A Bidirectional Transformation Approach towards Automatic Model Synchronization*

Yingfei Xiong, Dongxi Liu, Zhenjiang Hu, Masato Takeichi

Department of Mathematical Informatics
Graduate School of Information Science and Technology
University of Tokyo, Hongo 7-3-1, Bunkyo-ku, Tokyo 113-8656, Japan
{Yingfei_Xiong, liu, hu, takeichi}@mist.i.u-tokyo.ac.jp

Model-driven architecture (MDA) [1] is a discipline in software engineering that relies on models as first class entities and that aims to develop, maintain and evolve software by performing model transformations. Refinement, abstraction, refactoring, and integration of models are special cases of model transformations that can be found in many areas of software engineering [2].

However, there are situations where only a transformation from one model to another is not enough. Users may change models after the transformation in both the source side and the target side, causing the maintenance problem. Consider the following scenario: first designers create a UML model to describe the system design, and then perform a model transformation on the UML model to produce a Java source model for programmers to implement the system. Programmers add code and methods to the Java source model, while at the same time, designers change the name of a class on the UML model. Now the UML model and the Java source model become inconsistent and need to be synchronized. Simply performing the transformation from UML model to Java source model again will not synchronize the models, because the modifications on the Java source model will be lost. To synchronize the models, we need to propagate the changes both from the UML model to the Java source model and from the Java source model to the UML model. We call the process of propagating changes across heterogeneous models (models of different formats) as *model synchronization*.

Existing model synchronization systems and model synchronization frameworks [3] require users to manually write code to deal with each type of modification on each type of model. However, this is often a difficult task and it is also difficult to know whether manually-written code is correct or not.

In this paper we propose a more automated approach: to extract necessary information from the existing transformations, and to automatically synchronize models without requiring users to write extra code.

Our approach is based on bidirectional transformation. Bidirectional transformation is originated from the view-updating techniques [4] and now intensively studied by researchers on XML transformation [5][6]. In bidirectional transformation, researchers give semantics to uni-directional transformations so

* This research was supported in part by a grant from Japan Society for the Promotion of Science (JSPS) Grant-in-aid for Scientific Research (A) 19200002, and by the National Natural Science Foundation of China under Grant No. 60528006.

that the transformation system can reflect modifications on the target model back into the source. In other words, transformations can be *bidirectionally* executed. In our approach, we apply the idea of bidirectional transformation to model transformation, and use bidirectional model transformations to support model synchronization.

Our system is built upon ATL Virtual Machine on which ATL bytecode is executed. ATL byte code is part of a widely-used model transformation language called Atlas Transformation Language (ATL) [7]. ATL consists of a high-level declarative language and the low-level byte-code language. If other model transformation languages can be translated into ATL byte-code, they can also be executed by the ATL engine.

The main contributions of our work can be summarized as the following:

- We propose an approach to automatic model synchronization by extracting necessary information from existing transformations.
- We propose a set of properties to ensure the correctness of a synchronization process. Also, we discuss how our approach is related to the properties.
- We have implemented our approach on ATL Virtual Machine and have tested our system on several ATL transformations. In addition, our system can support other transformation language if these languages can be translated to ATL byte-code.

One limitation of our current system is that we cannot deal well with insertions on the target model. We are solving this problem by introducing virtual holes to the source side. This is one of our future work.

References

1. Frankel, D.S.: Model Driven Architecture: Applying MDA to Enterprise Comput ing. John Wiley & Sons (2003)
2. Czarnecki, K., Helsen, S.: Classification of model transformation approaches. In: OOPSLA 03 Workshop on Generative Techniques in the Context of Model-Driven Architecture. (2003)
3. Ivkovic, I., Kontogiannis, K.: Tracing evolution changes of software artifacts through model synchronization. In: ICSM '04: Proceedings of the 20th IEEE International Conference on Software Maintenance, Washington, DC, USA, IEEE Computer Society (2004) 252–261
4. Bancilhon, F., Spyratos, N.: Update semantics of relational views. ACM Trans. Database Syst. **6**(4) (1981) 557–575
5. Liu, D., Hu, Z., Takeichi, M.: Bidirectional interpretation of XQuery. In: PEPM '07: Proceedings of the 2007 ACM SIGPLAN symposium on Partial evaluation and semantics-based program manipulation, New York, NY, USA, ACM Press (2007) 21–30
6. Foster, J.N., Greenwald, M.B., Moore, J.T., Pierce, B.C., Schmitt, A.: Combinators for bi-directional tree transformations: a linguistic approach to the view update problem. In: ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Long Beach, California. (2005) 233–246
7. Jouault, F., Kurtev, I.: Transforming models with ATL. In: Satellite Events at the MoDELS 2005 Conference, LNCS 3844, Springer (2006) 128–138