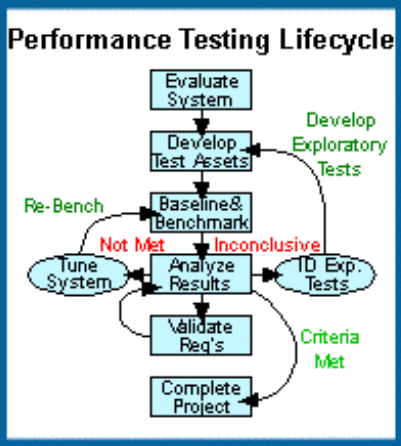
User Experience, not Metrics

Effective Performance Testing



By Scott Barber

www.perftestplus.com

Part 1: Introduction

(This item originally appeared on The Rational Developer Network, an online community for customers of IBM Rational Software. To find out more, including how you can get a free evaluation to the Rational Developer Network, please visit http://www.rational.com/services/rdn/find_out_more.jsp)

With so many professionals online and relying on the Internet to perform daily operations, application performance has become vital to the success of an eBusiness solution. In an effort to ensure success, many companies have developed tools and methodologies to test and tune applications for performance. These tools and methodologies have focused around optimizing system metrics, rather than optimizing user experience. The *User Experience, Not Metrics* Series of articles will address topics related to determining true user experience and application performance tuning using Rational Suite TestStudio coupled with a proven methodology of end-to-end Performance Engineering.

Introduction

How many times have you surfed to a website to accomplish a task only to give up and go to a different website because the home page took too long to download? "46% of consumers will leave a preferred site if they experience technical or performance problems." (Juniper Communications) In other words, "If your website is slow, your customers will go!" This is a simple concept that all Internet users are familiar with. When this happens, isn't your first thought always, "Gee, I wonder what the throughput of the web server is?" Well no, that is certainly not the thought that comes to mind. Instead, you think "Man, this is SLOW! I don't have time for this. I'll just find it somewhere else." Now consider this, what if it was YOUR website that people were leaving because of performance?

Face it, users don't care what your throughput, bandwidth or hits per second metrics prove or don't prove, they want a positive user experience. There are a variety of books on the market, which discuss how to engineer maximum performance. There are even more books that focus on making a website intuitive, graphically pleasing and easy to navigate. The benefits of speed are discussed, but how does one truly predict and tune an application for optimized user experience? One must test, first hand, the user experience! There are two ways to accomplish this. One could release a website straight into production, where data could be collected and the system could be tuned, with the great hope that the site doesn't crash or isn't painfully slow. The wise choice, however, would be to simulate actual multi-user activity, tune the application and repeat (until the system is tuned) before placing your site into production. Sounds like a simple choice, but how does one simulate actual multi-user activity accurately? That is the question this series of articles attempts to answer.

Terms and Concepts

Understanding the terms and concepts below is critical to getting the most out of this series of articles.

Performance Engineering: an extension of Load, Spike and Stress Testing, encompassing the performance validation and tuning of systems and applications. Activities focus on ensuring acceptable performance for the end users of the application being deployed. See Figure 1.

Workload Distribution: a representation of the functions performed by a user community on a system, sometimes known as a User Community Model. For example, during the course of a day on a retail-based website, most users are shopping, some are searching for a specific product, some are finalizing purchases and checking out, while a single administrator may be updating product prices. A workload distribution is based on a percentage of users performing a specific function over a given period of time. Using the above example a workload distribution could be: shopping – 83%, searching - 5%, checking out – 10% and administration - 2%.

Performance Requirements: are expressed as a maximum allowable response time or component measurement under pre-determined conditions. Requirements MUST be achieved for the system under test to be promoted into production.

When viewed from a user-experience perspective, Performance Goals are those criteria that are desired for the application which are in some way different than the previously stated requirements.

Response Time is measured from the end-user perspective, the time elapsed between when a request is made and when that request is fulfilled. May occur on any tier or combination of tiers of the system. Commonly the time from when a web browser has finished sending a request to when it starts receiving the response from the web application.

Session Duration is the total amount of time a single user is using the system during a single site visit (expressed in fractions of an hour). Hourly users divided by average session duration results in a heavily averaged estimated of concurrent usage.

Concurrent Usage is a “Dangerously misleading statistic” representing the total number of overlapping users who are actually accessing the system or who have active sessions at a specific instant in time.

Total Hourly Usage is the number of users accessing the system in a given hour.

Baselines, in this context, are single user, single script tests that are recorded to provide a starting point for time comparisons.

A User Delay is a wait time incorporated into a script so that when that script is played back it plays at the same pace as an actual user.

This series of papers may use some common terms that have different meanings to different people. These terms are defined below for the context of this series.

Load Test: A load test is a multi-user test that accurately simulates the expected user community, including user delays. These tests may be executed with differing user loads to find information such as the maximum number of users a system can support while still meeting the stated performance goals.

Stress Test: A stress test is any combination of scripts that are played back at a high user load excluding user delays. These types of tests are useful in determining system stability, and functionality under load, but are NOT valid for determining user experience.

Benchmarks: are generally metrics gathered about system hardware, and supporting software, but NOT application code. For instance, Web server throughput and hits per second when accessing a large graphic can be determined through benchmark testing.

Performance Benchmarks: or a light load scenario is generally a small community of users compared to the target load. This community of users much be large enough to approximate a reasonable sample of the entire user community model while still being significantly smaller than the expected system capacity, 15% of total expected user load is generally a good Benchmark volume. Executing Benchmark tests ensures that the testing environment behaves as expected under light load as well as validates that the scripts have been developed correctly.

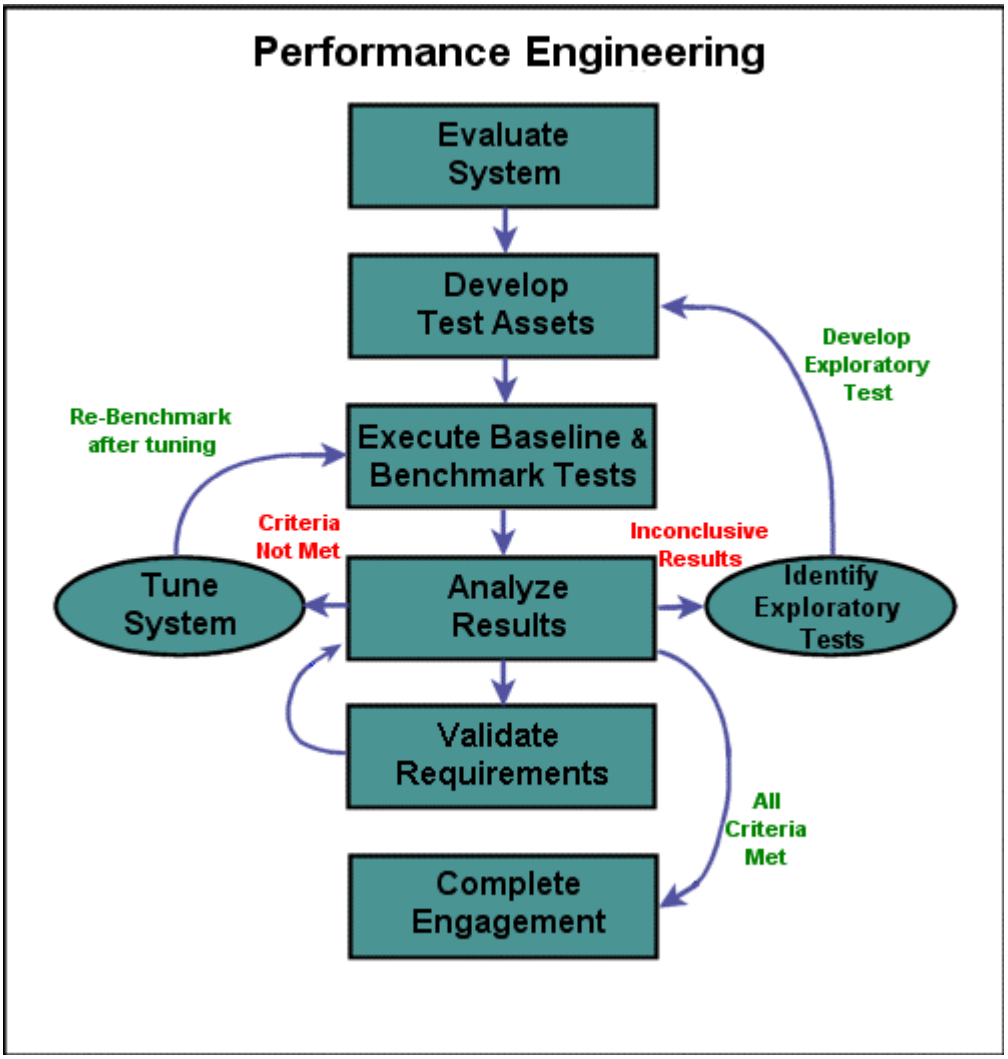


Figure 1: Common Performance Engineering Process Flow

Overview of this Series

The "User Experience, Not Metrics" series will have a new article submitted monthly. All articles will include discussions of the practical applications of the methodology/technique being introduced, real world examples and/or code samples and "Now you try it." exercises. Each article will be identified as Beginner, Intermediate or Expert level. This level generally refers to the complexity of the code required to accomplish the technique being discussed. The concepts presented in each article will be applicable at all levels of expertise. The first 12 articles have already been identified and are described briefly in the following sections.

Modeling Real Users

One of the keys in determining true user experience is to effectively model actual users and user communities. Most performance tuning approaches today do not account strongly enough for either the distribution of tasks across entire user communities, or the high level of randomness among actual users. The first three articles in the series discuss how to use Rational Test Studio to accurately model both individual users and entire user communities from the application's perspective. Specific topics include:

1. Modeling Individual User Delays
2. Modeling Individual User Patterns
3. Modeling Groups of Users

Meaningful Times

After modeling actual users, it is imperative to capture the actual system/application response time from the perspective of those users. Simply capturing those times is not sufficient. The times are useless unless the patterns of those times can be interpreted. The next three articles discuss how to use Rational Test Studio to capture and interpret true experience times. Specific topics include:

1. What should I time and where do I put my timers?
2. What is an outlier and how do I account for one?
3. Consolidating and interpreting Times

Reports to Stakeholders

As much as I hate to admit it, stakeholders and decision makers need reports on results. I keep trying to convince my clients that all they need from me at the end of a Performance Engineering engagement is a Post-It note with either the words "Go Live", or "Don't" written on it, but they don't seem to think that provides enough value. If you have clients similar to mine, you'll be required to take the vast amount of data collected from a Performance Engineering effort (often several Gigabytes) and consolidate it into concise, yet meaningful report. The articles in this section will discuss what types of tests provide the most value to managers and decision makers as well as how to use the data collected from Rational's Test Manager reports to create multiple run summaries. Specific topics include:

1. What Tests add value to stakeholders?
2. Summarizing across multiple tests with accuracy
3. Creating a Degradation Curve

Advanced Topics

The final group of topics in this series will focus around specific advanced issues that have caused stress to the authors. These articles take the format of case studies. Each case study outlines the specific need of the (unnamed) client in question, the author's thought process to developing a solution, an outline of the potential solutions, and a detailed description of the selected solution. Specific topics include:

1. Handling Secure Session ID's
2. Conditional user path navigation (intelligent surfing)
3. Working with Unrecognized Protocols

Summary

The lesson in this introduction to *The User Experience, Not Metrics* article series is unmistakable; a user's point-of-view is a more reliable measure of website performance than today's customary metrics. This series of articles is designed to teach how multi-user activity can be simulated using Rational's TestStudio and Noblestar's proven Performance Engineering Methodology. The articles promise to share valuable information about the how the methodology works and how the Rational toolset is utilized. The articles will even divulge useful tips in getting around those issues that have stumped the experts. I hope your interest has been piqued and that you will return next month for your first dose of *The User Experience, Not Metrics* article series.

About the Author

Scott Barber is a System Test Engineer and Quality Assurance Manager for AuthenTec, Inc. and a member of the Technical Advisory Board for Stanley-Reid Consulting, Inc. With a background in network architecture, systems design, database design and administration, programming, and management, Scott has become a recognized thought leader in the context-driven school of the software testing industry. Before joining AuthenTec, he was a consultant specializing in Performance Testing/Analysis, a Company Commander in the United States Army, a DBA and a Government Contractor in the transportation industry.

Scott is a co-founder of WOPR (the Workshop on Performance and Reliability), a semi-annual gathering of performance testing experts from around the world, a member of the Context-Driven School of Software Testing and a signatory of the Agile Manifesto. He is a Discussion Facilitator in the Rational Developer Network public forums and a moderator for the Performance Testing and Rational TestStudio related forums on QAForums.com. Scott's Web site complements this series. Please visit it to find more detail on some topics and view slides from various presentations he's given recently. You can address questions/comments to him on either forum or contact him directly via e-mail.

AuthenTec, Inc. is a leading semiconductor company providing advanced biometric fingerprint sensors to the PC, wireless, PDA, access control and automotive markets. AuthenTec's FingerLoc and EntréPad product families utilize the Company's patented TruePrint™ technology, the first technology capable of imaging everyone under virtually any condition.

Stanley Reid Consulting, Inc. is a small, niche consulting company that focuses on IT organizational improvement and recruiting and staffing of technical experts. Stanley Reid Consulting helps your team achieve consistent, successful delivery of IT projects through the following services: IT Organizational Improvement Consulting, Expert Supplemental Staffing, Technical Recruiting & Placement.