**Review of “A Practical Approach to Programming With Assertions”**

This article is talking about the Embedded Assertions, which is considered as a good tool for automatic runtime detection of software faults during debugging and testing. Although assertion has so many shining points, it is still not popular in programmers. The author concludes two reasons: 1) previous assertion processing tools didn’t integrate easily with existing programming environments; 2) it’s not well understood what kinds of assertions are most effective at detecting software faults.

To deal with the first problem, the author introduces us a tool, named APP (Annotation PreProcessor), which he has used for many years. According to his experience, this software can provide really good assertion support for C programs in UNIX-based programming environment, including flexibility in specifying how violated assertions are handled at runtime and how much or how little checking is to be performed each time a self checking program is executed. And also the author shows us the example of detection of faults with the help of APP to support his suggestion on this software.

For the second problem, the author implies that the issue comes up just because people don’t know how to use assertions. So he tries to make us some categories of assertions that can guard against many common kinds of faults and errors. He separates these categories with two big groups, Specification of Function Interfaces and Specification of Function Bodies. Function Interfaces. Specification of function interfaces is mainly designed for ensuring the arguments; return values and global state are valid with respect to the intended behavior of the function. And specification of function bodies is more likely to be a comment in function bodies to prevent faults.

I have to admit that the author does a good job in his article, his examples and code pieces are really easy to read and works well to explain his point of view. The software APP, like what the author says, it really makes assertions formal and easy to use, especially some of its special assertion, like assertion for subrange membership of data, can overcome the weakness of C program which doesn’t allow the specification of subrange constraints on numeric types. However, APP also has his problem. First, it’s a software based on UNIX environment and only for C program. Surely we can use C to program on UNIX environment, but some of us will still need a lot of efforts to change their habit to use that. Second, I think the software still need to improve its function to support more assertions for function bodies. Comparing to the assertions that mentioned for specification of function interfaces, the assertions for specification of function bodies are meaningless. Third, the example for APP at last doesn’t show us a good result for this “amazing” tool. Obviously, APP need to improve its categories of assertions and more important is to specify where is the right place to write assertion.

Anyway, this article is fine to read. The author broadens our mind of using assertions. It’s just need more efforts to get these in good order to be applicable.