Triple Graph Grammars and OCL for Validating System Behavior *

Duc-Hanh Dang

University of Bremen, Computer Science Department, Database Systems Group
hanhdd@informatik.uni-bremen.de

Abstract. We propose an approach based on the integration of Triple Graph Grammars (TGGs) and the Object Constraint Language (OCL) for checking the conformance between use case and design models.

1 Introduction

Triple graph grammars (TGGs) have been first formally proposed in [1]. They seem to be a promising approach for explaining relationships between models within model-driven development. A need and a challenge within software engineering is how to explain the relationship between behavioral models, a use case model as a functional requirement model and a design model as a realization of use cases. The major difficulty in this relationship lies in the informality of use cases, and the loose relationship between the use case model and the design model. We propose a novel approach employing TGGs for checking the conformance between a use case model and a design model. Within our approach, an integration of TGGs and OCL is proposed and implemented in our tool UML-based Specification Environment (USE).

A use case is a textual description of the system's behavior as the system responds to a request from actors (user types). A use case model may be realized by a design model. The relationship between the use case model and the design model should be a behavioral refinement. Explaining the relationship as well as concepts of refinement is a topic of ongoing research activities.

In [2], we propose an approach for explaining this relationship. The heart of the approach is to check the synchronization between system scenarios at the use case level and the design level. A step in a system scenario at the use case level may define the next step in the corresponding scenario at the design level, and vice versa, a step at the design level may define the next step at the use case level. In order to realize the approach, we define metamodels of system models at the use case level and the design level. The metamodels are extended by graph transformation rules, which allow defining a dynamic model evolution as a simulation of system evolution. We employ TGGs in order to relate transformation rules of the graph transformation systems towards synchronization between system scenarios.

* I would like to thank my supervisor Martin Gogolla for contributions to this work.

Within our approach, a mechanism for checking the conformance between a use case model and a design model is established. Test cases, which are generated from the use case model, are employed for defining scenarios. The mechanism represents system states in an easy-to-understand and visual form. It is significant for the early design phase. Within our approach, we do not address structural refinement: use case and design models share a domain class diagram. However, our approach has the possibility to support the structural refinement.

Some technical challenges arise within our approach. Our approach requires the integration of TGGs and OCL. In the recent literature, a need for the integration has been expressed. TGGs and QVT (Queries, Views, Transformations) share many building blocks. But in contrast to TGGs, QVT includes the declarative language OCL, which allows to express properties and to navigate in complex models. Attaching OCL constraints to TGG rules seems not difficult, but a challenge arises when inducing operations from the integration. To the best of our knowledge, an approach for the integration has not been proposed. Another challenge within our approach is that use cases and a design model must be specified as graph transformation systems. Some papers have specified a design model, e.g., UML interaction diagrams, as a graph transformation system. However, a metamodel for representing scenarios has not been proposed.

The rest of this paper is structured as follows. Section 2 discusses our approach for the integration of TGGs and OCL. Section 3 describes how TGGs can be employed for synchronizing scenarios. The paper is closed with a summary of our expected results. Due to the paper format, full references are not included. The final work will include a detailed discussion of related work.

2 Integrating TGGs and OCL

This section discusses our approach for the integration of TGGs and OCL. First of all, we define requirements for the integration of TGGs and OCL as follows.

- Supporting OCL formulas for attribute expressions.
- Supporting OCL conditions as pre- and posconditions of TGG rules (corresponding to negative application conditions and the right-hand side of TGG rules respectively), which cannot be represented by graphs.
- Supporting constraints on the metamodel of the middle domain of TGG rules. (Supposing that all of the constraints must be taken from TGG rules)
- Supporting bi-directionally executable TGG rules in a synchronous way.

We propose the language USE4TGG for the integration of TGGs and OCL: One part of USE4TGG has a one-one mapping to TGGs and the remaining part covers OCL concepts. The approach is similar to our previous work [3, 4]. A TGG rule in USE4TGG is transformed into an operation, which is realized in USE by taking two views on it: OCL pre- and postconditions are employed as operation contracts, and command sequences are taken as an operation realization.

USE4TGG fulfills all above requirements except the final requirement. Our restriction is that OCL formulas must be presented in an assignment-like style if we want to obtain a corresponding operation realization. We plan to develop an algorithm for translating TGG rules into operations for synchronizing models.

3 TGGs and OCL for Synchronizing Scenarios

This section describes our approach employing TGGs and OCL for synchronizing scenarios of a use case model and a design model.

In order to define graph transformation systems controlling scenarios of a use case model and a design model, we have already developed the language UCL for specifying use cases and the language DML for specifying scenarios for executing design models. A set of rules for selecting scenarios of the use case model and the design model has been defined. Due to the paper format, we do not describe these languages in detail.

Effects of TGG rules on scenarios at the use case level include: (1) To assign input values to variables; (2) to select the next activity; (3) to select the next activity from effects of the design model. Effects of TGG rules on scenarios at the design level include: (1) To assign input values to variables and to select the next operation; (2) to select the next action of the current operation; (3) to select the next operation corresponding to the current activity of use cases.

Our next task is to define TGGs rules for relating the graph transformation systems, and to implement a TGG engine for realizing the synchronization.

4 Expected Contributions

The followings are expected results of our work.

- Discovering a new application scenario of TGGs for synchronizing graph transformation systems. The result is significant for software engineering.
- Proposing an approach for the integration of TGG and OCL and implementing a corresponding TGG engine in the USE tool.
- Proposing and realizing an approach based on TGGs for checking the conformance between a use case model and a design model.

References

1. Schürr, A.: Specification of Graph Translators with Triple Graph Grammars. In Mayr, S., ed.: WG, Springer-Verlag, LNCS 903 (1995) 151–163
2. Dang, D.H.: Validation of System Behavior Utilizing an Integrated Semantics of Use Case and Design Models. In Pons, C., ed.: Proc. Doctoral Symposium ACM/IEEE 10th Int. Conf. MoDELS 2007, CEUR Workshop Proceedings, Vol. 262 (2007)
3. Gogolla, M., Büttner, F., Dang, D.H.: From Graph Transformation to OCL using USE. In Schürr, A., Nagl, M., Zündorf, A., eds.: AGTIVE, Springer Berlin (2008)
4. Dang, D.H., Gogolla, M., Büttner, F.: From UML Activity Diagrams to CSP Processes: Realizing Graph Transformations in the UML and OCL Tool USE. In: AGTIVE 2007 Tool Contest, 3rd International Workshop and Symposium on Applications of Graph Transformation with Industrial Relevance, http://gtcases.cs.utwente.nl/wiki/UMLToCSP/USE (2007)