Debugging and Performance Monitoring for Distributed Systems: The Software Angle

Zary Segall
Carnegie-Mellon University

Software engineering is emerging as a discipline for organizing the life cycle of software systems. The software life cycle includes a number of phases, such as, requirements, specification, design, testing, integration, validation and verification (V&V) and maintenance. A software package is successful when it fulfills its functional and performance requirements.

The issue this position paper is raising is the role of monitoring in software performance V&V. We are interested in applications where software correctness depends on its performance characteristics. In other words, we are interested in applications where a correct program with certain performance characteristics is the end goal. Such applications include between others, real-time programming, distributed/parallel computing and supercomputing.

One could call the theory and practice of designing such systems, the discipline of Software Performance Engineering. There is no question that such discipline is critically needed and the fact that substantial effort is needed to sort, integrate the existing technology and to create badly needed additional knowledge. The monitoring system could perform a significant part of this process.

Point in fact is the PIE system at CMU. PIE is an integrated technology for distributed/parallel programs V&V, based on automatic instrumentation and monitoring.

The PIE environment consists of a set of integrated tools for performance efficient distributed/parallel programming. Central to this environment is the concept of performance degradation prevention, detection and avoidance. Performance prevention is the process of predicting, before completion of the implementation process, the performance of a parallel algorithm on a specific parallel architecture. Performance degradation detection are the set of techniques applied after the coding process (also known as performance debugging). Performance degradation avoidance is a run-time process consisting in restructuring dynamically the application in the presence of predicted or detected performance degradations.