Five Minutes Overview

Table of contents

1. ModelCVS at a glance

Seamless exchange of models among different modeling tools increasingly becomes a crucial prerequisite for effective software development processes. Due to lack of interoperability, however, it is often difficult to use tools in combination, thus the potential of model-driven software development cannot be fully utilized. To tackle this problem, we propose ModelCVS, a system aiming at model-based tool integration. ModelCVS enables transparent transformation of models between different tools' languages and exchange formats, as well as versioning exploiting the rich syntax and semantics of models, thus going beyond existing low-level model transformation approaches. For this, ModelCVS utilizes semantic technologies in terms of ontologies and supports different integration patterns at the metamodel level. To foster reuse, a knowledge base captures essential information relevant for tool integration.

ModelCVS at a glance

2. Research Goals

ModelCVS at a glance

2.1. New Language for Scalable Model-Based Tool Integration

Metamodel bridging. Model-based tool integration comprises creating so-called metamodel bridges between the different tool metamodels (i.e., the metamodels of the modeling languages supported by the tools). These metamodel bridges define the model transformations facilitating transparent model translation. The main problems in creating such bridges arise due to metamodel heterogeneity in various aspects and due to the fact that existing implementation technologies are not exactly appropriate for the metamodel bridging task. While we do not attempt to fully solve metamodel heterogeneity in any case – for certain reasons heterogeneity is actually considered a necessity – we aim at providing improved technologies for dealing with metamodel heterogeneity in more efficient and evolvable ways.

Integration patterns and bridging operators. For these reasons, we aim at defining a language specifically tailored to metamodel bridging. We will identify architectural model integration patterns (integration patterns for short) that ensure openness, scalability, and evolvability of a tool integration solution. These will serve as basis to define specific bridging tasks and to develop appropriate bridging operators forming a metamodel bridging language that supports the identified integration patterns. An initial set of integration patterns is proposed, namely translation (i.e., bridging syntactic and semantic heterogeneity between largely overlapping tool metamodels), alignment (i.e., bridging cross-cutting concerns of partly overlapping tool metamodels), modularization (i.e., decomposing monolithic tool metamodels as a prerequisite for scalable bridging), and versioning (i.e., semantic-based migration of different versions of tool metamodels).

2.2. Innovative Technologies for Ontology-Based Metamodel Integration

Ontologies for metamodel integration. The proposed project makes extensive use of semantic technologies for the integration of tool metamodels as well as for the realization of semantically aware model versioning mechanisms. We assume that addressing the integration problem at the semantic level using ontologies improves the quality of automation support that can be achieved. Given the fact that a huge amount of work already exists in the area of ontology integration, the question arises as how these research results can be employed for ontology-based metamodel integration. The essential difference between metamodels and ontologies is that metamodels define the concepts of a modeling language in terms of their syntax, whereas ontologies focus on the semantics of concepts, disregarding syntactical concerns. Therefore, in order to harness the potential of ontologies for metamodel integration and semantic versioning, the difference in abstraction level and semantic expressiveness between metamodels and ontologies needs to be dealt with.

Metamodel lifting. In this respect, we aim to enable transitioning from the mostly syntactic metamodel level to the semantic ontology level in terms of a translation and subsequent syntax abstraction and semantic enrichment, furtheron called metamodel lifting. The lifting process should result in a mapping between metamodel level and ontology level such that both levels can be used synergically. Regarding utilization of the expressiveness and reasoning capabilities of the semantic level, we aim in particular at supporting the various integration tasks as outlined above.

2.3. Open Knowledge Base for Tool Integration

Reuse capabilities. The basic idea behind the semantic infrastructure in ModelCVS is to enrich metamodels with specific semantics. As suggested above, this can be achieved by deriving tool ontologies from tool metamodels, which provide proper semantics for modeling languages. The entailment of specific semantics through an enrichment of tool ontologies, however, shall be possible with reasonable effort. Therefore, a key requirement is to provide reuse capabilities for the process of defining specific semantics for a tool ontology.

Tool integration knowledge base. Hence, our research aims at constructing a tool integration knowledge base that, similar to a library, provides reusable concepts for the enrichment of individual tool ontologies. The knowledge base should enable semantic support for ontology-based metamodel bridging, as well as improved detection of versioning conflicts as motivated by the introductory example. The knowledge base should furthermore be open for usage outside the scope of ModelCVS.

Content of the knowledge base. Specific research tasks comprise, first, identification of generic, reusable concepts and development of a structure to organize the contents of the knowledge base and to enable efficient reuse. Second, devising a set of reference examples, which will be the result of our case study, to populate the tool integration knowledge base with, to be used to enhance ModelCVS' matching and reuse capabilities. Third, defining knowledge about semantic merging conflicts as required for enhanced model versioning capabilities, i.e., automated identification and subsequent resolution of such conflicts. Fourth, establishment of a public platform enabling Internet-wide access and contributions to the knowledge base as to maximize reuse effects.