The Maintenance and Evolution of Field-Representative Performance Tests

Mark D. Syer
Software Analysis and Intelligence Lab
School of Computing, Queen's University, Canada
mdsyer@cs.queensu.ca

***Abstract*—The rise of large-scale software systems poses new challenges for the software performance engineering field. Failures in these systems are typically associated with performance issues, rather than with feature bugs. Therefore, performance testing has become essential to ensuring the problem-free operation of these systems. However, the performance testing process is faced with a major challenge: evolving field workloads and deployment configurations, often lead to "outdated" tests that are not reflective of the field. Hence performance analysts must maintain their field-representative tests by evolving these tests. Therefore, we propose new approaches for maintaining and evolving field-representative performance tests. We believe that field-representative tests can be autonomically maintained and evolved, i.e., the configurable parameters of a performance test can be optimized, such that the resulting test exercises the system in a similar manner as the field. Such approaches should help performance analysts to ensure the "realism" of their performance testing efforts, resulting in higher quality systems.**

Keywords*—Performance Engineering; Performance Testing; Maintenance and Evolution;

I. Introduction

Software Performance Engineering (SPE) represents the entire collection of software engineering activities and related analyses used throughout the software development cycle, which are directed to meeting performance requirements [1]. These performance requirements describe the responsiveness (i.e., the ability to respond to user demands in a timely manner) and scalability (i.e., the ability to retain responsiveness despite increasing user demand) of a software system [2]. SPE is particularly relevant to today's large-scale software systems (e.g., Amazon.com and Google's GMail) because failures in these systems are typically associated with performance issues, rather than with feature bugs [3], [4]. These performance issues have led to several high-profile failures, incusing the NASDAQ's initial public offering of Facebook [5] and the United States government's roll-out of healthcare.gov [6], with significant financial and reputational repercussions [7].

Performance testing is the primary SPE approach used to prevent these failures [1]. Performance analysts perform performance tests that study the behaviour of a System Under Test (SUT) under various workloads. Such tests are designed to satisfy a particular test objective (e.g., determining the maximum operating capacity of a SUT, validating non-functional performance requirements or uncovering bottlenecks).

The goal of performance testing is to examine how the system behaves under realistic workloads. Therefore, performance tests are composed of a test workload and a configuration of the SUT that are reflective of the field.

Test workloads are usually derived from the field (i.e., beta testing data or actual production data). However, field workloads continuously evolve as the user base changes and as feature preferences change. Evolving field workloads often lead to "outdated" tests that are not reflective of the field [8], [9], yet these workloads have a major impact on the performance of the system [4], [10].

Configuration parameters, such as the max size of thread pools, the number of concurrent database connections or the spin up time of virtual machines, are also usually derived from the field. These configuration parameters have a major impact on the performance of the SUT. However, not all configuration parameters are readily available. For example, some configuration parameters maybe outside the system itself (e.g., virtual-machines and/or hosting infrastructure).

In this thesis, we propose new approaches to maintain and evolve field-representative performance tests. First, we propose an approach to determine whether a test is field-representative. We then propose an approach to autonomically evolve field-representative tests. Such approaches should help performance analysts to improve their performance testing efforts, resulting in higher quality systems.

References

[1] M. Woodside, G. Franks, and D. C. Petriu, "The future of software performance engineering," in *Proceedings of the Future of Software Engineering*, May 2007, pp. 171–187.

[2] L. Williams and C. Smith, *Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software*, 1st ed. Addison-Wesley Longman Publishing Co., Inc., 2002.

[3] E. Weyuker and F. Vokolos, "Experience with performance testing of software systems: issues, an approach, and case study," *Transactions on Software Engineering*, vol. 26, no. 12, pp. 1147–1156, Dec 2000.

[4] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, no. 2, pp. 74–80, Feb 2013.

[5] P. Ausick, "NASDAQ gets off cheap in Facebook IPO SNAFU," http://finance.yahoo.com/news/nasdaq-gets-off-cheap-facebook-174557126.html, 2013, Last Accessed: 05-May-2014.

[6] J. Bataille, "Operational progress report," http://www.hhs.gov/digitalstrategy/blog/2013/12/operational-progress-report.html, 2013, Last Accessed: 01-Jun-2014.

[7] Coleman Parkes, "The avoidable cost of downtime," http://www.ca.com/~/media/Files/SupportingPieces/acd_report_110110.ashx, 2011, Last Accessed: 14-Apr-2014.

[8] L. Bertolotti and M. C. Calzarossa, "Models of mail server workloads," *Performance Evaluation*, vol. 46, no. 2-3, pp. 65–76, Oct 2001.

[9] J. Voas, "Will the real operational profile please stand up?" *IEEE Software*, vol. 17, no. 2, pp. 87–89, Mar 2000.

[10] Z. Zhang, L. Cherkasova, and B. T. Loo, "Benchmarking approach for designing a mapreduce performance model," in *Proceedings of the International Conference on Performance Engineering*, Apr 2013, pp. 253–258.