

Model-Based Validation for Dealing with Operator Mistakes

Kiran Nagaraja, Andrew Tjang, Fábio Oliveira, Ricardo Bianchini, Richard P. Martin, Thu D. Nguyen

{knagaraj, atjang, fabiool, ricardob, rmartin, tdnguyen}@cs.rutgers.edu

Department of Computer Science, Rutgers University, Piscataway, NJ 08854

Online services are rapidly becoming the supporting infrastructure for numerous users' work and leisure, placing higher demands on their availability and correct functioning. Increasingly, these services are comprised of complex conglomerates of distributed hardware and software components. Added to this complexity, these services evolve quite frequently accumulating considerable heterogeneity within them, while allowing little time for their in-depth understanding by service personnel. Thus, it is not surprising that mistakes by service operators are common, and have been deemed to be the primary cause of service downtime.

Responses from an extensive survey we conducted of network and system administrators further substantiate this problem. Among common failures reported by administrators, 43% of them (37 of 87 reported) can be attributed to operator actions, with software and hardware being the other prominent causes. The results show that mistakes happen due to a variety of reasons including, the lack of understanding of system functioning by operators, complexity of the maintenance task, inattentiveness on the part of the operator, and lack of appropriate tools to help in operations. While offline testbeds should help check some of these mistakes, 56% of the 41 respondents were dissatisfied with such environments since they differ considerably from the online system.

These findings point to the requirement for effective system support to carry out mistake-free service maintenance. Previously, we proposed an approach for operators to check the correctness of their actions in a virtual sandbox environment that extends the online system [1]. We experimented with two techniques within a prototype implementation of the environment for a multi-tier online service. The prototype caught two-thirds of all mistakes we observed in human-factor experiments using volunteer operators. Our techniques relied on comparing the behavior (i.e., message flow) under test with a known correct behavior, in the form of traces or an online replica. Such comparison techniques, however, cannot be applied in the face of behavior altering maintenance operations. In such instances, the service operator often relies on a mental model of the way the system behaves to verify its correctness. What constitutes such mental models? How can they be succinctly, simply, and accurately represented in a systematic manner? And, how can the system be validated as per these models? These questions are at the core of the problem we address here.

In this work, we propose a model-based approach to specify the system's *operational model* or behavior in the form of assertions, and a runtime to evaluate and enforce such assertions. Assertions may be specified by service designers well-versed with the system's characteristics and their interactions. For example, consider the working of a load-balancer component. It implements various scheduling algorithms, but a salient property is to achieve uniform resource utilization across the receiving components. Then, its correctness can be tested by asserting an equality of either the flow of messages to each receiving node, or of their resource utilization.

We have designed a declarative language that maps language objects closely to service components, simplifying the specification of the expected model. The model is represented as a dynamic set of assertions on performance (e.g. throughput) and structural (e.g., connectivity) aspects of a service, and their change across discrete events (e.g., adding a new node). Assertion sets or programs, can then assert expected system behavior both under normal and maintenance operations that modify service behavior. Programs execute over a runtime that provides support for evaluating assertions and enforcing a limited set of actions associated with failed assertions. The runtime gathers and analyzes low-level data and events from a monitoring framework, providing statistical processing of the time-series data. Overall, assertions can be written in a flexible manner to describe system behavior at varying degrees of accuracy, correlating information both across time and other components within the service. Using the extensive data about operator practices and mistakes gathered from our survey mentioned above, we are currently evaluating the model-based approach to detect and mask operator mistakes both within the validation environment and the online service itself. Some open questions for this work are: What is the right level at which operator actions should be checked? What if operators deviate from expected operational paths associated with corresponding system behaviors? Will language-based model specification lead to more mistakes? And, can such languages lend themselves to convenient and accurate laying down of complex mental models maintained by service personnel?

References

- [1] K. Nagaraja, F. Oliveira, R. Bianchini, R. P. Martin, and T. D. Nguyen. Understanding and Dealing with Operator Mistakes in Internet Services. In *Proceedings of the 6th USENIX Symposium on Operating Systems Design and Implementation (OSDI'04)*, Dec. 2004.