Abolish Runtime Systems: Operating Systems Should Control the Execution Environment

James R. Larus
Microsoft Research
larus@microsoft.com

Abstract

Singularity [1] is a research project in Microsoft Research that started with a question: what would a software platform look like if it was designed from scratch with the primary goal of dependability? Singularity is working to answer this question by building on advances in programming languages and tools to develop a new system architecture and operating system (named Singularity), with the aim of producing a more robust and dependable software platform.

Singularity made some design decisions that distinguish it from other systems. First, Singularity is written, for the most part, in safe, managed code and it will only run verifiably safe programs. Second, the system is the runtime; there is no separate JVM or CLR. Third, each process's execution environment is independent, with its own, distinct runtime, garbage collector, and libraries. As a consequence, Singularity uses control of the execution environment as a mechanism to enforce system policy and enhance system dependability.

This talk will describe Singularity and then explain why conventional runtime systems, such as the JVM and CLR, should go away, like punch cards, teletypes, time sharing, etc.

Bio

James Larus is a Research Area Manager for programming languages and tools in Microsoft Research, where he manages the Advanced Compiler Technology, Human Interaction in Programming, Runtime Analysis and Design, and Software Reliability Research groups and leads the Singularity research project. He joined Microsoft Research as a Senior Researcher in 1998 to start and, for five years, lead the Software Productivity Tools (SPT) group, one of the most innovative groups in the area of program analysis and programming tools. Products of this group's research have shipped as products as well as being widely used within the company.

Before joining Microsoft, Larus was an Associate Professor of Computer Science at the University of Wisconsin-Madison, where he published approximately 60 research papers and co-led the Wisconsin Wind Tunnel (WWT) research project with Professors Mark Hill and David Wood. His research covered a number of areas: including new and far more efficient techniques for measuring and recording executing programs' behavior, tools for analyzing and manipulating compiled and linked programs, new programming languages, tools for verifying program correctness, and techniques for compiler analysis and optimization. Larus received his MS and PhD in Computer Science from the University of California, Berkeley in 1989, and an AB in Applied Mathematics from Harvard in 1980.

Reference

[1] Hunt, G., Larus, J., Abadi, M., Aiken, M., Barham, P., Fähndrich, M., Hawblitzel, C., Hodson, O., Levi, S., Murphy, N., Steensgaard, B., Tarditi, D., Wobber, T. and Zill, B. An Overview of the Singularity Project, Microsoft Research Technical Report MSR-TR-2005-135, October 2005.

VEE'06 June 14–16, 2006, Ottawa, Ontario, Canada.

ACM 1-59593-332-6/06/0006.