

# HybridMDS D - Multi-Domain Engineering with Ontological Foundations <sup>★</sup>

Henrik Lochmann and Birgit Grammel

SAP Research CEC Dresden  
Chemnitz Str. 48, 01187 Dresden, Germany  
{henrik.lochmann|birgit.grammel}@sap.com

## 1 Introduction

This poster presents the HybridMDS D project [1, 2], which promotes model-driven software development with multiple DSLs through the application of Semantic Web technologies as foundation for metamodel and model mediation.

Section 2 introduces the major context and technologies of this work. Section 3 sheds light on the challenges and problems that arise in this context. In Sect. 4 we shortly describe the HybridMDS D solution approach and architecture.

## 2 Context and Technologies

As the project title already reveals, the HybridMDS D project deals with *model-driven software development (MDS D)*. MDS D promotes modeling constructs to first class software artifacts that are used to derive actual implementations, e.g., through code generation or interpretation. The employment of multiple *domain-specific languages (DSLs)* in model-driven scenarios becomes necessary when software is sufficiently complex as it is the case with, e.g., in large enterprise systems. Through the application of multiple DSLs, a proper separation of concerns can be ensured and dedicated developers can unfold their entire potential of domain-specific expertise. Another paradigm that counts to the corner stones of HybridMDS D is *software product line engineering (SPLE)*. Here, the management of variability is a key concern, which is also common in large-scale software systems that are to be configured for different use cases. We try to improve variability management through our concept of explicit inter-model references.

## 3 Challenges and Problems

Software applications comprise various different technical parts, such as data structure descriptions, behaviour specifications, persistence technologies or user interfaces. All these parts are meshed together through certain mechanisms that

---

<sup>★</sup> This work is partly supported by the feasiPLE project partly financed by the German Ministry of Education and Research (BMBF)

reach from statically typed languages to XML descriptor files. When implementing software with multiple DSLs that abstract from according application parts, the inter-dependencies between these parts persist on modeling level but cannot anymore be resolved. There is no way to ensure referential integrity and inter-model consistency.

## 4 Solution Approach and Architecture

Within the HybridMDS approach we employ an ontology for software models [2] to build a proper connection basis between different DSLs. We capture the semantics of dedicated language constructs by mapping them to concepts and roles of our ontology. During system modeling, when employed DSLs are instantiated, this mapping information is used to create an ontology knowledge base.

This in turn, forms the basis to provide valid and explicit inter-model reference possibilities (see Fig. 1). This way, the software modeler is enabled to draw semantic sound references between different models in the solution space of an application. Because available as ontology knowledge base, these inter-model references are amenable to Semantic Web reasoners. This enables us to provide sophisticated reasoning support to

1) ensure consistency in inter-model references, 2) facilitate inter model navigation, 3) help with active guidance during software development and 4) propose model adaptations in case of feature-driven modifications.

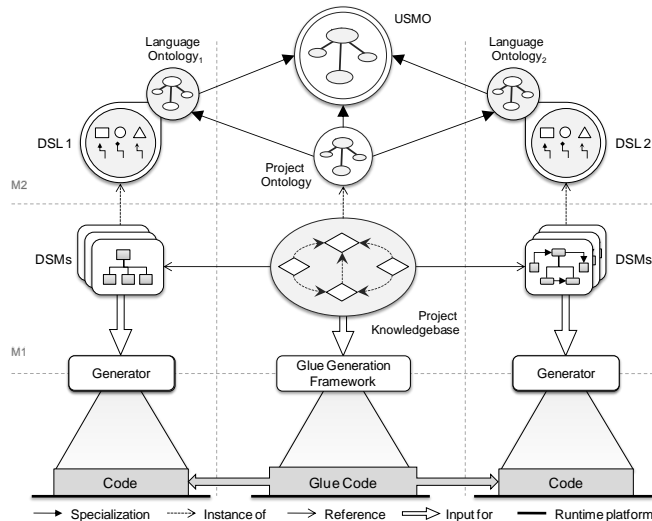


Fig. 1. HybridMDS Architecture Overview

## References

1. M. Bräuer and H. Lochmann. Towards Semantic Integration of Multiple Domain-Specific Languages Using Ontological Foundations. In *4th Int. Workshop on (Software) Language Engineering (ATEM'07)*, Nashville, TN, USA, Oct. 2007.
2. M. Bräuer and H. Lochmann. An Ontology for Software Models and its Practical Implications for Semantic Web Reasoning. In *Proceedings of 5th European Semantic Web Conference*, 2008.