Dynamic Software Architectures

A Style-Based Modeling and Refinement Technique with Graph Transformations

Abstract

A good architectural design allows to capture the overall complexity of large, distributed systems at a higher level of abstraction. This is especially important for reconfigurable systems where the architectural configuration is subject to (constant) changes at runtime. When designing such a dynamic architecture, the software architect has to bring the functional business requirements and the available communication and reconfiguration mechanisms of the intended target platform in line.

As it is a complex task to incorporate these often diverging requirements into the architectural model, we propose a stepwise approach similar to the MDA initiative. We start with a platform-independent model capturing the business requirements and add platform-specific details in a later step. For each level of platform abstraction and associated platform, we define an architectural style which describes the characteristics of the platform. This way, conformance to the architectural style entails consistency with the underlying platform.

Besides run-time configurations of components and connections, architectural models also comprise the description of processes that control the communication and reconfiguration behavior. To provide operational semantics, architectural models are formalized as graphs, and architectural styles are formalized as graph transformation systems. UML is added as high-level modeling language on top, and profiles are used to adapt UML to certain architectural styles.

Due to the stepwise procedure, we also have to ensure the mutual consistency between models at different levels of abstraction. For this purpose, we define formal criteria which require that both structural and behavioral properties are preserved at the lower level of abstraction. Based on refinement relationships between abstract and platform-specific architectural styles, an algorithm allows to verify that all abstract, business-level behavior can also be realized in the platform-specific architecture and that no new behavior is added. These refinement techniques facilitate a stepwise, platform-consistent development of dynamic software architectures.