ADVANCED AND RAPID DEVELOPMENT OF DYNAMIC ANALYSIS TOOLS FOR JAVA

Alex Villazón, Walter Binder, Danilo Ansaloni and Philippe Moret

ABSTRACT

Low-level bytecode instrumentation techniques are widely used in many software-engineering tools for the Java Virtual Machine (JVM), that perform some form of dynamic program analysis, such as profilers or debuggers. While program manipulation at the bytecode level is very flexible, because the possible bytecode transformations are not restricted, tool development based on this technique is tedious and error-prone. As a promising alternative, the specification of bytecode instrumentation at a higher level using aspect-oriented programming (AOP) can reduce tool development time and cost. Unfortunately, prevailing AOP frameworks lack some features that are essential for certain dynamic analyses. In this article, we focus on three common shortcomings in AOP frameworks with respect to the development of aspect-based tools - (1) the lack of mechanisms for passing data between woven advices in local variables, (2) the support for user-defined static analyses at weaving time, and (3) the absence of pointcuts at the level of individual basic blocks of code. We propose @J, an annotation-based AOP language and weaver that integrates support for these three features. The benefits of the proposed features are illustrated with concrete examples.

Keywords: Aspect-Oriented Programming, Aspect Weaving, Analysis at Weaving Time, Bytecode Instrumentation, Dynamic Program Analysis.