Generation of Language-Specific Transformation Rules based on Metamodels

Konrad Voigt

SAP Research CEC Dresden, Chemnitzerstr. 48, 01187 Dresden, Germany
konrad.voigt@sap.com

Abstract. Model Transformations are the core of Model Driven Development. They are used in many areas (e.g. to bridge different levels of abstractions, synchronize inter-model commonalities or realize model evolution) and supported by well-established standards such as Query, View and Transformation (QVT). Nowadays, transformations have to be defined manually which is an error-prone, complex, and time-consuming task. In this paper we address the problem of manual definition and describe an existing approach for an automatic mapping of metamodels and generation of rules for transformation languages. We target the use of the generated rules in order to provide guidance and support during the definition process. However, this novel approach lacks of usability, completeness and quality. Therefore, we identify the need for improving this approach and outline how automatic mapping concepts apply to the area of Service Engineering in context of the Internet of Services (IoS).

Key words: Model transformation, model matching, transformation generation.

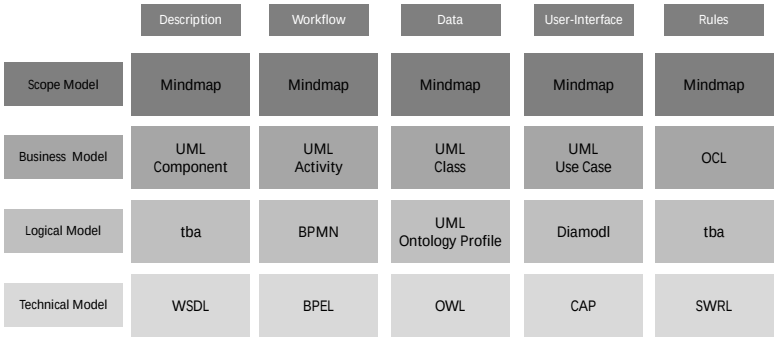
1 Research Problem

The *Internet of Services* (IoS) envisions a common network infrastructure connecting people, goods and services. Therefore, the IoS combines approaches such as Service Description Framework, Service Level Agreement, Service Oriented Architecture (SOA), etc. The main concept in SOA is an interaction of loosely coupled building blocks; so-called Services. Baida et al. [2] define among other things an e-Service consisting of a business model, a business process, and a technical process. Consequently, we see the description of a Service as a set of different models with different levels of abstraction.

These different models have to be handled and integrated. One approach for this problem is to use a Model Driven Architecture (MDA) approach to provide standards for model definition and handling. We propose to use MDA concepts to define a services models. Each model is concerned with a specific dimension of a Service such as: description, workflow, data, user interface, and rules. We choose these dimensions based on the Zachman-Framework [11] adapting them to an SOA-environment. We also use the separation of each dimension into four levels of abstraction. The resulting concept can be seen as a matrix where a column

represents a specific dimension and each row represents a level of abstraction. Figure 1 depicts this matrix whereas each cell has a specific model assigned to it. This matrix allows to follow a divide-and-conquer approach and provides a structured view on the models describing a Service since each model is represented in one cell.

Fig. 1. The Structure of Models of the Inter-Enterprise Service Engineering (ISE) Framework



The resulting framework is composed of twenty models which requests for model transformations. Inter-model commonalities as well as the representations of artifacts on different levels of abstraction have to be considered. An example is the models of the workflow column. In our approach we propose to describe processes in simplified (e.g. without the modeling of data flow) UML-Activity diagrams, then to refine them in a BPMN diagram and finally further to enrich them within BPEL. Each of these representations contains the same artifacts on different levels of abstraction; in order to implement the mapping between the different levels a model-to-model transformation is needed. Another example is the need for synchronization between data artifacts defined in the data column, data structure defined for user interaction (user-interface column) and data artifacts defined in BPMN for modeling a data flow. Again a model-to-model transformation is needed to implement the mapping between the model elements mentioned.

Following the MDA-Guide [9] a model-to-model transformation is defined as: "the process of converting one model to another model of the same system". We will refer to model transformations as model-to-model transformations. The area of model transformation has seen significant development during the last years. Czarnecki and Helsen investigated and classified in 2003 different languages for model transformations in a feature-based approach [4]. In 2005 the OMG released the specification of *Query, View and Transformation* (QVT) [10] as a standard for metamodel- and rule-based model transformation.

Nowadays, many implementations of QVT-engines allow model transformation via an interpretation of a transformation rule model. Furthermore suitable tools have been developed supporting the definition of model transformations with well-known concepts like: context-sensitive content assists, error-detection and highlighting, and debugging capabilities. Nevertheless, like in the area of software engineering [3], models inherit a certain complexity from their problem space which also applies to model transformations. Therefore, a transformation definition is a complex and challenging task. Imagine the mapping of metamodels like UML, BPMN or XML-Schema with forty and more elements, along with five and more attributes per element. This leads to more than forty transformation rules (considering at least a one-to-one mapping of all elements). This is an error-prone, time-consuming and challenging task requiring expertise in the area of model transformation. To cope with this complexity the need for guidance and support arises.

2 Related Work

Fabro and Valduriez [6] are the only ones tackling this problem by proposing a semi-automatic matching of metamodels resulting in a so-called *Weaving Model*. This model represents the mapping between two models by linking corresponding model elements. Their iterative matching approach uses different matching techniques like linguistic and structural similarity which they combine with heuristics. Their approach results in a huge number of generated ATL-rules compared to manual rule definition. In one case their strategy generates 250 lines of textual ATL-rules; compared to four rules specified manually. This amount of rules shows the complexity of further manual refinement. They admit that their approach does not cover the complete mapping and outline the need for improving the generated transformation model and rules.

Besides them other approaches exists like in the areas of data and schema integration [1][5][8]. Especially the COMA++ approach seems promising for an adaption in the area of model transformation. COMA++ defines a combined matcher framework based on heuristics and matcher like instance-based and reuse-oriented. Lopes et al. mention in their proposal of a transformation language [7] the future work of applying concepts of schema matching to the area of model transformation. We intend to follow this proposal using the results of the COMA++ research.

3 Work Plan

Starting from the work of Fabro and Valduriez we intend to pursue a twofold approach:

1. An improvement of the generated rules by quantity and quality and research on a meaningful application to the process of model transformation development.

2. A development of a tool providing support and guidance for the process of transformation development.

We propose, as a first step, research on a classification of model transformations scenarios. We start our classification by proposing two categories of scenarios: (1) a feature-driven one and (2) a set/similarity-based one. The feature-driven classification aims at developing a hierarchy of typical model transformation like bridging levels of abstraction, in-place transformation for updating a model or perfective model evolution transformation. The second one provides a notion for the comparison of metamodels to be transformed based on their elements. Interpreting all elements of one metamodel as a set a distance function has to be defined comparing the metamodel element sets regarding their overlap.

Following these possible classifications, we intend to identify transformation use cases within a special domain denoted by the different classes. Improved matching concepts with respect to the identified classes can be applied following the classification. In order to improve the matching concepts we plan to adapt the schema (ontology) matching concepts given by COMA++. We want to adapt the matchers algorithm and reuse the given concepts. We expect effort regarding the adaption of the different matchers, because of the differences between the model transformation and schema matching. Furthermore we propose to include user-interaction like a clustering of models to be transformed, and learning algorithms based on user decision regarding the reuse of existing rule definition.

The resulting generic mapping model contains the correspondences between model elements according to the mapping. This model serves as a base for a model transformation into executable transformation models and rules. By combining schema matching and model transformation we expect fewer rules to be generated with a higher quality regarding the usability.

Finally we combine the results providing assistance for model transformation engineering by defining transformation rules proposals based on user interaction. We don't intend to replace the manual definition of transformation rules; instead we want to support and guide the rule-definition process to enable a structured, reuse-oriented and flawless transformation definition.

Acknowledgments

This work is done in context of a doctoral work at the chair of Software Technology of Prof. Uwe Aßmann at the Technical University of Dresden. The project was funded by means of the German Federal Ministry of Economy and Technology under the promotional reference '01MQ07012'. The author takes the responsibility for the contents.

References

1. David Aumueller, Hong-Hai Do, Sabine Massmann, and Erhard Rahm. Schema and ontology matching with COMA++. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 906–908, New York, NY, USA, 2005. ACM.

2. Ziv Baida, Jaap Gordijn, and Borys Omelayenko. A shared service terminology for online service provisioning. *ICEC '04: Proceedings of the 6th international conference on Electronic commerce*, pages 1–10, 2004.
3. Fred Brooks. No silver bullet. *IEEE Computer*, 20(4), 1987.
4. Krzysztof Czarnecki and Simon Helson. Classification of model transformation approaches. In *OOPSLA'03 Workshop on Generative Techniques in the Context of Model-Driven Architecture*, 2003.
5. Robin Dhamankar, Yoonkyong Lee, AnHai Doan, Alon Halevy, and Pedro Domingos. iMAP: discovering complex semantic matches between database schemas. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 383–394, New York, NY, USA, 2004. ACM.
6. Marcos Didonet Del Fabro and Patrick Valduriez. Semi-automatic model integration using matching transformations and weaving models. *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*, pages 963–970, 2007.
7. Denivaldo Lopes, Slimane Hammoudi, Jean Bzivin, and Frdric Jouault. Generating transformation definition from mapping specification: Application to web service platform. In *Advanced Information Systems Engineering: 17th International Conference, CAiSE 2005*, 2005.
8. Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm. Generic schema matching with cupid. In *The VLDB Journal*, pages 49–58, 2001.
9. OMG. *Model Driven Architecture Guide, Version 1.0.1.* Object Management Group, June 2003. omg/03-06-01.
10. OMG. *Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification.* Object Management Group, 2005. ptc/05-11-01.
11. John A. Zachman. A framework for information systems architecture. *IBM Syst. J.*, 26(3):276–292, 1987.