UCF Stands For Opportunity

Henrique Rebjlo

Federal University of Pernambuco

"An Aspect-Oriented Approach to implement JML Features"

Monday, November 17, 2008 • 4:00 p.m. • HEC (Harris Center) 101

The JML is used to specify designs of Java classes and interfaces. To this end, JML has a rich set of features for specifying methods, including specification inheritance. Thus, the most fundamental motivation for employing JML is to improve functional software correctness of Java applications, and helps to reduce corrective maintenance effort of those applications. In this talk, I will pre- sent a new JML compiler - ajmlc (AspectJ JML Compiler) - that generates aspects (AspectJ) for contract enforcement concern.

Mr. Rebjlo will also present some mapping mechanisms from JML to AspectJ. Another important point to note is that the classical JML compiler uses features not supported to other Java dialects, such as Java ME. In this way, he will also show how our compiler generates a instrumented bytecode that can run in standard Java SE as well as in constrained environments such as Java ME. Finally, he will talk about some conducted studies that compare the final code generated by ajmlc with the one produced by jmlc, and some future issues to investigate.

Mr. Henrique Rebjlo

Henrique Rebjlo received his MSc from University of Pernambuco (UPE), Brazil. Since undergraduate studies, he works with programming languages, including functional languages as well. In particular, he worked with the functional language Erlang. In 2006, he started his master and another research focus, behavioral interface specification languages (BISL) like Java Modeling Language (JML). During his master, he also started to study the benefits of aspect-oriented programing (AOP) when employed on OO systems. He focus is on AspectJ, an aspect-oriented extension do Java. In the middle to the end of his mastering (May 2008), he combined these languages (JML and AspecJ), which is his research focus currently, and designed a new JML compiler by replacing its back-end to a new one which generates AspectJ aspects to check programs written in Java SE and Java ME platforms.