Formal Specifications on Industrial-Strength Code—From Myth to Reality

(Invited Talk)

Manuvir Das

Center for Software Excellence
Microsoft Corporation
`manuvir@microsoft.com`

Abstract. The research community has long understood the value of formal specifications in building robust software. However, the adoption of any specifications beyond run-time assertions in industrial software has been limited. All of this has changed at Microsoft in the last few years. Today, formal specifications are a mandated part of the software development process in the largest Microsoft product groups. Millions of specifications have been added, and tens of thousands of bugs have been exposed and fixed in future versions of products under development. In addition, Windows public interfaces are formally specified and the Visual Studio compiler understands and enforces these specifications, meaning that programmers anywhere can now use formal specifications to make their software more robust.

How did this happen? The key ingredients of success were picking a critical programming error that costs software companies real money (buffer overruns), and building an incremental solution in which programmers obtain value proportional to their specification effort. The key technical aspects of this incremental approach include SAL, a lightweight specification language for describing memory access behaviour of C/C++ programs; espX, a heavyweight modular checker that enforces consistency between the code and the specification and validates memory accesses; and SALinfer, a lightweight global analysis that infers and inserts a large fraction of the memory specifications automatically.

The goal of this talk is to share the technical story of the insights that enabled SAL, espX and SALinfer, as well as the social and practical story of how we were able to move organizations with thousands of programmers to an environment where the use of specifications is routine.