

Specifying and Testing Software Components Using ADL

Sriram Sankar and Roger Hayes

Introduction by Alberto Savoia

Mission and business critical software must operate as specified, but ensuring that an implementation meets the specification has always been a challenge. There are two primary ways to determine if a program works as intended. The first one is to study the code and "prove" that the implementation does what the specification says. Since it takes even people with PhDs in math several days to generate a proof of correctness for even a trivial program, this approach is used very rarely. The second approach is to test the implementation of the software by creating and running a set of tests to exercise it. This approach is more practical, but most tests are developed manually and in a very ad-hoc manner, making it very difficult to determine if the set of tests is sufficiently thorough to provide the required level of confidence. The ADL project focused on bridging the huge gap between formal proofs of correctness and ad-hoc testing; its goal was to leverage the best of both worlds by borrowing formal specifications from the proof of correctness camp and executable tests from the traditional testing camp.

The key component, and one of the major innovations of the ADL project, was the ADL language (Assertion Definition Language). Unlike other formal specification languages, ADL was designed to be easily used and understood by average programmers (as opposed to mathematicians). ADL's syntax, for example, looks a lot like Java™ code, which minimizes the learning curve for most users. The other key component of the project was ADLT (ADL Translator). ADLT takes ADL specifications and generates self-checking tests that are used to verify the implementation.

ADL technology was developed concurrently with, and applied to, the Spring project. This was a particularly powerful combination, since a core component of Spring was the use of "contracts" to specify interface syntax. ADL was used to augment this syntax specification with a semantics section, which not only enabled automated test generation for Spring methods, but also increased the chances that the implementation would reflect the original intent of the methods.

When the Java™ programming language started to make waves, the ADL group thought that it would be a great help to Java's cause if early adopters had specialized testing tools for this new language. We also realized that a lot of the theory and technology behind ADL and ADLT could be applied to Java testing. Armed with some good arguments and lots of chutzpah, we enlisted the help of Bert Sutherland, Jim Mitchell, and Eric Schmidt, and got Scott McNealy's blessing for a new technology transfer model: starting a new business unit to develop and market technology developed by the lab. The new business unit, called SunTest™, grew to almost 50 people in over two years, and in addition to helping Java adoption, it generated almost six millions dollars in revenue in its first year of sales, after which it was absorbed into SunSoft™.

In addition to SunTest™, the results of the ADL projects have been used both at Sun and by standards organizations such as X/Open® (which was also involved in the project).

One of the most interesting aspects of the ADL project is that most of its funding (almost \$4M) came in the form of a research grant from Japan's MITI. Like many organizations that develop or utilize mission critical software, MITI was extremely concerned that the current state-of-the-art for software specification and testing was not adequate to provide the level of confidence that

they required. So, in order to advance the state-of-the-art, they contacted the top researchers in the field of software verification and testing, dangled a \$2M carrot in front of them, and asked them for proposals. Sun's proposal by the PrimaVera group beat all the others, including one from one of Sun's foes and competitors at the time, DEC. The first phase of the project was so successful that the project was extended and funded another \$2M.