Creating Reliable Programs from Unreliable Programmers

ALFRED V. AHO

Lawrence Gussman Professor of Computer Science

For Professor Alfred Aho, the question is simple: "How can we get reliable software from unreliable programmers?" The issue is more than academic. Aho can point to such high-profile fiascos as a $1 billion write-off for failed flight control software and hundreds of millions of dollars spent fixing an airport's automated baggage handling system.

In fact, a 2002 National Institute of Standards and Technology (NIST) study found that software defects cost the economy $60 billion annually and account for 80 percent of software development costs. Even then, Aho estimates that most commercial software has 1,000 to 10,000 defects per million lines of code.

"If you're developing a computer game, that doesn't matter much. But if you're programming a pacemaker, it's a matter of life and death," he said.

Aho's goal is to create a system that automatically tags potential problems. He hopes to do this by using the technology behind compilers, programs that translate easy-to-use programming languages like C into instructions a computer processor can understand.

When a compiler translates a program, it captures details about how it was built. Aho wants to compare this actual implementation with the program's technical specifications, which define such things as naming conventions, allowable operations, associations, data sets, and order of functions. This is similar to inspecting a building's structure, wiring, and plumbing against schematics and code.

Software, however, is more complex. "Let's say the source program has 1 million lines of code, and you want to look for all examples of addition," Aho explained. "It's written by several different people. Some may not have consulted the specification. They might use their own names for variables. Instead of writing 'add,' they might write it as 'plus.'"

Those subtle changes make it incredibly difficult to track errors. A plus function might use different data types than an add function, and produce unequal results. Or a programmer may discover a problem involving add functions, but fail to look for plus functions to see if the same problem exists.

"All large programs have a specification document that itemizes how the program should be written. I would like to specify a property from this document and test for its properties in the software. We already know how to create tools that do some of this in compilers. Now we want to extend these tools to software development," Aho said."This is a long-term project, but if we can make a small dent in software development and maintenance costs, we can save billions of dollars."

B.A.Sc., Toronto, 1963; M.A., Princeton, 1965; Ph.D., 1967