels is not limited to a certain level. To "finish" the modelling hierarchy, it is important to find a useful level of abstraction. To use concepts such as "thing", "property" and "relation" may be helpful, but lack of semantics especially if the language of the "finishing" level should provide the foundation for implementing the lower levels. In practice a four layer metamodel architecture is widely used such as shown in figure 3 [e.g. 5, 9, 15, 18, 19]. The lowest level is the original, from which a model is build on the second level. Often the lowest level is seen as runtime data, but we prefer to use the expression "original" because its not always runtime data from which a model is build. The highest level in the four layer architecture is the meta^2-level, which describes the concepts for building metamodels.

2.2 Syntax, Semantics, Notation, and Mechanisms

A (graphical) modelling language is described by its syntax, semantics, and notation.

The *syntax* describes the elements and rules for creating models and is described by a grammar. For modelling languages two major approaches exist to describe their syntax: graph grammars [21] or metamodels [8]. Often, UML class diagrams are used to describe the metamodel of the syntax. For syntactical rules, which cannot be fully expressed by class diagrams, additional constraint languages are used such as OCL [17, p. 6-1ff] or AdoScript [4, p. 589ff].

The *semantics* describes the meaning of a modelling language and consists of a semantic domain and the semantic mapping. The semantic domain describes the meaning by using ontologies, mathematical expressions etc. The semantic mapping connects the syntactical constructs with their meaning defined in the semantic domain ("semantic schema"). To formulate semantic definitions approaches such as denotational semantics, operational semantics, axiomatic semantics or algebraic semantics are used [8]. Sometimes, only (informal) textual descriptions are used to define the semantics, e.g. in the definition of the UML [17, p. xxviii].

Bernar + Popa

The *notation* describes the visualisation of a modelling language. Static approaches define the symbols for visualizing the syntactical constructs e.g. using pixel-based graphics or vector graphics, but they do not consider the state of the modelling constructs during modelling. Dynamic approaches consider the model state by splitting the notation in a representation part and a control part. The representation part maps to the static approach. The control part defines rules to query the model state and to influence the representation depending on the model state [4, p. 105ff].

Mechanisms provide the functionality to use and evaluate the models built by using the modelling language. Mechanisms can be classified into generic, specific, and hybrid. *Generic mechanisms* are implemented on the meta^2-model, so they can be used for all metamodels based on the meta^2-model. *Specific mechanisms* are implemented for a particular metamodel. *Hybrid mechanisms* are implemented on the meta^2-model, but are adapted to particular metamodels, e.g. to improve usability [15, p. 85f].

2.3 Roles in Metamodelling

Stoica + Brumar

Brumar + Popa

Considering the elements of a modelling method described in figure 2 and the components of metamodelling platforms shown in figure 4, different roles in administering and using such platforms can be distinguished.

The *method engineer* is responsible for a consistent and properly defined modelling method. Additional to his technical skills, the method engineer often has professional skills in an application domain. Application domains can be divided into verticals such as financial services, telecommunications, public administration, and manufactur-

Stoica + Bauman

Bauman + Popa

ing and horizontals such as business process modelling, application development, workflow management, and knowledge management.

The *language engineer* defines the modelling language. He is responsible for an adequate definition of the syntax, semantics, and notation.

The *process engineer* is responsible for the definition of the modelling procedure. Often the process engineer is an expert in applying modelling languages and has considerable experiences in project management and project execution.

The *tool engineer* configures the mechanisms of a metamodelling platform for particular metamodels. If additional mechanisms are needed, he is the responsible for implementing these mechanisms.

The *infrastructure engineer* provides the necessary IT infrastructure to run a metamodelling platform and to integrate the platform into existing infrastructures.

The *method user* applies the modelling method by using the platform. He creates models by using the modelling language, following the modelling procedure and applying the available mechanisms.

3 Metamodelling Technologies

Section 3.1 presents a generic architecture for metamodelling platforms. In section 3.2 a brief overview of existing metamodelling approaches and platforms is given. Section 3.3 describes the BPMS lifecycle as a framework for metamodelling.

3.1 Metamodelling Architecture

Bauman + Popa

To support the topics mentioned in chapter 1, metamodelling platforms should be realised on a component-based, distributable, and scalable architecture. Figure 4 shows a generic architecture for metamodelling platforms.

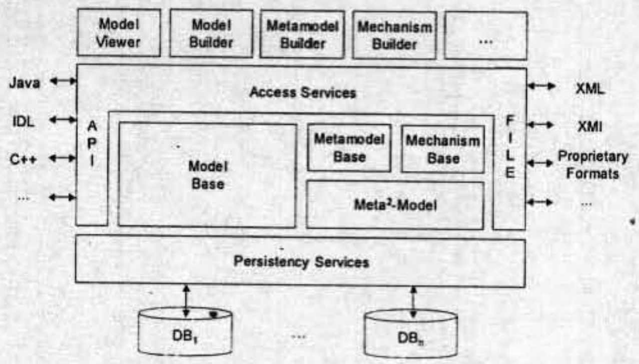

Fig. 4: Generic architecture of metamodelling platforms

The storage of all model and metamodel information is managed by *persistency services*. These services provide transparency of concrete storage types such as spe-