A metamodeling approach for pattern specification and management

一种模式描述和管理的元模型方法

Citation: Liang Dou, Qiang Liu, Zong-yuan Yang, 2013. A meta-modeling approach for pattern specification and management. *Journal of Zhejiang University-Science C (Computers & Electronics)*, 14(10):743-755. [doi:10.1631/jzus.C1200040]

- The design patterns are widely adopted as a mechanism in disseminating the best practices. However, patterns are originally specified in the Unified Modeling Language (UML) diagrams and English annotations. They are intended to be read by humans, and not to be processed by machines

- The community proposes the dedicated Pattern Specification Language (PSL). But PSL is lacking some essential concepts such as Variable and Set. The inherent incompatibility between the formal methods and the modeling paradigm makes it extremely difficult to build a practical tool

- Therefore, interest has been growing in metamodeling design patterns, which holds potential for higher-level abstraction

- In this paper, we propose a metamodeling approach for pattern specification and management
- Each pattern is modeled as an EClass. Each EClass owns a set of EReferences for modeling the pattern participants. By using EReference, the concepts of Variable and Set are incorporated into our approach, which allows for specifying patterns in a more abstract manner
- In specifying the behavior of the pattern, we introduce another concept of dynamic relation, which is unavailable in UML. By explicitly defining the dynamic relations, the precise specification and preservation of the behavior of the pattern are feasible

We address three pattern-related problems, providing a way to “program at the design level”:

- Pattern instantiation: Our approach provides an easy way to define an EOperation in the EClass of a pattern, which creates UML elements, assigns EReferences, and ensures that the invariant is satisfied after execution
- Pattern evolution: Our solution defines a set of evolving operations in the EClass of a pattern. Each operation implements an evolving rule of the pattern and manipulates the legal pattern instances properly to make sure they are still legal after execution
- Pattern implementation: Our approach can programmatically generating Java Modeling Language specifications for pattern-level operations