| Fişa suspiciunii de plagiat / Sheet of plagiarism's suspicion | Indexat la: 00165/00 |
|---|---|

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

| | |
|---|---|
| OS | BRUMAR, A. Bogdan and POPA, M. Emil. Advanced techniques for metamodelling. In: *Proc. of the 11th WSEAS International Conference on Computers. Agios Nikolaos, Crete Island, Greece.* July 26-28, 2007. p.258-262. |
| 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.258.Abstract: 01- p.258.Abstract: 09 | p.182(1).Abstract:02 - p.182(1).Abstract:13 |
| p.258:01s – p.259:11d | p.182(2):01 – p.182(2): 27 |
| p.258: Fig.1 | p.182(1): Fig.1 |
| p.259: Fig.2 | p.182(5): Fig.5 |
| p.259:12s – p.259: 20d | p.182(5):20 – p.182(6):24 |
| p.260:01d – p.260: 30d | p.182(6):25 – p.182(6):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

**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

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.

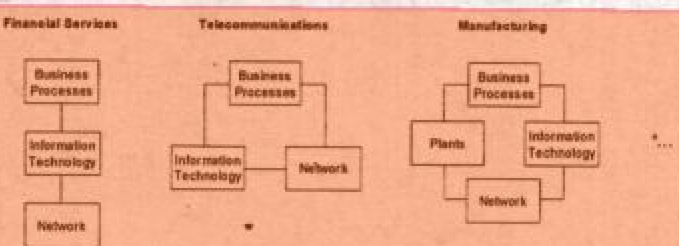


Fig. 1: Branch-specific business architectures

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.

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

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 corre-

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

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

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.

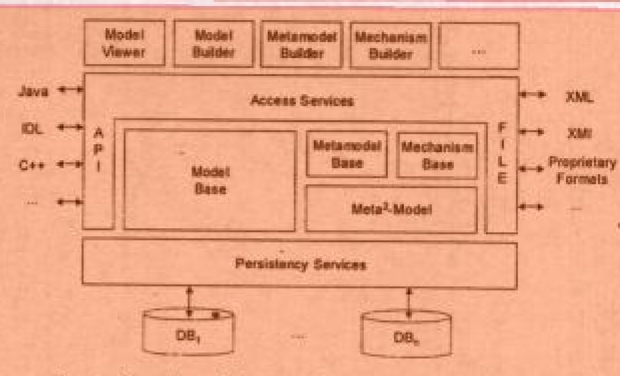


Fig. 4: Generic architecture of metamodelling platforms

The storage of all model and metamodel information is managed by *persistency services*. These services provide transparency of concrete storage types such as specific databases, files systems etc. Furthermore the persistency services enable the distribution of parts of models and metamodels.

The *meta²-model* provides the basic concepts to create metamodels and mechanisms. Typical concepts are "classes", "relations", "attributes", "modeltypes", "scripts" etc. The meta²-model is the central part of the architecture, as it provides the conceptual foundation and is connected with all other parts.

The *metamodel base* contains all information about the metamodels currently managed by the modeling platform. Changes in the metamodel base are delegated to the model base accordingly, to keep the models and their corresponding metamodels consistent.

The *mechanism base* contains information about functionalities to be applied to models and metamodels. These functionalities can be either stored directly in the mechanism base or outside of the metamodelling platform. If they are stored outside, the mechanism base holds only information how to find the appropriate mechanisms e.g. by using external name services.

The *model base* contains all models based on the metamodels. The model base communicates with the metamodel base to track metamodel changes and to forward them to the corresponding models.

*Access services* provide file-based and online interfaces to the different types of bases. According to access rights the appropriate information from the bases can be queried or even changed.

On top of the access services, different *viewer and builder components* support the usage and maintenance of the metamodelling platform such as model builder, metamodel builder, and mechanism builder.

### 3.2 Metamodelling Approaches

There exist various metamodelling approaches, different in richness of concepts and ranging from conceptual proposals to already implemented products. In the following, some of them will be illustrated briefly.

*Atkinson* proposes a modelling hierarchy aligned with the MOF hierarchy [1]. The focus is modelling in the area of distributed object systems. Atkinson stresses the dichotomy of "class" and "instance" which occurs changing the language level and proposes requirements for metamodelling approaches.

Frank proposes within his *MEMO* approach ("multi perspective enterprise modeling") a three level modelling hierarchy. Based on this hierarchy a modelling framework with the same named is suggested [6].

The *Resource Description Framework (RDF)* provides a modelling hierarchy for semantic networks. The foundation of RDF is build by three object types ("resource", "property" and "statement") for representing named properties and property values [16].

The *CASE Data Interchange Format (CDIF)* is based on a four level model architecture [5]. CDIF is a standard designed for the exchange of CASE models between tools of different tool providers. CDIF is not be further developed but major parts of