

# Semantic Modelling for Variabilities

Mikhail Roshchin

Volgograd State Technical University (Russia),  
in collaboration with Siemens AG, CT SE 2 (Germany)  
[roshchin@gmail.com](mailto:roshchin@gmail.com)

**Abstract.** The aim of our work is to present solutions and a methodical support for automated techniques and procedures in domain engineering, in particular for variability modelling. Our approach is based upon Semantic Modelling concepts, for which both, semantic description, representation patterns and inference mechanisms are defined. Thus, model-driven techniques enriched with semantics will allow flexibility and variability in representation means, reasoning power and the required analysis depth for the identification, interpretation and adaptation of artefact properties and qualities.

## 1 Problem Statement

Let us assume that we require a software system that is specifically tailored to rely on our needs; that is valid and consistent within the reality of the environment and involved domains. But the cost issue plays an important role, and the development of specific and generic products is not that cost-effective as we expect. For reduction of costs, software engineering aims of an increasing reuse by collecting and composing artefacts and assets, components and products into complex systems and new applications. Also, the ideas and concepts of families of systems and product lines are formalized for easier way of future artefact implementation.

Behind the system composition process and derivation of new product implementation based on reuse, there is a heavy and massive layer of computing model-based procedures. Therefore models are considered to be interchangeable and valid for particular task and requirements. Model-driven engineering introduces models together with techniques for system design and artefact adaptation into business process and software lifecycle. At the same time, domain engineering provides with deep understanding of the targeted domain and its specifics, and variability modelling specifies commonalities, variants and features, their relations and restrictions, for the whole product family of systems realized and presented as models.

But, due to the high diversity of modelling techniques, distinctions between models of different aspects, domain-dependent and company-specific knowledge and specifications, the reuse is still difficult and non-trivial. The lack of formal semantics for MDAs [2], domain and variability models and requirements engineering, affects with the impossibility of pragmatic and cost-effective solution for automated reasoning techniques. The absence of well-established semantic model does not allow

us to provide self-configuring techniques, consistency verification procedures and advanced selection of valid artefacts.

Domain engineering has been proved to handle a high priority share in the entire model-driven engineering, but the state of the art shows that the lack of formal semantics and proper tool support for automated reasoning have hindered the development in this area. So far, the knowledge representation techniques based on semantics are being developed in isolation from software engineering activities, in particular from feature and variability modelling. Existing semantic approaches are not aligned with the entire modelling process, and need an advanced review on conceptual level for the proper role and place of formal methods within existing software engineering streams.

No doubts, that model-driven architecture, domain engineering, variability and feature models are perfect approaches themselves. But there is an urgent need to enrich them with formal methods of knowledge representation and benefit from that in the near future [4].

Here we focus just on variability modelling, assuming that our approach can be used in a wider range, in particular for MDE and domain engineering. It is shown how semantic modelling can handle and support variability modelling, and how software engineering will benefit from that.

## **2 Approach**

The need for variability modelling and its role within the scope of domain engineering in the software development area are obvious and generally accepted. Variability modelling becomes necessary when we derive new specifications for further artefact implementation from the set of commonalities and variants related to particular system family. Also, it is important for describing dynamical behaviour of systems. We take a variability model as proposed by [5]. But still, there are open questions and issues, mentioned by different research institutes and software communities, which have hindered the expected development in the field of knowledge reuse.

### **Goal**

Our goal is to provide proper methods and tool support for formally expressing, processing and analyzing models and variants.

We need to introduce formal semantics and appropriate automated reasoning techniques. Based on that, we achieve explicit consideration of environmental, behavioural and business model aspects, interoperability of the diversity of components. Semantic modelling allows acquisition, interpretation and adaptation of different variability models into one decision process.

Our Semantic Modelling approach presented in [1] is based on two concepts, which are significant for the whole procedure and aligned with requirements to semantics. These concepts are Logic-on-Demand and Triple Semantic Model.

### **The Triple Semantic Model Concept**

Our Semantic Model is based on the principles of the Triple Semantic Model concept, which aims in defining a distributed computing model for the whole lifecycle of variability model and to provide mechanisms to distinguish between different entities represented within that model. It consists of three levels: the *Ontology Level*, the *Dynamic Annotation Level*, and the *Annotation Level*. The ontologies on the *Ontology Level* are intended to provide a general framework, in most cases based on a specific application domain, to describe any kind of product line and related information. Since ontologies enforce proper definitions of the concepts in the application domain, they also play an essential role in standardising the definitions of component or service properties [3], requirements and interfaces with respect to their domain. Ontologies hold independently from actual circumstances, the situation in the environment or the actual time. However, such dependencies from actual, dynamically changing circumstances do have an important influence in the compositional approach. Hence, rules determining how to cope with this dynamicity have to be provided if one has to include it into the reasoning. They are specified on the *Dynamic Annotation Level*: Dynamic annotations play the role of mediators between the ontology and the static semantic annotations that describes the artefact variants and features, and in particular its requirements with respect to reuse and composition. It becomes possible to express behavior variants, and options depending on dynamic features, and it enables the reasoning about particular situations and dynamically changing lifecycle conditions. The *Annotation Level* comprises the static descriptions of the properties and qualities of artefacts.

### **The Logic-on-Demand Concept**

Semantic modelling of products and families involves a large variety of information from different application domains and of various categories, like terms and definitions, behaviour rules, probability relations, and temporal properties. Thus, it seems to be the obvious to choose the most expressive logical formalism that is capable to formulate and formalise the entire needed information. But, doing so very likely results in severe decidability problems.

Our semantic modelling approach, based on the concept of Logic-on-Demand (LoD), is supposed to overcome the problems of complexity of formal semantics by accommodating the expressivity of the proposed ontology languages to the varying needs and requirements, in particular with respect to decidability. The main purpose of the LoD concept is to provide an adequate and adaptive way that is based on uniform principles for describing all the notions, relations and rules, the behaviour and anything else that proves necessary during the component or service annotation process. To achieve this, LoD means to define a basic logical formalism that is adequate and tailored to the application domain and to incorporate additional logic formalisms and description techniques with further expressivity as optional features that can be used whenever needed. These additional formalisms share notions and terms with the basic formalism which will be grounded syntactically in OWL and semantically in the description logics.

## Validation of the Approach

The validation of the approach includes three aspects:

- Evaluation of the applied formal semantics with respect to sufficiency and decidability. The work on Logic-on-Demand concept is still in progress. Our intention is to avoid complexity issues and to guarantee adequate system response time.
- Feasibility issue. So far, we implement proposed techniques in a prototype tool. It covers the whole lifecycle of semantic modelling – starting out from defining semantic patterns and domain-specific information and eventually providing fully automated composition techniques based on semantic models.
- Estimating the additional cost and time for semantic modelling according an approach. Do a creation of semantic models and an implementation of formal methods and techniques really pay off in software engineering? This question touches a most important issue of our work and will be investigated in concordance the prototype tool development.

## 4 Conclusion

Introducing a well-structured semantic modelling procedure for variability modelling provides with flexibility of representation means and methods. It allows correct (self) configuration and composition of different shares among the whole set of domain pieces during the entire modelling process, by taking into account behavioural, environmental and business aspects. Improved acquisition, interpretation and adaptation techniques allow to increase reuse among different domains and system families. Formal methods in modelling support automated derivation of an executable and sufficient model for further system or artefact implementation based on semantic mapping of requirements criteria and the given set of features and variants.

## References

1. Peter Graubmann, Mikhail Roshchin, "Semantic Annotation of Software Components", Accepted for 32th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), Component-Based Software Engineering Track, 2006
2. Jack Greenfield, Keith Short, *Software Factories*. Wiley Publishing, 2004
3. Claus Pahl, "Ontologies for Semantic Web Components," ERCIM News No 51, Oct. 2002. [http://www.ercim.org/publication/Ercim\\_News/enw51/pahl.html](http://www.ercim.org/publication/Ercim_News/enw51/pahl.html)
4. Uwe Assmann, Steffen Zschaler, Gerd Wagner, *Ontologies, Meta-Models and the Model-Driven Paradigm*
5. Stan Buehne, Kim Lauenroth, Klaus Pohl: *Modelling Requirements Variability across Product Lines*. Proceedings of the 2005 13th IEEE International Conference on Requirements Engineering (RE '05), IEEE