Chapter XII

Quality–Driven Model Transformations: From Requirements to UML Class Diagrams

Silvia Abrahão
Valencia University of Technology, Spain

Marcela Genero
University of Castilla – La Mancha, Spain

Emilio Insfran
Valencia University of Technology, Spain

José Ángel Carsí
Valencia University of Technology, Spain

Isidro Ramos
Valencia University of Technology, Spain

Mario Piattini
University of Castilla – La Mancha, Spain

ABSTRACT

Model-Driven Architecture (MDA) is a software engineering approach that promotes the use of models and model transformations as primary development artifacts. Usually, there are several ways to transform a source model into a target model. Alternative target models may have the same functionality but may differ in their quality attributes (e.g., understandability, modifiability). This chapter presents an approach to deal with quality-driven model transformations. Specifically, it focuses on a specific set of transformations to obtain UML class diagrams from a Requirements Model. A set of alternative transformations are identified, and the selection of the best alternative is done through a controlled experiment.
The goal of the experiment is to empirically validate which alternative transformation produces the UML class diagram that is the easiest to understand. This evidence can be further used to define high-quality transformation processes, as it will be based on empirical knowledge rather than on common wisdom and the intuition of the researchers and developers.

**INTRODUCTION**

Nowadays, the software development community is moving towards model-driven development processes whose goal is the development of software at a higher level of abstraction based on models and model transformations. Within this context, the Model-Driven Architecture (MDA) initiative (OMG, 2003) has attracted interest from both the research community and software practitioners. This approach comprises the use of models in all the steps of a software development project, until the delivery of the software on a given platform.

A MDA development process basically transforms a platform-independent model (PIM) into one or more platform-specific models (PSM), which are transformed into code (code model – CM). The CM is just the actual code generated from PSMs through transformation. Here, the goal is to decouple the way, in which software systems are currently defined, which is dependant on the technology they use (OMG, 2003).

A model transformation is a process of converting one model to another model. A model may be transformed to several alternative models that may have the same functionality but different quality attributes. For example, one model may be more reusable while another model may be more comprehensive to its stakeholders. Therefore, it is necessary to identify those transformations that produce models with the desired quality attributes.

To cope with the problem of selecting alternative transformations, this chapter presents an approach for quality-driven model transformations. The mechanisms to choose the appropriate alternatives can greatly differ depending on the nature and the domain of the transformations as well as the quality perspective that is chosen. We focus on a set of transformations defined to obtain UML class diagrams from a Requirements Model (Insfran, 2003). Assuring quality in representing the system’s conceptual model from requirements is particularly important, as the traceability between these models is not properly dealt with. Moreover, a conceptual model of good quality can help to minimize communication problems and misunderstandings of requirements among the stakeholders.

The quality perspective that we are interested in is the pragmatic quality (Lindland, Sindre & Solvberg, 1994). This quality category addresses the comprehension aspect of the model from the stakeholders’ perspective. Pragmatic quality captures how the model has selected an alternative “from among the many ways to express a single meaning”, and it essentially deals with making the model easy to understand.

The comprehension goal specifies that all audience members (or interpreters) completely understand the statements in the model that are relevant to them. This is an import quality attribute since it is recognized as one of the main factors that influences maintainability (Selic, 2003) (Otero & Dolado, 2004) (Reinhartz-Berger & Dori, 2005) (Genero et al., 2005; 2007). A UML class diagram must first be understood before any desired changes to it can be identified, designed, or implemented. In terms of the Lindland et al. framework, improving pragmatic quality means increasing the degree of correspondence between the set of statements in the model and the set of statements that the user thinks the model presents (i.e., their understanding of the model).