Multi-View Model-Driven Projection to Facilitate the Control of the Evolution and Quality of the Architecture

Salim Kadri, LASTIC Laboratory, Department of Computer Science, University of Batna II, Fesdis, Algeria
Sofiane Aouag, LASTIC Laboratory, Department of Computer Science, University of Batna II, Fesdis, Algeria
Djalal Hedjazi, LASTIC Laboratory, Department of Computer Science, University of Batna II, Fesdis, Algeria

ABSTRACT

In model driven-engineering, there is a myriad of approaches that use models and transformations to develop software systems. However, a few works in the literature have discussed the simplification of these models to make them more readable, understandable, and easy to navigate. This article proposed an approach that simplifies models in order to assist architects in controlling architecture evolution and quality, especially with large systems. This approach consists of two main concepts, multi-view modeling and on-demand model projection. In the former, formally specified models are divided into two views. The first one is dedicated to describe quality attributes. The second view represents the architectural view. The multi-viewing can enrich stakeholder reasoning about the developed architecture and simplify the mapping between quality attributes and architectural decisions at different abstraction levels. In the latter, the projection concept consists of extracting from source models only elements of interest to generate simpler and narrower models as output.

KEYWORDS
Alloy, Architecture Evolution, Multi-View, Projection, Quality Attributes, Quality, Software Architecture, Weaving

1. INTRODUCTION

Model driven engineering MDE (Arboleda & Royer, 2013) is a promising approach that uses models as first artifact to define software development methodologies, to develop systems at any abstraction level, and to organize and automate systems testing and validation. It allows specifying software systems by using models. These models could be progressively transformed (from abstract to more concrete) into executable applications for different platforms. Models can improve understandability, facilitate maintenance, and document software systems. However, due to some system’s high complexity, models can seem chaotic and difficult to understand by stakeholders. Many solutions in the literature have proposed to use views to describe software systems from the viewpoints of each member of these stakeholders. Philippe Kruchten (1995) in his paper “The 4+1 View Model of Software Architecture” had presented an approach for describing the architecture of software-intensive systems by the use of five main views: logical, physical, process, development, and scenario views. Each distinct view has its notation, abstraction, concern, and architectural pattern. In our work, the main concern is not just how to model the system’s architecture, but also how to control the evolution and the quality of
that architecture throughout the development process. The purpose of quality control is to achieve and to maintain quality attributes by ensuring conformance between the planned architecture and the implemented one (Chavez & Shen, 2013). The conformance can be attained by preserving architectural decisions and avoiding or limiting any deviation from the intentional architecture. The tight mutual coupling between architecture and quality attributes led us to propose models with two main views. The first view covers the system’s architecture and its adopted architectural styles and decisions. The second view covers the quality attributes promoted by those architectural decisions. These views are woven together to construct models at different abstraction levels to model architecture on one hand and to clarify the mapping between quality attributes and that architecture on the other hand.

A view is a sub-model that can show a subset of the model’s details, possibly with a transformation (Clements et al., 2003). All models in our research Work, regardless of their abstraction level, are specified formally with a general-purpose modeling language called Alloy (Jackson, 2012).

The multi-view paradigm can enhance stakeholders’ reasoning about the system and help them when performing activities like maintenance. However, with large systems, models become overcrowded and unclear, which makes such activities a challenging task. If the development team wants to maintain a component or a subsystem, they have to navigate through these fuzzy and oversized models to locate and understand the architecture of elements under maintenance, which is a tedious and time-consuming task. To overcome this issue, we have used a model projection to extract only elements of interest or only element intended to be maintained, which construct in the process a new simplified and easy-to-understand model that preserve the same multi-view architecture of the source model. The correctness of the produced models does not constitute a big issue with our projection approaches because we depend on the Alloy analyzer (Alloy Tools, 2020) to verify the consistency and the correctness of these models.

Our goals are to assist and facilitate architecture evolution control and enhancing development teams’ awareness about quality attributes at stake. Thus, our idea is to create customizable, simplified, and more understandable views-based models from other existed models using projection.

The remainder of this paper is organized as follows. Section 2 depicts multi-view models with architectural and quality views. Section 3 illustrates the model projection. Section 4 proposes a taxonomy for multi-view models according to three abstraction levels. Section 5 explains how to specify and check multi-view models with Alloy. Section 6 explains model projection with a case study of web auction application. Section 7 depicts some projection approaches. In section 8, we discuss some related work. The last section represents the conclusion and future works.

2. MULTI-VIEW MODELING

A software architecture is a complex entity that cannot be described in a simple one-dimensional fashion (Bass et al., 2012). A view is a representation of a set of system elements and relations among them. In this paper, we are interested in two fundamental dimensions, the quality dimension, and the architectural dimension. Consequently, modeling architecture should be based on two points of view. A view that deals with quality attributes and a view that deals with the architecture itself. In our case, the basic idea of multi-view modeling consists of creating a quality view and weave it with an architectural view in the same model, which results in multi-view (or bi-view) models that can represent the system at different abstraction levels (Figure 1). The transition between these levels is performed by the use of model projection and transformation.

As shown in Figure 1, the model is divided into two views or sub-models, which are quality view and architectural view. The quality view illustrates that a quality attribute QA can hold a set of sub-quality attributes. Each quality attribute can be measured by structural or dynamic metrics. The structural metrics are used to statically measure the architecture at the implementation phase; however, dynamic metrics are used to measure the system at the runtime phase. The quality attributes
Related Content

Agile Quality or Depth of Reasoning? Applicability vs. Suitability with Respect to Stakeholders’ Needs
[www.igi-global.com/chapter/agile-quality-depth-reasoning-applicability/5067?camid=4v1a](www.igi-global.com/chapter/agile-quality-depth-reasoning-applicability/5067?camid=4v1a)

Adaptive Modeling and Dynamic Targeting for Real Time Analytics in Mobile Advertising
[www.igi-global.com/article/adaptive-modeling-and-dynamic-targeting-for-real-time-analytics-in-mobile-advertising/190411?camid=4v1a](www.igi-global.com/article/adaptive-modeling-and-dynamic-targeting-for-real-time-analytics-in-mobile-advertising/190411?camid=4v1a)
The Case for Privacy Awareness Requirements
www.igi-global.com/article/the-case-for-privacy-awareness-requirements/152245?camid=4v1a

Modeling and Scheduling of Crude Oil Operations in Refinery: A Hybrid Timed Petri Net Approach
www.igi-global.com/chapter/modeling-scheduling-crude-oil-operations/76948?camid=4v1a