

Public Review: Software error early detection system based on run-time statistical analysis of function return values

Alex Deputovitch and Michael Stumm

Public Reviewer: Emre Kıcıman
Microsoft Research

It is unfortunate but true that anticipating the behaviors (and misbehaviors) of today's large computer software programs is near impossible. Complex software, new hardware devices and changing workloads all contribute to the difficulties of characterizing the correct behavior of a system. Even more difficult, however, is characterizing what the same system's incorrect behavior might be. Large software systems inevitably have bugs and anticipating and characterizing their possible misbehaviors of a system would help us quickly detect problems as they manifest. A good monitoring systems would even raise the possibility of catching and responding to issues early, before they cascade and cause massive failures.

This paper attacks this grand challenge head-on, advocating a monitoring approach that fundamentally accepts that software is poorly understood, and asks the question "what can we do if we must assume that we do not understand the system?" The advocated approach is to avoid monitoring difficult-to-characterize program-specific assertions of what might be good or bad behavior, and instead capturing and analyzing broad classes of micro-parameters, such as function return values, queue lengths, and performance metrics. Each class of micro-parameters is analyzed and used to generate a single macro-parameter, interpreted as a generic indicator of the system's health.

In this paper, the example class of micro-parameters are function return values, and the resultant macro-parameter is the rate of function calls returning errors. In this example, the paper is very similar to the DIDUCE project by Hangal et al., but effectively generalizes that work by considering the behavior of the program over time: a critical element of interpreting a macro-parameter is to compare its time-varying behavior to the behaviors of the program during believed-correct operation.

Of course, many open questions remain. As suitable for a workshop, this paper describes research still in its early stages. It will be interesting to see how well this framework generalizes in practice to different classes of micro-parameters as well as to a broader range of test systems and

workloads. An interesting question for this workshop on autonomic computing is whether remedial actions can be automatically taken in response to macro-parameters that indicate problems: will the advantages of taking quick action to fix problems outweigh the potential for mistaken action in response to false alarms? Will such automated responses make the overall behavior of the system more complex or will macro-parameters be reliable enough that automated responses will help systems systematically avoid unpredictable behaviors and failures?