Chapter 39
Data Mining System Execution Traces to Validate Distributed System Quality-of-Service Properties

James H. Hill
Indiana University-Purdue University Indianapolis, USA

ABSTRACT

System Execution Modeling (SEM) tools enable distributed system testers to validate Quality-of-Service (QoS) properties, such as end-to-end response time, throughput, and scalability, during early phases of the software lifecycle. Analytical capabilities of QoS properties, however, are traditionally bounded by a SEM tool's capabilities. This chapter discusses how to mine system execution traces, which are a collection of log messages describing events and states of a distributed system throughout its execution lifetime, generated by distributed systems so that the validation of QoS properties is not dependent on a SEM tool's capabilities. The author uses a real-life case study to illustrate how data mining system execution traces can assist in discovering potential performance bottlenecks using system execution traces.

INTRODUCTION

Enterprise distributed systems, such as mission avionic systems, traffic management systems, and shipboard computing environments, are transitioning to next-generation middleware, such as service-oriented middleware (Pezzini & Natis, 2007) and component-based software engineering (Heineman & Councill, 2001). Although next-generation middleware is improving enterprise distributed system functional properties (i.e., its operational scenarios), Quality-of-Service (QoS) properties (e.g., end-to-end response time, throughput, and scalability) are not validated until late in the software lifecycle, i.e., during system integration time. This is due in part to the *serialized-phasing development* problem (Rittel & Webber, 1973).

DOI: 10.4018/978-1-4666-2455-9.ch039

As illustrated in Figure 1, in serialized-phasing development, the infrastructure and application-level system entities, such as components that encapsulate common services, are developed during different phases of the software lifecycle. Software design decisions that affect QoS properties, however, are typically not discovered until final stages of development, e.g., at system integration time, which is too late in the software lifecycle to resolve performance bottlenecks in an efficient and cost effective manner (Mann, 1996; Snow & Keil, 2001; Woodside, Franks, & Petriu, 2007).

System Execution Modeling (SEM) tools (Smith & Williams, 2001), which are a form of model-driven engineering (Schmidt, 2006), assist distributed system developers in overcoming the serialized-phasing development problem shown in Figure 1. SEM tools use domain-specific modeling languages (Ledeczi, Maroti, Karsai, & Nordstrom,1999) to capture both platform-independent attributes (such as structural and behavioral concerns of the system) and platform-specific attributes (such as the target architecture of the system) as high-level models. Model interpreters then transform constructed models into source code for the target architecture. This enables distributed system testers to validate QoS properties continuously throughout the software lifecycle while the "real" system is still under development. Likewise, as development of the real system is complete, distributed system testers can incrementally replace *faux* portions of the system with its real counterpart to produce more realistic QoS validation results.

Although SEM tools enable distributed system developers and testers to validate distributed system QoS properties during early phases of the software lifecycle, QoS validation capabilities are typically bounded to a SEM tool's analytical capabilities. In order to validate QoS properties unknown to a SEM tool, distributed system testers have the following options:

- **Use handcrafted solutions:** This option typically occurs outside of the SEM. Moreover, this option is traditionally not applicable across different application do-

Figure 1. Overview of serialized-phased development in distributed systems