Semantic Conflict Detection in Software Models

Author: Martin Olejár | Supervisor: Dr. Karol Rástočný | Slovak University of Technology in Bratislava

Scenario: UML class models

Base version

Derived version no. 1

Derived version no. 2

Transformation of model versions into ontology

Ontology mapping algorithm

- mapping of ontology representing base version with ontologies representing derived versions
- mapping of ontologies representing derived versions

Identification of same_as relationships
- semantically most similar elements
- comparison of all element properties
- defined weights for each property
- similarity value compared with similarity threshold

Identification of is_a relationships
- classes in hyperonymy relation

Identification of changes in model versions

Our goal and motivation

- Our goal is to detect static semantic and ontological conflicts in UML class models that can occur during parallel development.
- Our motivation is to prevent occurrence of defects in models and invalid merged model version which can directly influence final quality of models and source code written based on models.
- Correct detection of conflicts with their visualization is a good prerequisite for their resolution and successful synchronization of model versions.

Conflict detection based on defined conditions

Semantically similar elements
- added elements into versions
- elements must be same_as

Classes in hyperonymy relation
- added classes into versions
- classes must be is_a
- classes inherit from same_as classes

Inheritance hierarchy cycle
- creation of merged version
- recursion starting from each class

Evaluation with manually created dataset