A Generic Framework for Realizing Semantic Model Differencing Operators[⋆]

Philip Langer, Tanja Mayerhofer, and Gerti Kappel

Business Informatics Group, Vienna University of Technology, Vienna, Austria
{langer,mayerhofer,gerti}@big.tuwien.ac.at

Abstract. As models constitute the main software artifacts in model-driven engineering, managing their evolution has attracted much research. One important technique in this realm is model differencing concerned with identifying differences among models. The majority of existing model differencing approaches identify differences by applying a fine-grained analysis of models based on their abstract syntax representation. Thereby, syntactic differences among models can be revealed. However, syntactic differences can only approximate semantic differences among models. We propose a generic framework for realizing semantic model differencing operators revealing semantic differences among models [7]. Therefore, we utilize the behavioral semantics specification of the considered modeling language to execute the models to be compared and capture execution traces providing semantic interpretations over the models. By comparing these semantic interpretations, semantic differences among models can be identified.

1 Introduction

Managing the evolution of models requires techniques for identifying differences among independently developed or consecutive versions of models. The majority of existing model differencing approaches compare models based on their abstract syntax representation. This can be done in a generic manner by incorporating the metamodel to which the compared models comply into the differencing algorithm [1]. Thereby, the differencing algorithm first identifies corresponding model elements among the models to be compared and then performs a fine-grained comparison of all corresponding model elements. This results in a set of *syntactic differences* among the models, which are usually represented in terms of edit operations, such as add, delete, and update.

Syntactic differences among models constitute valuable and efficiently processable information sufficient for several application domains. However, they can only approximate *semantic differences* among models. As pointed out by Maoz *et al.* [8], few syntactic differences among models may induce very different semantics and syntactically different models may still induce the same semantics. Semantic model differencing enables several additional analyses compared to syntactic differencing, such as the verification of semantic preservation of changes like refactorings and the identification of

[⋆] This work is partly funded by the European Commission under the ICT Policy Support Programme grant no. 317859 and by the Austrian Federal Ministry of Transport, Innovation and Technology (BMVIT) under the FFG BRIDGE program grant no. 832160.

semantic conflicts among concurrent changes. Moreover, the identification of semantic differences among models provides the basis for comprehending the evolution of a model, as it enables to reason about the meaning of a change, that is the impact a syntactic change has on a model's semantics.

We propose a *generic* framework that enables to realize *semantic model differencing operators* for specific modeling languages [7]. According to the idea of generic syntactic differencing [1], we propose to utilize the behavioral semantics specification of a modeling language for supporting semantic model differencing. Thereby, we exploit the executability provided by the behavioral semantics specification to execute the models to be compared and to obtain execution traces. As these execution traces constitute semantic interpretations over the compared models, they can be used to identify semantic differences among the models. Therefore, the execution traces are syntactically compared by applying dedicated match rules defining which syntactic differences among them constitute semantic differences among the compared models. Execution traces, which lead to the identification of semantic differences constitute *diff witnesses*, that are manifestations of the semantic differences. They enable modelers to reason about a model's evolution and can be further processed for carrying out model management activities, such as model versioning.

In Section 2, we discuss existing work in semantic model differencing, before we introduce our approach in Section 3. Subsequently, we discuss in Section 4 an implementation of our approach for an existing semantics specification language and present evaluation results for this implementation. Finally, we conclude the paper in Section 5.

2 Related Work

Significant advances in semantic model differencing have been recently achieved by Maoz *et al.* [8], who proposed an approach for defining enumerative semantic differencing operators. In this approach, two models to be compared are translated into an adequate semantic domain whereupon dedicated algorithms are used to calculate semantic differences in terms of diff witnesses. By applying this approach, they defined differencing operators for UML class diagrams and UML activity diagrams called CDDiff [10] and ADDiff [9]. CDDiff computes object diagrams constituting valid instances of only one of two compared class diagrams. ADDiff calculates execution traces possible in only one of two compared activity diagrams. Gerth *et al.* [5] developed an approach for detecting semantically equivalent fragments of business process models. Therefore, business process models are translated into normalized process model terms, which are subsequently compared by syntactic differencing techniques. Reiter *et al.* [16] presented an approach for detecting semantic conflicts among change operations. In their approach, models are translated into so-called *semantic views*, which are then compared by syntactic differencing techniques. Fahrenberg *et al.* [4] propose an approach for defining non-enumerative semantic differencing operators. In their approach, the models to be compared are translated into a semantic domain having an algebraic structure that enables to define the difference among two models in terms of an operator on the semantic domain. They applied this approach to define semantic differencing operators for feature models and automata specifications [4], as well as UML class diagrams [3].

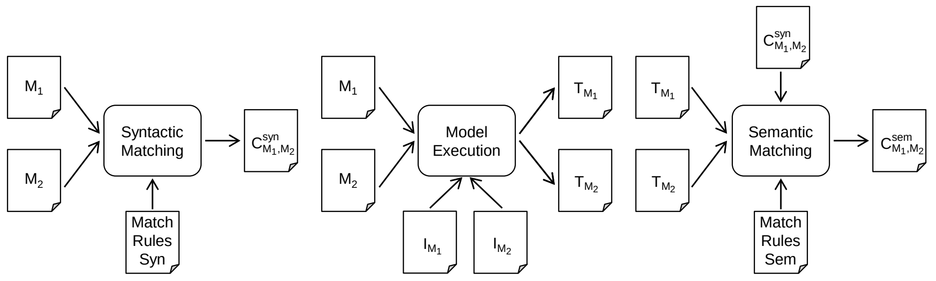

Fig. 1. Overview of semantic model differencing framework

3 Semantic Model Differencing

While the approaches presented by Maoz *et al.* [8] and Fahrenberg *et al.* [4] are generally applicable for developing semantic differencing operators, doing so poses the following major challenge. For developing a semantic differencing operator for a specific modeling language, one has to develop an often non-trivial transformation into a semantic domain encoding the semantics of the considered modeling language, implement an analysis algorithm dedicated to semantic differencing in this semantic domain, and translate the results back to the modeling language.

To mitigate this challenge, we propose a *generic* framework for realizing semantic model differencing operators for specific modeling languages [7]. This framework utilizes the behavioral semantics specification of a modeling language, which can be defined using existing semantics specification languages, such as xMOF [12], Kermeta [14], or DMM [2], for reasoning about semantic differences among models. Such semantics specifications can be used for various application domains, such as model simulation, verification, and validation. We aim at utilizing such semantics specifications also for semantic model differencing. Therefore, we exploit the executability provided by behavioral semantics specifications enabling to execute the models to be compared and obtain execution traces constituting semantic interpretations over the models. These semantic interpretations act as basis for identifying semantic differences.

Figure 1 depicts an overview of our semantic model differencing framework. In the *syntactic matching* step, syntactically corresponding elements of the two compared models M_1 and M_2 are identified based on syntactic match rules $MatchRulesSyn$. Thereby, syntactic correspondences $C^{syn}_{M_1,M_2}$ between the models are established. In the *model execution* step, the models M_1 and M_2 are executed for relevant inputs I_{M_1} and I_{M_2} based on the behavioral semantics specification of the modeling language. During the model execution, the traces T_{M_1} and T_{M_2} are captured, which constitute semantic interpretations over the models. In the *semantic matching* step, the captured traces T_{M_1} and T_{M_2} are compared based on semantic match rules $MatchRulesSem$ establishing semantic correspondences $C^{sem}_{M_1,M_2}$. Thereby, two models M_1 and M_2 are semantically equivalent, if the traces captured during their execution T_{M_1} and T_{M_2} match according to the semantic match rules.

Our semantic model differencing approach is *generic*, as it enables to implement semantic differencing operators for any modeling languages whose behavioral semantics is defined such that conforming models can be executed and execution traces can be obtained. Only the semantic match rules are specific to the realization of a semantic differencing operator. This is an important differentiator of our approach compared to currently existing semantic model differencing approaches.

4 Implementation

We implemented the proposed semantic model differencing framework for the operational semantics specification language xMOF [12]. xMOF integrates Ecore with UML's action language enabling the definition of a modeling language's behavioral semantics in terms of UML activities. Thereby, models can be executed by executing the activities defined in the semantics specification using the fUML virtual machine [15].

In our approach, execution traces constitute the basis for reasoning about semantic differences among models. We defined a generic trace format that serves as interface of our semantic differencing framework. Hence, our framework does not directly depend on a specific semantics specification language or virtual machine, but only operates on traces conforming to this trace format. For fUML, a dedicated trace model exists [11], which enables to create a trace conform to the defined format. The execution traces are compared according to semantic match rules, which define based on the runtime information captured in the traces, which model elements semantically correspond to each other and whether two models are semantically equivalent. For defining semantic match rules, our implementation integrates the model comparison language ECL [6].

We evaluated the expressive power of our generic model differencing framework by carrying out two case studies based on our implementation. In these case studies, we realized semantic differencing operators for UML activity diagrams and UML class diagrams according to ADDiff [9] and CDDiff [10]. This enabled us to assess whether our generic framework provides sufficient expressive power to define non-trivial semantic differencing operators. Therefore, we implemented the semantics of UML activity diagrams and UML class diagrams using xMOF as well as semantic match rules using ECL, both according to the definitions provided by Maoz *et al.* The semantic differencing operators defined with our framework enabled us to detect the same diff witnesses as Maoz *et al.* among their case study example models. Thus, we conclude from the case studies, that the expressive power of our generic semantic differencing framework is sufficient for defining non-trivial semantic differencing operators. More details about the case studies including performance measurements may be found in [7,13].

5 Conclusion

In this paper, we presented a generic framework for realizing semantic model differencing operators. In contrast to existing approaches, our approach follows the spirit of generic syntactic model differencing by employing the behavioral semantics specification of a modeling language to reason about semantic differences among models. Thus, non-trivial transformations into a semantic domain specifically required for enabling semantic differencing can be avoided. We discussed how our framework can be realized for the operational semantics specification language xMOF. The evaluation of our approach based on two case studies revealed that our approach is expressive enough to define semantic differencing operators for specific modeling languages.

References

1. M. Alanen and I. Porres. Difference and Union of Models. In *Proc. of 6th Int. Conf. on the Unified Modeling Language (UML'03)*, volume 2863 of *LNCS*, pages 2–17. Springer, 2003.
2. G. Engels, J. H. Hausmann, R. Heckel, and S. Sauer. Dynamic Meta Modeling: A Graphical Approach to the Operational Semantics of Behavioral Diagrams in UML. In *Proc. of 3rd Int. Conf. on the Unified Modeling Language (UML'00)*, volume 1939 of *LNCS*, pages 323–337. Springer, 2000.
3. U. Fahrenberg, M. Acher, A. Legay, and A. Wasowski. Sound Merging and Differencing for Class Diagrams. In *Proc. of 17th Int. Conf. on Fundamental Approaches to Software Engineering (FASE'14)*, volume 8411 of *LNCS*, pages 63–78. Springer, 2014.
4. U. Fahrenberg, A. Legay, and A. Wasowski. Vision Paper: Make a Difference! (Semantically). In *Proc. of 14th Int. Conf. on Model Driven Engineering Languages and Systems (MODELS'11)*, volume 6981 of *LNCS*, pages 490–500. Springer, 2011.
5. C. Gerth, J. M. Küster, M. Luckey, and G. Engels. Precise Detection of Conflicting Change Operations Using Process Model Terms. In *Proc. of 13th Int. Conf. on Model Driven Engineering Languages and Systems (MODELS'10)*, volume 6395 of *LNCS*, pages 93–107. Springer, 2010.
6. D. Kolovos, L. Rose, A. García-Domínguez, and R. Paige. *The Epsilon Book*. Online available at: `http://www.eclipse.org/epsilon/doc/book`, March 2014.
7. P. Langer, T. Mayerhofer, and G. Kappel. Semantic Model Differencing Utilizing Behavioral Semantics Specifications. In *Proc. of 17th Int. Conf. on Model Driven Engineering Languages and Systems (MODELS'14)*, volume 8767 of *LNCS*, pages 116–132. Springer, 2014. Accepted for publication.
8. S. Maoz, J. O. Ringert, and B. Rumpe. A Manifesto for Semantic Model Differencing. In *Models in Software Engineering: Workshops and Symposia at MODELS 2010, Reports and Revised Selected Papers*, volume 6627 of *LNCS*, pages 194–203. Springer, 2011.
9. S. Maoz, J. O. Ringert, and B. Rumpe. ADDiff: Semantic Differencing for Activity Diagrams. In *Proc. of 19th ACM SIGSOFT Symposium and 13th European Conf. on Foundations of Software Engineering (ESEC/FSE'11)*, pages 179–189. ACM, 2011.
10. S. Maoz, J. O. Ringert, and B. Rumpe. CDDiff: Semantic Differencing for Class Diagrams. In *Proc. of 25th European Conf. on Object-Oriented Programming (ECOOP'11)*, volume 6813 of *LNCS*, pages 230–254. Springer, 2011.
11. T. Mayerhofer, P. Langer, and G. Kappel. A Runtime Model for fUML. In *Proc. of 7th Workshop on Models@run.time (MRT'12)*, pages 53–58. ACM, 2012.
12. T. Mayerhofer, P. Langer, M. Wimmer, and G. Kappel. xMOF: Executable DSMLs Based on fUML. In *Proc. of 6th Int. Conf. on Software Language Engineering (SLE'13)*, volume 8225 of *LNCS*, pages 56–75. Springer, 2013.
13. Moliz project. `http://www.modelexecution.org`.
14. P.-A. Muller, F. Fleurey, and J.-M. Jézéquel. Weaving Executability into Object-Oriented Meta-Languages. In *Proc. of 8th Int. C. on Model Driven Engineering Languages and Systems (MODELS'05)*, volume 3713 of *LNCS*, pages 264–278. Springer, 2005.
15. Object Management Group. Semantics of a Foundational Subset for Executable UML Models (fUML), Version 1.0, February 2011. `http://www.omg.org/spec/FUML/1.0`.
16. T. Reiter, K. Altmanninger, A. Bergmayr, W. Schwinger, and G. Kotsis. Models in Conflict - Detection of Semantic Conflicts in Model-based Development. In *Proc. of 3rd Int. Workshop on Model-Driven Enterprise Information Systems (MDEIS'07)*, pages 29–40, 2007.