Models in Conflict – Towards a Semantically Enhanced Version Control System for Models*

Kerstin Altmanninger

Department of Telecooperation, Johannes Kepler University Linz, Austria
kerstin.altmanninger@jku.at

Abstract. For a widespread success of the model-driven paradigm, appropriate tools such as "Version Control Systems" (VCS) are required to adequately support a model-based development process. First attempts to model-based versioning, however, perform conflict detection mainly on basis of a syntactic representation of models without exploiting their semantics. Consequently, in this paper the approach towards a semantically enhanced VCS is presented which enables for semantic conflict detection allowing not only a more precise conflict detection but also the determination of a conflict's reason, which can simplify the merge process. This is achieved by introducing the concept of semantic views which explicate a certain aspect of a modeling language's semantics relevant for conflict detection.

1 Motivation

The shift from code-centric to model-centric software development places models as first class entities in "Model-driven Development" (MDD) processes. "Version Control Systems" (VCS) are essential when the development process proceeds in parallel. In case the employed modeling tools are not tightly coupled to the VCS, certain approaches that rely on pessimistic methods (e.g., locking) or tracking model modifications (e.g., operation-based mechanisms) are not applicable. Instead a loosely-coupled and optimistic VCS has to be provided which operates in a state-based manner [1].

2 Problem

The challenges emerging when realizing a loosely coupled, optimistic and state-based VCS span from *model comparison* over *conflict detection* and *conflict resolution* to model *merging* [2, 3]. First, *model comparison* should not rely on text- or tree based VCS [4–6] since they do not take the logical structure of models into account which is required for effective model comparison. Hence existing graph-based approaches have to be employed. Second, *conflict detection*

* This work has been partly funded by the Austrian Federal Ministry of Transport, Innovation and Technology (BMVIT) and FFG under grant FIT-IT-810806.

does not only need to consider the syntactical structure of models but should also "understand" the models semantics to be able to properly identify conflicts. But, full formal specifications of the semantics [7] underlying a model are not feasible since only certain aspects are relevant to support the process of conflict detection. Third, *conflict resolution* requires appropriate identification of the reasons of conflicts especially when going beyond just supporting syntactical conflict detection. Model *merging*, finally, must produce one consistent new model which can be facilitated by model transformations.

3 Approach

In the light of the aforementioned challenges, a semantically enhanced, graph-based VCS for models is proposed. As the core mechanism for representing certain aspects of the model's semantic so-called "views of interest" are introduced with which syntactic sugar can be eliminated and hidden concepts can be made explicit.

The basis of the approach is the metamodel which describes the syntax of the models which have to be versioned. Additionally, to be able to provide semantic conflict detection a metamodel representing a certain view of interest has to be defined. On basis of those metamodels a transformation can be specified such that the rules of a model transformation relate the elements of the metamodel (abstract syntax) to which the original model conforms to and the elements of the metamodel representing the definition of the view of interest. As a consequence of the transformation realizing a *semantic mapping*, conflict detection can be carried out on both, model and semantic view. Conflicts that are determined purely upon the comparison of two versions of a model are *syntactical conflicts* whereas a *semantic conflict* is detected between the representations of such a model's versions in a semantic view. The actual finding of conflicts in both the original model and the view works analogous to the graph-based detection of structural conflicts in existing VCS like [8–11]. Accordingly, four possible combinations of scenarios that can occur when a model's semantic views are incorporated into conflict detection can be identified.

Fig. 1 depicts a simplified "Business Process Execution Language" (BPEL) example. The metamodel contains *Activities* which are either contained ordered in a *Sequence* or are explicitly connected to another preceding *Activity* by a *Link*. The metamodel of a concrete semantic view defined upon BPEL eliminates the "syntactic sugar" of *Sequence* expressing everything in terms of *Activity* and *Link*. The model transformation between those metamodels can then be executed on concrete models to create the necessary semantic view on the working copies during the 3-way-conflict detection process. Fig. 1 also shows examples for the four possible combinations of scenarios ($A-D$) during conflict detection between three model versions (the last revision in the repository V and the two model versions V' and V'' edited by model developers).

Preceding work in this area [1] concentrates on different *conflict detection strategies* and states *OCL expressions* for the *identification of conflict sets*.

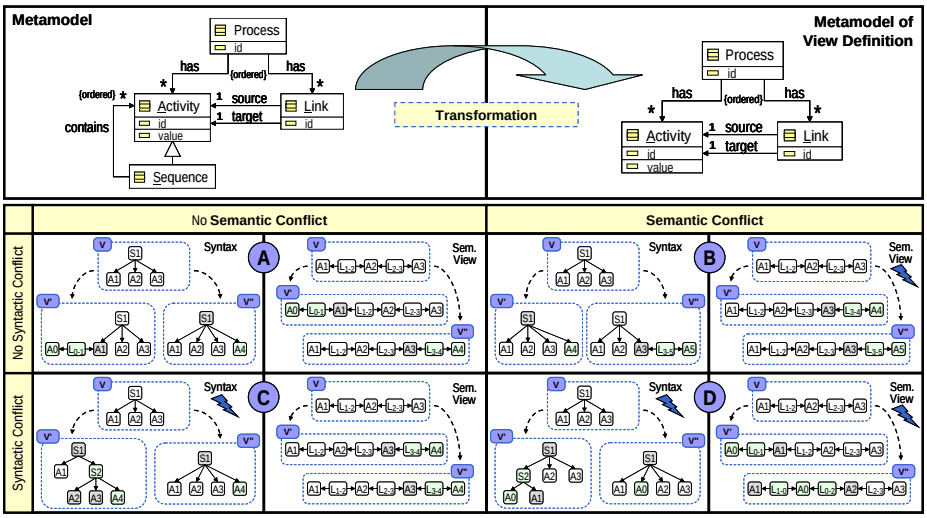

Fig. 1. Overview on the conflict detection process.

Through the definition of views of interest the proposed system can find conflicts more precisely during the conflict detection process since it is now possible to identify whether a conflict occurs due to syntactic differences or differences in the respective view. Consequently, previously falsely indicated syntactic conflicts can be avoided and previously undiscovered semantic conflicts can be found.

4 Prototypical Implementation

A first prototype realizing the approach presented has already been implemented. The prototype is open with respect to the usage of different modeling environments, using XMI as an exchange format. Furthermore, the proposed VCS relies on metamodeling techniques and MDD standards like the "Eclipse Modeling Framework" (EMF) [12], the "Atlas Transformation Language" (ATL) [13] and the EMF reference implementation of "Service Data Objects" (SDO) [14] for computing change summaries between two models. These standards are providing flexibility of the approach to operate on virtually any modeling language. The realized prototype is extensible to include the semantics definitions needed for advanced conflict detection [1] thus realizing a light-weight semantic VCS.

5 Related Work

The closest approach to the work presented in this paper is laid out by *SemVersion* [15], which is itself based on RDF, proposing the separation of language specific features (e.g., semantic difference) from general features (e.g., structural

difference or branch and merge). To perform the semantic difference the semantics of the used ontology language (e.g., RDF Schema) are taken into account in form of calculating a structural difference of two versions of an RDFS ontology. *SemVersion*, however, is not flexible to operate on any modeling language and furthermore does not provide version control functionalities. In terms of optimistic state-based VCSs, *Odyssey-CVS* [9] presents a graph-based system for versioning UML elements, aiming to support UML-based CASE tools in evolving their artifacts. *Odyssey-CVS*, therefore, is not flexible in the used modeling language but open to incorporate with any modeling environment. Conflicts found are identified purely on basis of a structural comparison of two versions of a model without incorporating semantics. *Ohst et al.* [16] addresses the problem of how to detect and visualize differences between versions of UML models such as class or object diagrams. Their approach is loosely coupled to the modeling environment and provides difference computation for UML models, only. The proposed difference computation algorithm detects only structural differences visualized to the developer and is not extensible to "understand" the semantics of a model. VCSs which focus on models but are tightly coupled to the modeling environment like e.g. [10, 11] are only remotely related.

Summarizing, current research areas in the field of VCSs for models are often limited to specific modeling languages, tightly coupled to the modeling environment and do not incorporate modeling language's semantics in contrast of the approach proposed.

6 Conclusion

In this paper a light-weight approach for incorporating semantics into VCSs for models is proposed. By means of transforming a model into a semantic view, syntactic sugar can be eliminated and hidden concepts can be explicated. Hence, the joint use of model transformations expressing certain semantic aspects of a modeling language, and the employment of graph-based comparison techniques on models and views, allows for a more precise conflict detection (laid out in [1]) between versions of models. Thus, with a relatively small amount of effort for establish the necessary transformations benefits in the conflict detection can be gained.

6.1 Future Work

Future research, in the short distant prospect, will focus on the conflict detection strategies (cf. [1]) and how to extend the conflict detection approach to also operate on multiple views of interest. To be able to define such views future researches, however, will deal with semantic view definition challenges. Furthermore, it is necessary to investigate into conflict resolution concepts and techniques and how to support the user in a loosely coupled scenario. During evaluation it also needs investigation in the development of techniques how the approach can be specifically fine-tuned towards different modeling languages.

6.2 Evaluation

For evaluating the feasibility of the approach it is planned to apply the approach to a series of large-scale complex examples for exploring the approach's limitations. For this, appropriate examples stemming from real-world scenarios will be identified. The overall evaluation will be conducted on basis of a comparative analysis comparing the proposed prototype with existing graph-based VCS for models like [8–11] applying quantitative measures.

References

1. Altmanninger, K., Bergmayr, A., Kotsis, G., Reiter, T., Schwinger, W.: Models in Conflict – Detection of Semantic Conflicts in Model-based Development. In: Proc. of the 3rd Int. Workshop on Model-Driven Enterprise Information Systems (MDEIS). (2007) 29–40
2. Westfechtel, B.: Structure-Oriented Merging of Revisions of Software. In: SCM. (1991) 68–79
3. Mens, T.: A State-of-the-Art Survey on Software Merging. IEEE Trans. Software Eng. **28**(5) (2002) 449–462
4. Subversion. http://subversion.tigris.org/
5. Concurrent Versions System. http://www.nongnu.org/cvs/
6. Bendix, L., Larsen, P.N., Nielsen, A.I., Petersen, J.L.S.: CoEd – A Tool for Versioning of Hierarchical Documents. In: ECOOP '98: Proc. of the SCM-8 Symposium on System Configuration Management. Volume 1439 of LNCS. (1998)
7. Harel, D., Rumpe, B.: Meaningful Modeling: What's the Semantics of "Semantics"? Computer **37**(10) (2004) 64–72
8. Rational Software Architect. IBM Homepage (2007)
9. Oliveira, H., Murta, L., Werner, C.: Odyssey-VCS: a Flexible Version Control System for UML Model Elements. In: Proc. of the 12th Int. Workshop on Software Configuration Management (SCM), ACM Press (2005)
10. Nguyen, T.N.: A Novel Structure-Oriented Difference Approach for Software Artifacts. In: Proc. of the 30th Int. Computer Software and Applications Conference. (2006) ISBN: 0-7695-2655-1.
11. Oda, T., Saeki, M.: Generative Technique of Version Control Systems for Software Diagrams. In: Proc. of the 21st IEEE Int. Conf. on Software Maintenance. (2005)
12. EMF Homepage. http://www.eclipse.org/modeling/emf/ (2007)
13. Allilaire, F., Bézivin, J., Jouault, F., Kurtev, I.: ATL – Eclipse Support for Model Transformation. In: Proc. of the Eclipse Technology eXchange Workshop (eTX) at ECOOP. (2006)
14. SDO Homepage. http://www.eclipse.org/modeling/emf/?project=sdo (2007)
15. Völkel, M.: D2.3.3.v2 SemVersion – Versioning RDF and Ontologies. $http://www.aifb.uni-karlsruhe.de/Publikationen/showPublikation?publ_id = 1163$ (January 2006)
16. Ohst, D., Welle, M., Kelter, U.: Differences between versions of UML diagrams. In: Proc. of the 9th European Software Engineering Conference (ESEC). Number ISBN: 1-58113-743-5, ACM Press (2003) 227–236