Position Statement

Evolving Directions in Formal Methods

D. Richard Kuhn
National Institute of Standards and Technology
Gaithersburg, Maryland, USA

Abstract - *Formal methods have demonstrated their effectiveness in a number of application areas, but are still not widely used in the computing industry. Advances in theorem proving tools, particularly those combining model checking with traditional interactive proof techniques are reducing the cost of formal techniques. Although traditionally used for analyzing the correctness of specifications against requirements (and to a lesser extent the correctness of source code), formal methods can help reduce the cost of test generation, making formal methods more cost effective.*

Advances in software development technology have made it possible to build highly complex and sophisticated software at an ever increasing rate. Ensuring that software is dependable is a difficult problem, not only because of the size and complexity of the software, but also because source code is often not available for components that are acquired rather than developed in-house. Improved methods of assurance are essential for complex component-based systems.

Many observers have noted that the process of developing a formal specification is often as effective for finding errors as the verification effort in which the specification is to be used. Developing a formal specification requires a detailed and precise understanding of the system, which helps to expose errors and omissions. Yet despite their advantages, formal specifications are rarely used in practice.

Formal methods have been developed with the goal of allowing rigorous proof of system properties, a task that requires precise descriptions of systems. In practice, formal specifications are used to show system conformance to a set of formally stated requirements. (See [Craigen et al, 1993], [Clarke and Wing, 1996] for examples.) The specification can also be used in implementing the system in code. Thus the cost of developing a specification and proving its correctness must outweigh the cost of errors that might otherwise find their way into a released product.

But developing rigorous system tests also requires a precise, complete description of system functions, and practical system assurance requires testing, even when formal methods are used. Test development is typically an enormous expense, and may even exceed the cost of application development. Thus any increases in the efficiency of test development can have a significant impact on product cost.

To date, most research on automated software testing has focused on structural testing, i.e., testing based on execution paths within the code that implements a specified function. However, structural testing is not possible with many systems, as there is no access to source code. An alternative is to use specification-based testing, in which tests are derived from the specification alone [e.g., Weyuker et al., 1994; Tai, 1996].

Methods for generating tests from specifications can make formal methods cost effective for a much larger class of systems. At least in the United States, use of formal methods is largely confined to secure systems or safety-critical systems, i.e., those systems whose failure can have catastrophic cost. But if the high cost of formal methods can offset the possibly higher cost of test development, formal techniques become much more attractive. The past two decades have seen great advances in methods and tools for formal verification. More effective tools can do a great deal to increase the use of formal techniques in industry. In addition, better methods and tools for specification based testing could reduce the cost and increase the effectiveness of system testing.

References

Clarke, E.M., J.M. Wing, Formal Methods: State of the Art and Future Directions, ACM Computing Surveys, Dec. 1996.

Craigen, D., S. Gerhart, T. Ralston, An International Survey of Industrial Applications of Formal Methods, NIST GCR 93/626 (vols. 1 and 2).

Tai, K.C., Theory of Fault-Based Predicate Testing for Computer Programs, IEEE Trans. Software Eng., Vol. 22, No. 8, 1994.

Weyuker, E ., T. Gorodia, A. Singh, Automatically Generating Test Data from Boolean Expressions, IEEE Trans. Software Eng., Vol. 20, No. 5, 1994.