

Enhancing Aspect-Oriented Reusability with a Domain Engineering Approach

Alexander Gold

ABSTRACT

The exponential growth of data and information in recent years has made information and software systems essential parts of our life. However, it also causes a rapid increase of system complexity, while the requirements for reliability, ease of use, and performance have not been changed (or even became stricter). Software engineering offers different solutions to face these emerging challenges, one of which is aspect-orientation. Aspect-oriented software development aims at providing a new way of modularization according to which crosscutting concerns are separated from traditional units during the entire software development lifecycle. However, most of the works in this area mainly deal with general aspects that either fit all systems or are particularly developed for a specific system, limiting the reusability of these aspects. In this thesis, we suggest recruiting domain engineering in order to be able to define families of aspects and their weaving rules to families of applications during the design phase. In particular, we apply a specific domain engineering approach, called Application-based DOmain Modeling (ADOM), which separates between three levels of abstraction: applications, domains, and languages. In each level of abstraction, aspect and base models are specified, as well as the relevant weaving rules. An aspect model is comprised of three parts: a concern specification, which models the aspect itself, a match pattern, which defines the base models to which this aspect can be woven, and merge guidance, which contains rules for weaving the aspect to any base model that fulfills the match pattern. The merge guidance can further be specialized into different weaving models, each of which concentrates on the rules for weaving the aspect to a particular base model. Exemplifying the approach on UML 2.0 class and sequence diagrams, we found that the expressiveness of the approach promotes its support for evolvability, scalability, composability, and modularity, increasing the reusability, maintainability, and comprehensibility of its models.