Bringing Ontology Awareness into Model Driven Engineering Platforms[1]

Srdjan Živković, Marion Murzek, Harald Kühn

BOC Information Systems GmbH, Wipplingerstrasse 1,
1010 Vienna, Austria
{srdjan.zivkovic, marion.murzek, harald.kuehn}@boc-eu.com

Abstract. In state-of-the-art of MDE platforms semantic technologies such as ontologies are rarely used. Our aim is to understand the role of ontologies in supporting model-driven engineering, in particular MDE platforms. MDE platforms may benefit from semantic technologies in formal model semantics and automated reasoning on different levels of the metamodelling architecture. We present an ontology-aware MDE platform architecture and outline some application scenarios where ontologies and automatic reasoning may bring benefit to such platforms. Additionally, an example of using ontologies for verification checks of mapping models in the course of metamodel composition is illustrated.

Keywords: metamodelling, model-driven engineering (MDE), ontology-aware architecture, ontology application scenarios.

1 Introduction

In state-of-the-art MDE platforms semantic technologies such as ontologies are rarely used. Currently, such platforms focus on aspects such as metamodelling, metamodel composition, model integration, and model transformation [1]. We are convinced that MDE platforms may benefit from semantic technologies particularly in formalizing semantics of models on different metamodelling levels, which in turn allows for application of automated reasoning.

Based on this hypothesis, in this research-in-progress paper, we discuss first how a MDE platform architecture may be extended to be considered as ontology-aware (section 2). Referring to the introduced architecture, we describe some possible application scenarios where ontologies and automatic reasoning may bring benefit to such platforms (section 3). Out of these scenarios, we highlight in more detail an example from the metamodel composition domain, to illustrate how an ontology-based approach may enhance quality of process by supporting verification checks of

[1] *Acknowledgement: This research has been co-funded by the European Commission and by the Swiss Federal Office for Education and Science within the 7th Framework Programme project MOST N° 216691, http://most-project.eu.*

metamodel mappings (section 4). Finally, we summarize the discussion and give outlook for future work.

2 Architecture of Ontology-aware MDE Platforms

Metamodelling platforms are software environments providing means for the management of models and metamodels. Usually, such model-aware platforms allow for: (1) definition, storage and maintenance of models and modelling languages, (2) execution of mechanisms working on models, metamodels and the meta-metamodel and (3) guidance on how to apply a metamodelling language and/or modelling languages together with corresponding mechanisms to produce metamodels and/or models [2]. Besides these capabilities, metamodelling platforms need to meet other functional and non-functional requirements such as multi-productability, web-enablement, multi-client ability, adaptability, extensibility, scalability and interoperability [3].
The architecture for MDE platforms may be seen as an incarnation of the generic metamodelling platform architecture [2]. Furthermore, adding the ontology aspect, we envision an enriched MDE platform architecture including semantic technologies as depicted in fig. 1.

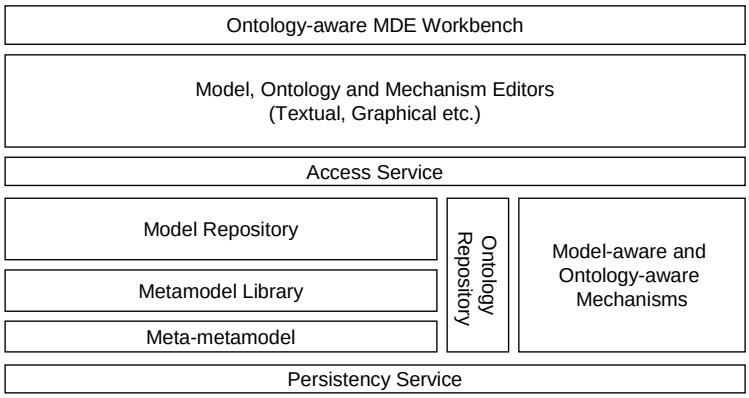

Fig. 1. Logical Architecture for Ontology-aware Model Driven Engineering Platforms

The root core architectural element is the meta-metamodel which defines the concepts available for the definition of modelling languages. Based on it, the metamodel library contains metamodels of defined modelling languages. The metamodel library conforms to the meta-metamodel and, in turn, forms the foundation of the model repository, in which all models are stored.
As an extension to models on different levels, the ontology repository serves as storage of the semantics of models, metamodels and the meta-metamodel. Semantics can be formally described by using the notion of ontology [4]. Reasoning on ontologies is part of the ontology-aware mechanisms.

Model and ontology editors are used for the definition and maintenance of models, metamodels, and ontologies.
All mechanisms used for evaluating and using models are stored in the mechanism base. A fundamental mechanism within MDE environments is model transformation. Further important mechanisms are model/metamodel integration, comparison and mapping mechanisms. The ability to manage different versions of models and metamodels or its parts is another key characteristic of model-aware systems, which should be enabled by means of model/metamodel versioning mechanisms. To support syntactically and semantically correct modelling, validation mechanisms on models and metamodels are used. Furthermore, querying mechanisms of ontology-enriched models and metamodels are needed features to allow various model analyses. Traceability of transformations, which should support guiding the software development tasks, is another needed mechanism within MDE platforms. Mechanism editors are used for definition, configuration and maintenance of mechanisms.
Guidance describes the application of modelling and metamodelling languages and mechanisms. Particularly in context of MDE, this can include guidance on what to consider when defining certain models or guidance on the decision what the next step in a particular model-driven software development process would be. Guidance information can be stored in process models or ontologies, and/or extracted out of existing modelling artefacts and/or traceability links.
Persistency services support the durable storage of models, metamodels, and ontologies. These services abstract from concrete storage techniques and permit storing of modelling information in heterogeneous data sources such as files, databases or web services.
Access services serve two main tasks. On the one hand they enable the open, bi-directional exchange of all metamodel, model and ontology information. On the other hand they cover all aspects concerning security such as access rights, authorization, and en-/decryption.
An ontology-aware MDE workbench serves as a common environment for integrating different editors.

3. Some Ontology Application Scenarios in MDE Platforms

The “ontology” concept, as the literature describes it, seems to enjoy as many definitions as there are attempts to define it. In the course of this paper, we will favour the one we believe to best match the context under consideration. Hence, we understand an ontology as an *explicit conceptual model extended by formal logic based semantics* [5]. Formal semantics expressiveness of ontological models is a de facto advantage compared to conceptual models in software engineering i.e. MDE. Model checking, model enrichment and dynamic classification are some of the identified usage scenarios when thinking about marrying ontological and metamodelling technical spaces [6].
In the following, we describe some scenarios, where ontologies may find its usage within MDE platforms. We concentrate particularly on the following core elements of the MDE platform (see section 2): the model, metamodel and meta-metamodel

element (section 3.1), the mechanism element (section 3.2) and the guidance element (section 3.3). Each section starts by introducing a problem domain, followed by an ontology application scenario description and finalized with a list of possible benefits gained by using semantic technology.

3.1 Ontologies and Models on M1, M2 and M3 Level

Problem Domain: The 3-layer MDE architecture enables definition of modelling languages, and based upon it, creation of models. The M3 model, i.e. the meta-metamodel, defines the syntax of the metamodelling language which is used for modelling metamodels on the M2 layer. Similarly, the M2 model, the metamodel, constructs the syntax of the modelling language for the application domain used on level M1. Even though the abstract syntax is structurally well defined, metamodelling language (M3) and modelling languages (M2) lack well-defined semantics. Currently, language semantics on M3 and M2 level may be expressed algorithmically in the core implementation of the M3 model or in the M2 models. Another approach is to use a declarative language, such as the Object Constraint Language (OCL [7]) to additionally define the semantics of M3 and M2 models.
Furthermore, modelling task on M1 level requires adequate knowledge about the subject under consideration. Such knowledge may currently be captured and reused via reference models or patterns. However, more sophisticated mechanisms to facilitate modelling task semantically are missing, which would prevent heterogeneity problems, model ambiguity etc.
Ontology Application Scenario: The syntax-rich languages (M2 and M3 languages) in MDE platforms may be extended by semantically more expressive ontology languages, in order to take advantage of automated reasoning. There are different approaches, which tackle the problem of converging languages from different technical spaces, such as UML+OWL integration [8]. Having integrated languages, their synergetic effect may be exploited. On the one side, automated reasoning may be used for model consistency checks which may outperform existing solutions in terms of semantic soundness. On the other side, ontology-based approaches may be used both for the definition of modelling languages (M2) and their models (M1). Relying on common domain ontologies in form of machine-readable and reusable domain knowledge, quality of modelled solutions may be raised.
Gained Benefit: Semantically enriched modelling languages and models; machine-readable formal semantics of models; enhanced quality of modelling solutions.

3.2 Ontologies and Mechanisms

Problem Domain: Mechanisms are applied on models residing on different abstraction levels (M1, M2 M3 models). According to the abstraction level, mechanisms may be generic, metamodel specific or hybrid. *Generic mechanisms* are defined on the M3 level, thus being independent of languages defined on the M2 level; e.g. generic import/export interface for model exchange. *Metamodel specific mechanisms* require explicit knowledge about metamodels, in order to work on their

underlying models on M1 level; e.g. business process model simulation mechanism. *Hybrid mechanisms* are a combination of the previous two. They are generic, but they are adaptable to specific metamodels; e.g., model transformation is a hybrid mechanism which uses M3 level constructs to define transformation rules between M2 models, which are, in turn, executed on models on M1 level. Other MDE mechanisms such as model integration, metamodel composition or model comparison fall into this category as well. The challenge of applying metamodel specific or hybrid mechanisms lies in their configuration. For example, to enable a simulation on a specific process language, mappings to generic process concepts need to be defined. Similarly, the specification of metamodel mappings for hybrid mechanisms such as transformation, metamodel composition or model integration is, in most cases, performed manually (see section 4 for a detailed example).
Ontology Application Scenario: Ontologies may be applied to fill the formal semantic gap towards support of automated configuration of metamodel-specific and hybrid mechanisms. For instance, in the case of model transformations, this would mean e.g. an ontology based model transformation approach. On the other side, metamodel specific mechanisms such as simulation would benefit from ontologies, by relying on e.g. generic, machine-readable process ontology. The prerequisite is that different simulation-enabled languages, i.e. process modelling languages have to conform to particular generic process ontology. Consequently, the process of configuring a simulation mechanism for different process languages may be automated by inferring mappings out of ontology.
Gained benefit: Reduced costs through automated configuration of mechanisms; increased potential for reuse; low efforts for substitution and extensions of mechanisms.

3.3 Ontologies and Guidance

Problem Domain: Guidance in the MDE context may include information on what to consider when defining certain models, or information of the next step in the model-driven software development process. Often, execution of a step within the software process is influenced by many factors, such as pre and post-conditions, defined rules etc.
Ontology Application Scenario: Ontologies and automated reasoning may leverage execution of process models, by formalizing its semantics in the form of *process guidance ontologies* that formalize rules, conditions and actions a software engineer has to conduct in specific situations. This way, reasoning technology would infer the next step within the process based on the process guidance ontology and by deriving implicit knowledge from corresponding modelling artefacts.
Gained Benefit: Flexibility of process definitions; enhanced quality of guidance.

4 An Example: “Verification of Mapping Models”

Mappings define correspondences between elements of different models. Particularly, metamodel mappings have an important role in the MDE approach by building

semantic bridges between different metamodels. They add knowledge about integrative usage of different modelling languages, still leaving the integrating languages independent. Bridging of metamodelling and ontology languages is done via mappings [10]. Rules for MDA based model transformations may be built based on existing mappings [11][12]. Furthermore, mappings are used as input for metamodel composition rules by stating about structural-semantic relationships of metamodel elements from different metamodels [13]. However, two problems arise when the mappings are applied: First, discovery of mappings by means of metamodel matching is a complex task, being a tedious work when executed manually and a challenging task for semi-automatic identification [14]. Second, mappings are managed as models and are built based on a mapping language. Thus, mappings need to be verified against their syntax and defined semantics. This may imply not only checking the mapping model, but also crossing the model boarder and diving into the semantics of metamodels being integrated, in order to verify certain mapping statements against e.g. cyclic generalization relationships, multiple inheritances, redundancy etc.

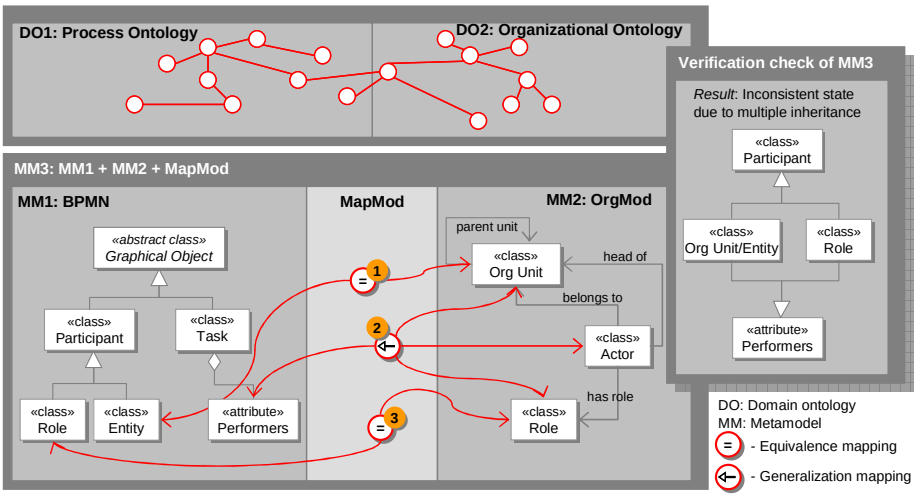

Figure 2 Ontology-aware Modelling and Verification of Mappings for Metamodel Composition

Figure 2 illustrates the simplified case of a metamodel composition using mappings. Let us assume that the MDE platform supports two modelling languages, e.g. BPMN [15] and the ADONIS Language for Organizational Modelling [16], separately. The envisioned integrated metamodel (MM3) should exhibit characteristics of a business process modelling language extended by the concepts for modelling organizational structures (in Fig. 2 as MM1 and/or MM2). A proposed approach is to assemble existing metamodels by utilizing metamodel mappings. The manual process of metamodel matching results in three identified mappings (Fig. 2, 1, 2 and 3), which are candidates for integration points. At first glance, everything seems correct, as the defined mappings conform to the mapping language which uniquely state correlations between elements. However, after generating the integrated metamodel (MM3) (see [13] for detailed integration rule definitions) a verification check founds an

unconformity against the meta-metamodel, which disallows multiple inheritance of elements. The drawback of the solution is, that the verification of a mapping model may only be done after having generated the integrated metamodel, since such cross-model correlations may not be examined in the mapping design time. Here, ontologies may be used guiding both metamodelling and modelling of metamodel mappings. If metamodels are designed with the support of corresponding ontologies, as depicted in the Figure 2, not only the metamodel matching process may be enhanced, but also an early verification of a mapping model in design time against integrated semantics stemming from both problem domains may be possible. In addition, large scale metamodel integration scenarios may especially benefit from the ontology based composition approach reducing ambiguities and improving quality of modelling solutions.

5 Summary and Outlook

In this research-in-progress paper we presented a possible extension of the generic architecture for MDE platforms [3] towards an ontology-aware MDE platform. Based on this architecture and its core elements, possible ontology application scenarios have been discussed. Some of their expected benefits are:

- Semantic-enriched models (on M1, M2, and M3 level).
- Machine-readable formal semantics of models.
- Semi-automated configuration of mechanisms.
- Increased potential for reuse of mechanisms.
- Flexibility of process definitions for guidance.

We are currently working on the refinement of the presented ontology-aware MDE platform architecture to support first prototype implementations. This includes the application of ontologies for enhanced software process guidance.

6 References

1. Bezivin, J., Jouault, F., and Touzet, D. 2005. Principles, Standards and Tools for Model Engineering. In Proceedings of the 10th IEEE international Conference on Engineering of Complex Computer Systems (June 16 - 20, 2005).
2. D. Karagiannis; H. Kühn: Metamodelling Platforms. Invited paper in: Bauknecht, K.; Tjoa, A Min.; 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.
3. H. Kühn, M. Murzek: Interoperability Issues in Metamodelling Platforms. In: Konstantas, D; Bourrières, J.-P.; Léonard, M.; Boudjlida, N. (Eds.): Proceedings of the 1st International Conference on Interoperability of Enterprise Software and Applications (I-ESA'05), Geneva, Switzerland, February 2005, Springer Verlag, pp. 215-226.
4. Gruber, T. R.: Toward principles for the design of ontologies used for knowledge sharing. In: Guarino, N.; Poli, R. (Eds.): Proceedings of the International Workshop of Formal Ontology, Padova, Italien, August 1993.

5. Oberle, D.: Semantic Management of Middleware, Volume I of The Semantic Web and Beyond Springer, New York (2006).
6. Fernando Silva Parreiras, Steffen Staab, Andreas Winter: On Marrying Ontological and Metamodeling Technical Spaces. 2007. ACM Press. Proceedings of the 6th joint meeting of ESEC/FSE, 2007, Dubrovnik, Croatia, September 3-7.
7. OCL – Object Constraint Language, http://www.omg.org/. December 2007.
8. Kiko, K. and Atkinson, C.: Integrating Enterprise Information Representation Languages. In: Proc. of Int. VORTE Workshop, VORTE 2005, Enschede, The Netherlands, 2005.
9. S. Roser and B. Bauer. An approach to automatically generated model transformations using ontology engineering space. In Proceedings of Workshop on Semantic Web Enabled Software Engineering (SWESE), Athens, GA, U.S.A., 2006.
10. Fernando Silva Parreiras, Steffen Staab, Andreas Winter: TwoUse: Integrating UML Models and OWL Ontologies. Universität Koblenz-Landau, Fachbereich Informatik. 2007. Arbeitsberichte aus dem Fachbereich Informatik.
11. Lopes, D, Hammoudi, S, Bézivin, J, and Jouault, F : Mapping Specification in MDA: from Theory to Practice. In: Proceedings of the First International Conference on Interoperability of Enterprise Software and Applications (INTEROP-ESA'2005). Springer-Verlag, pages 253—264. 2005.
12. G. Kappel, E. Kapsammer, H. Kargl, G. Kramler, T. Reiter, W. Retschitzegger, W. Schwinger, and M. Wimmer. Lifting metamodels to ontologies: A step to the semantic integration of modeling languages. In Proc. of MoDELS? 2006, volume 4199 of LNCS, pages 528-□542. Springer, 2006.
13. Zivkovic, S.; Kühn, H.; Karagiannis, D.: Facilitate Modelling Using Method Integration: An Approach Using Mappings and Integration Rules. In: Österle, H.; Schelp, J.; Winter, R.; (Eds.): Proceedings of the 15th European Conference on Information Systems (ECIS2007) - "Relevant rigour - Rigorous relevance", St.Gallen, Switzerland. June 2007, pp. 2038-2050.
14. Kappel, G., Kargl H., Kramler G., Schauerhuber A., Seidl M., Strommer M., Wimmer M., Matching Metamodels with Semantic Systems - An Experience Report, BTW 2007 Workshop Model Management und Metadaten-Verwaltung, Aachen; March 2007.
15. BPMN – Business Process Modelling Notation, http://www.omg.org/spec/BPMN/1.1/. January 2008.
16. BPMS Method Handbook, ADONIS 3.9 Manuals, BOC Group, 2007.