Pattern-Based Model Transformation

Sheena R. Judson
Computer Science Department
Louisiana State University
Baton Rouge, Louisiana USA
1-817-935-4518
judson@csc.lsu.edu

ABSTRACT

Model Driven Architecture (MDA), which supports the development of software-intensive systems through the transformation of models to executable components and applications, requires a standard way to express transformations. The approach developed by this research focuses on defining pattern-based transformation at the metamodel level. This research has two primary objectives. The first objective is to support systematic application of patterns. The use of patterns as model building blocks (through pattern-based transformations) helps raise the level of abstraction at which systems are developed. The second research objective is to support controlled model evolution by specifying pattern-based transformations at the metamodel level.

Categories and Subject Descriptors

D.2.2 [**Software Engineering**]: Design Tools and Techniques – *Object-oriented design methods*; D.2.13 [**Software Engineering**]: Reusable Software – *Reuse models, Domain engineering.*

General Terms

Design

Keywords

model transformation, model evolution, UML, design patterns, Query/Views/Transformations (QVT), Model Driven Architecture (MDA)

1. INTRODUCTION

Since the emergence of the Model Driven Architecture (MDA) initiative, numerous techniques have been proposed for the transformation of models (e.g., [1, 2]). MDA supports the development of software-intensive systems through the transformation of models to executable components and applications. The main motivation behind MDA is to transfer the focus of work from programming to solution modeling by treating models as the primary artifacts of development. As stated in [3], “MDA provides a set of guidelines for structuring specifications expressed as models and the mappings between those models”. The mappings transform the elements of a source model, which conforms to a particular metamodel, into elements of another model, the target model, which conforms to a metamodel [3].

In response to the need for a standard approach to define mapping functions that map between metamodels, the Object Management Group (OMG) issued the MOF 2.0 Query/View/Transformation (QVT) Request for Proposals (RFP) [3]. “The principle requirement of QVT is to provide a … standard means for expressing transformations [4]”. QVT requires that model transformations be defined precisely in terms of the relationship between a source MOF metamodel and a target MOF metamodel.

The transformation of, and mapping between, models are the key aspects of MDA [3]. In particular, well-defined transformations that support rigorous model evolution, refinement, and code generation are considered key elements of an MDA approach and the QVT. This research focuses on developing a technique that supports rigorous modeling of pattern-based model transformations. The modeled transformations can be used as the basis for developing tools that support rigorous and systematic application of reusable transformations.

2. GOALS

The research is aligned with the goals of both MDA and QVT and centers on developing a metamodeling approach for describing families of transformations. Specifically it is concerned with developing Unified Modeling Language (UML) [5] model transformations at the metamodel level. The focus is on a type of transformation referred to as *pattern-based model refactoring*. In pattern-based refactoring, a well-defined pattern is incorporated into a source design model. The result is a target model that contains an instantiation of the pattern. Controlled model refactoring can be accomplished by developing metamodels for the transformations. The metamodels can be used to constrain how the refactoring is carried out on the models and act as points against which the model-level transformations can be checked for conformance. The goals of this thesis are to:

1. Support the MDA goal of reducing software development time by raising the level of abstraction through the use of models and design patterns.
2. Capture transformations in a form that is reusable, which can lead to the development of tools that support controlled evolution of models (with respect to the reusable transformations) at the metamodel level.

3. METAMODELING APPROACH

Figure 1 provides a diagrammatical overview of the transformation approach that will be the base of this research. The M2’ level, an extension of the UML metamodel level (M2), supports metamodeling of transformations. The M1’ level, an extension of the UML model level (M1), supports representation of model transformations. A model transformation, T1, at the M1’ level takes a source UML model and transforms it to a target UML model. T1, which is denoted using a notation adopted from [6], is a member of the family of transformations characterized at the metamodel level by *Transformation Pattern*. A transformation pattern consists of characterizations of source and target models

OOPSLA’03, October 26–30, 2003, Anaheim, California, USA.
ACM 1-58113-751-6/03/0010.

(*Source Pattern* and *Target Pattern*, respectively) and constraints on relationships between source and target elements. The *Source Pattern* is a metamodel that characterizes source UML models and the *Target Pattern* is a metamodel that characterizes target models for the family of transformations characterized by *Transformation Pattern*.

A transformation (e.g., T1) *conforms* to a transformation pattern if: (1) the source model (*Source Model*) is an instantiation of the source pattern (*Source Pattern*), (2) the target model (*Target Model*) is an instantiation of the target pattern (*Target Pattern*), and (3) the relationship between elements of source and target models satisfy the constraints specified by the transformation pattern. A UML model that conforms to a source or target pattern metamodel is said to be an *instance* of the source or target pattern. Similarly, a model transformation that conforms to a transformation pattern is said to be an instance of the transformation pattern. The transformation pattern developed by this research consists of three parts: *Source Pattern*, *Transformation Schema*, and *Transformation Constraint.*

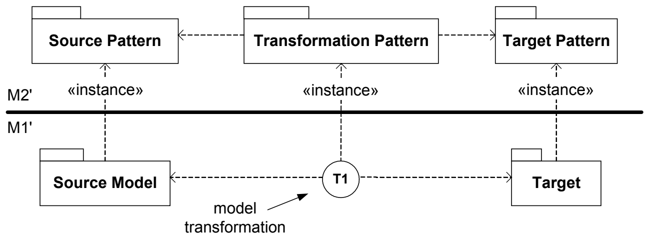

Figure 1. Transformation Overview.

The source pattern defines the set of source models to which transformations characterized by the transformation pattern can be applied. The pattern is expressed as a metamodel fragment that consists of classes characterizing model elements that are affected by the transformations. Each of the classes in the source pattern are specializations (subclasses) of classes in the UML metamodel. The source pattern thus determines a specialized UML metamodel for static structures.

The *transformation schema* shows the classes of model elements that are created by the transformations and the classes of source model elements that the transformations remove by conforming transformations. The classes in a transformation schema are specializations of UML metamodel classes. The schema is expressed as a metamodel fragment in which the classes of new model elements are enclosed in dashed boxes and classes of deleted source model elements are marked with an X.

The transformation schema determines the basic structure of the target model. The *transformation constraint* determines the relationships that must hold between target and source model elements. It is expressed as object structures, where the objects are prototypical instances of classes in the source pattern and the transformation schema (i.e., the objects are prototypical representations of UML model elements). The object structures describe transformation constraints in terms of relationships that must hold between elements of the source and target models.

4. STATUS AND FUTURE RESEARCH

This research presents an approach for model transformations that focuses on defining pattern-based transformations at the metamodel level. These pattern-based transformations are defined declaratively using the metamodel. We have applied the previously described approach to the Abstract Factory (AF) design pattern. As a result, AF transformation patterns have been developed for UML class and interaction design models. For both transformation patterns, the constraints are expressed both diagrammatically and through the use of the Object Constraint Language (OCL). We plan to apply our approach to other design patterns and UML models.

Future work includes the development of a technique for obtaining model level transformations from the metamodel characterizations. Currently, we are investigating whether the approach can be used to derive a SMW (Software Modeling Workbench) [7] transformation from a transformation pattern. Our goal is to show that the transformations conform to the UML metamodel through the use of the SMW transformation language.

In addition, we are developing a formal basis for model transformations. That is, we develop a formal basis for the metamodel descriptions and model level transformations so that one can establish that model level transformations conform to transformation specifications at the metal model level.

With the continued development of our model transformation approach, we hope to complete this work by May 2004.

5. REFERENCES

[1] Song, E., R. B. France, D. K. Kim, and S. Ghosh. Using Roles for Pattern-based Model Refactoring. in Proceedings of the Workshop on Critical Systems Development with UML (CSDUML'02). 2002.

[2] Akehurst, David and Stuart Kent. A Relational Approach to Defining Transformations in a Metamodel. in UML 2002 - The Unified Modeling Language: Model Engineering, Concepts, and Tools. Springer, October 2002.

[3] Object Management Group. Request for Proposal: MOF 2.0 Query / View / Transformations RFP. OMG 2002. http://www.omg.org/docs/ad/02-04-10.pdf.

[4] QVT Partners. QVT: The high level scope. QVT-Partners, 2003. http://qvtp.org/downloads/qvtscope.pdf.

[5] Object Management Group, UML 2.0 Superstructure Final Adopted Specification. OMG 2003. http://www.omg.org/cgi-bin/doc?ptc/2003-08-02.

[6] QVT Partners. Initial Submission for MOF 2.0 Query/View/Transformations RFP. QVT-Partners, 2003. http://qvtp.org/downloads/1.0/qvtpartners1.0.pdf.

[7] Porres, Ivan. A Toolkit for Manipulating UML Models, TUCS Technical Report No. 441. Turku Center for Computer Science, Åbo Akademi University, January 2002. http://www.tucs.fi/Research/Series/index.php.