

## **CAREER:**

### **A Study of the Effectiveness of Using Formal Specifications to Create Assertions**

Matthew F. Curtis-Maury & Jennifer M. Haddock-Schatz

#### **PROJECT SUMMARY**

Though the software engineering industry is relatively new, software has become pervasive in modern society. Despite the fact that software is commonplace today, it often proves to be prone to error and generally undependable. When software does fail, the consequences can range from loss of revenue to loss of life.

Software testing is a common defense against such failures. The key goal of testing is to identify program faults and errors before the software is actually deployed. One mechanism that can be used in concert with testing is that of the *software assertion*. An assertion is a self check on a portion of a running program's data state. When used properly, assertions can provide runtime verification that a program is correct.

We propose a technique for assertion creation based on deriving assertions from a program's formal specification. The goal of this work is to show that deriving a set of assertions from a program's formal specification and then embedding these assertions in the program is a worthwhile approach for identifying program errors. In order to evaluate the effectiveness of specification derived assertions, we will apply our approach for assertion creation to a software system and its associated formal specification.

#### **Intellectual Merit**

The key issue addressed by this proposal is to develop a methodology for assertion creation and therefore improve the process of software testing. Though the benefit of using assertions to validate software has been touted by developers and researchers alike, in practice assertions are often used in a haphazard manner, if they are even used at all. In fact, to the best of our knowledge, no sound method for the creation of effective assertions exists in the literature. The result is that the assertions that are added to a program address low-level details that have little relationship to each other or the high level purpose of the system as a whole. By deriving assertions from a program's specification, assertions can be embedded into the code that reflect relationships between the various components of a system. We intend to show that these specification derived assertions can identify program faults more effectively than simple assertions created by the programmer during development.

An additional issue we are addressing is the conceptual "gap" that exists between a program and its specification. Creating specification based assertions necessitates a complete translation of the specification into program code. Thus, we hope the process which we employ to create these assertions will shed light on how a specification maps to its implementation.

#### **Broader Impact**

Because software failure is still very much an issue in today's society, there exists a need to continue to improve the practice of software testing. The work proposed here seeks to make such an improvement by providing a process for the creation of effective assertions. Through the research proposed we hope to demonstrate that utilizing a program's specification to derive assertions is a fruitful approach for identifying program faults. Additionally, by applying our approach to a real system we hope to identify patterns and rules for translating a specification into assertions. Finally, because there are so few examples of how to write meaningful assertions in the literature, we hope that the work proposed here can begin to offer insight into this area and hence potentially advance the state of the art in the practice of software testing.