International Journal of Scientific Engineering and Research (IJSER)
www.ijser.in
ISSN (Online): 2347-3878
Volume 2 Issue 1, January 2014

Web Performance Testing: Methodologies, Tools and Challenges

Vinayak Hegde[1], Pallavi M S[2]

[1]Assistant .Professor, Dept. of Computer Science, Amrita Vishwa Vidyapeetham, Mysore Campus, India
[2]Lecturer, Department of Computer Science, Amrita Vishwa Vidyapeetham, Mysore Campus, India

Abstract: ***The Internet is gently becoming the essential requirement for the business world, with web applications as the brains. This means that software faults in web applications have potentially disastrous consequences. Most work on web applications has been on making them more powerful, but relatively little has been done to ensure their quality. Important quality attributes for web applications include reliability, availability, interoperability and security. Web Performance testing is a type of testing intended to determine the responsiveness, reliability, throughput, interoperability and scalability of a system and /or application under a given workload. It could also be defined as a process of determining the speed or effectiveness of a computer, network, software application or device. Testing can be conducted on software applications, system resources, targeted application components, databases and a whole lot more. It normally involves an automated test suite as this allows for easy, repeatable simultaneous a variety of normal peak and exceptional load conditions. Such forms of testing help verify whether a system or application meets the specifications claimed by its vendor. This paper emphasis on methodology of performance testing and explains about various diagnostic tools to implement testing to overcome from single point of failure. This paper also explains about challenges of web performance testing and helps for one who takes up further research activity.***

Keywords: Testing Tool, Methodology, Open source, speed, scalability, stability

1. Introduction

Testing is major component of any software engineering process meant to produce high quality application. Testing aims at finding errors in the tested object and giving confidence in its correct behavior by executing the tested object with input values [1]. Web applications are the fastest growing classes of software systems today. Web applications are being used to support wide range of important activities: business transaction, scientific activities like information sharing, and medical systems such as expert system-based diagnoses. Web applications have been deployed at a fast pace and have helped in fast adoption but they have also decreased the quality of software. Therefore, all entities of web application must be tested. In order to make web based application to be widely and successfully adopted, testing methodologies must be flexible, automatic, and be able to handle their dynamic nature [2].

Performance testing is defined as the technical investigation done to determine or validate the speed, scalability, and/or stability characteristics of the product under test. Performance-related activities, such as testing and tuning, are concerned with achieving response times, throughput, and resource-utilization levels that meet the performance objectives for the application under test. [3] Performance testing is commonly conducted to accomplish the following:

a) Assess production readiness
b) Evaluate against performance criteria
c) Compare performance characteristics of multiple systems or system configurations
d) Find the source of performance problems
e) Support system tuning
f) Find throughput levels

Performance testing is typically done to help identify bottlenecks in a system, establish a baseline for future testing, support a performance tuning effort, determine compliance with performance goals and requirements, and/or collect other performance-related data to help stakeholders make informed decisions related to the overall quality of the application being tested. [3] In software engineering, performance testing is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage. Performance testing is a subset of performance engineering, an emerging computer science practice which strives to build performance into the implementation, design and architecture of a system. [4]

For a performance testing project to be successful, both the approach to testing performance and the testing itself must be relevant to the context of the project. Without an understanding of the project context, performance testing is bound to focus on only those items that the performance tester or test team assumes to be important, as opposed to those that truly are important, frequently leading to wasted time, frustration, and conflicts.

The project context is nothing more than those things that are, or may become, relevant to achieving project success. This may include, but is not limited to:

a) The overall vision or intent of the project
b) Performance testing objectives
c) Performance success criteria
d) The development life cycle
e) The project schedule
f) The project budget
g) Available tools and environments
h) The skill set of the performance tester and the team

i) The priority of detected performance concerns
j) The business impact of deploying an application that performs poorly [3]

2. Performance Testing Methodology

2.1 Activities of Performance Testing

Performance testing is typically done to help identify bottlenecks in a system, establish a baseline for future testing, support a performance tuning effort, determine compliance with performance goals and requirements, and/or collect other performance-related data to help stakeholders make informed decisions related to the overall quality of the application being tested.

Figure 1: Core performance testing

2.1.1 Identify the Test Environment

Identify the physical test environment and the production environment as well as the tools and resources available to the test team. The physical environment includes hardware, software, and network configurations. Having a thorough understanding of the entire test environment at the outset enables more efficient test design and planning and helps you identify testing challenges early in the project. In some situations, this process must be revisited periodically throughout the project's life cycle.

The key factor in identifying the test environment is to completely understand the similarities and differences between the test and production environments. Some critical factors to consider are:

- **Hardware:** Configurations, Machine hardware (processor, RAM, etc.)
- **Network :** Network architecture and end-user location, Load-balancing implications, Cluster and Domain Name System (DNS) configurations
- **Tools :** Load-generation tool limitations, Environmental impact of monitoring tools
- **Software :** Other software installed or running in shared or virtual environments Software license constraints or differences, Storage capacity and seed data, Volume, Logging levels
- **External factors :** Volume and type of additional traffic on the network, Scheduled or batch processes, updates, or backups, Interactions with other systems

2.1.2 Identify Performance Acceptance Criteria

Identify the response time, throughput, and resource utilization goals and constraints. In general, response time is a user concern, throughput is a business concern, and resource utilization is a system concern. Additionally, identify project success criteria that may not be captured by those goals and constraints;

Classes of characteristics that frequently correlate to a user's or stakeholder's satisfaction typically include:

- **Response time.** For example, the product catalog must be displayed in less than three seconds.
- **Throughput.** For example, the system must support 25 book orders per second.
- **Resource utilization.** For example, processor utilization is not more than 75 percent. Other important resources that need to be considered for setting objectives are memory, disk input/output (I/O), and network I/O.

2.1.3 Plan and Design Tests

Identify key scenarios, determine variability among representative users and how to simulate that variability, define test data, and establish metrics to be collected. Consolidate this information into one or more models of system usage to be implemented, executed, and analyzed.

2.1.4 Configure the Test Environment

Prepare the test environment, tools, and resources necessary to execute each strategy as features and components become available for test. Ensure that the test environment is instrumented for resource monitoring as necessary.

2.1.5 Implement the Test Design

Develop the performance tests in accordance with the test design.

Consider the following key points when implementing the test design:

- Ensure that test data feeds are implemented correctly. Test data feeds are data repositories in the form of databases, text files, in-memory variables, or spreadsheets that are used to simulate parameter replacement during a load test.
- Ensure that application data feeds are implemented correctly in the database and other application components. Application data feeds are data repositories, such as product or order databases, that are consumed by the application being tested. The key user scenarios, run by the load test scripts may consume a subset of this data.
- Ensure that validation of transactions is implemented correctly. Many transactions are reported successful by the Web server, but they fail to complete correctly. Examples of validation are, database entries inserted with correct number of rows, product information being returned, correct content returned in html data to the clients etc.

2.1.6 Execute the Test. Run and monitor your tests

Validate the tests, test data, and results collection. Execute validated tests for analysis while monitoring the test and the test environment.

Test execution can be viewed as a combination of the following sub-tasks:

a. Coordinate test execution and monitoring with the team.
b. Validate tests, configurations, and the state of the environments and data.
c. Begin test execution.
d. While the test is running, monitor and validate scripts, systems, and data.
e. Upon test completion, quickly review the results for obvious indications that the test was flawed.
f. Archive the tests, test data, results, and other information necessary to repeat the test later if needed.
g. Log start and end times, the name of the result data, and so on. This will allow you to identify your data sequentially after your test is done.

2.1.7 Analyze Results, Report, and Retest

Consolidate and share results data. Analyze the data both individually and as a cross-functional team. Reprioritize the remaining tests and re-execute them as needed. When all of the metric values are within accepted limits, none of the set thresholds have been violated, and all of the desired information has been collected, you have finished testing that particular scenario on that particular configuration. Most reports fall into one of the following two categories:

a. **Technical Reports**
   - Description of the test, including workload model and test environment.
   - Easily digestible data with minimal pre-processing.
   - Access to the complete data set and test conditions.
   - Short statements of observations, concerns, questions, and requests for collaboration.

b. **Stakeholder Reports**
   - Criteria to which the results relate.
   - Intuitive, visual representations of the most relevant data.
   - Brief verbal summaries of the chart or graph in terms of criteria.
   - Intuitive, visual representations of the workload model and test environment.
   - Access to associated technical reports, complete data sets, and test conditions.
   - Summaries of observations, concerns, and recommendations.

2.2 Types of Performance Testing:

The following are the most common types of performance testing for Web applications.

a) **Performance testing**. This type of testing determines or validates the speed, scalability, and/or stability characteristics of the system or application under test. Performance is concerned with achieving response times, throughput, and resource-utilization levels that meet the performance objectives for the project or product. In this guide, performance testing represents the superset of all of the other subcategories of performance-related testing.

b) **Load testing.** This subcategory of performance testing is focused on determining or validating performance characteristics of the system or application under test when subjected to workloads and load volumes anticipated during production operations.

c) **Stress testing.** : This subcategory of performance testing is focused on determining or validating performance characteristics of the system or application under test when subjected to conditions beyond those anticipated during production operations. Stress tests may also include tests focused on determining or validating performance characteristics of the system or application under test when subjected to other stressful conditions, such as limited memory, insufficient disk space, or server failure. These tests are designed to determine under what conditions an application will fail, how it will fail, and what indicators can be monitored to warn of an impending failure.

d) **Capacity Test**: Main purpose is to determine how many users and/or transactions a given system will support and still meet performance goals. Capacity testing is conducted in conjunction with capacity planning, which

you use to plan for future growth, such as an increased user base or increased volume of data. Capacity testing helps you to identify a scaling strategy in order to determine whether you should scale up or scale out.

3. Web Performance Testing Tools

3.1 Apache JMeter: Open source load testing tool: It is a Java platform application. It is mainly considered as a performance testing tool and it can also be integrated with the test plan. In addition to the load test plan, you can also create a functional test plan. This tool has the capacity to be loaded into a server or network so as to check on its performance and analyze its working under different conditions. Initially, it was introduced for testing the web applications, but later its scope had widened. It is of great use in testing the functional performance of the resources such as Servlets, Perl Scripts and JAVA objects. Need JVM 1.4 or higher to run [Link: http://jmeter.apache.org/download_jmeter.cgi]. It works under UNIX and Windows OS.

3.2 NeoLoad : Load and performance testing software: This is a tool used for measuring and analyzing the performance of the website. The performance and the end result can be evaluated by using this tool and any further steps can be taken. This helps you in improving and optimizing the performance of your web application. This tool analysis the performance of the web application by increasing the traffic to the website and the performance under heavy load can be determined [Link: http://www.neotys.com/documents/download/neoload/v3.2/loadGenerator_windows_3_2_7.exe]. This tool is compatible on operating systems like Microsoft windows, Linux and Solaris

3.3 Load Runner: This is a HP product which can be used as a performance testing tool. This can be bought as a HP product from its HP software division. Also, it is very much useful in understanding and determining the performance and outcome of the system when there is actual load. One of the key attractive features of this testing tool is that, it can create and handle thousands of users at the same time. The LoadRunner comprises of different tools; namely, Virtual User Generator, Controller, Load Generator and Analysis.[5] [Link http://www8.hp.com/us/en/software/software-product.html?compURI=tcm:245-935779&pageTitle=loadrunner-software%C2%A0].

This tool provides good result in Microsoft Windows and Linux are the favorable OS for this measuring tool.

3.4 LoadUI: Open Source Stress Testing Tool: Load UI is yet another open source and load testing software used for measuring the performance of the web applications. This tool works effectively when it is integrated with the functional testing tool soapUI. LoadUI is the most flexible and interactive testing tools. This allows you to create, configure and update your tests while the application is being tested. It also gives a visual Aid for the user with a drag and drop experience. This is not a static performance tool. The advanced analysis and report generating features allows you to examine the actual performance by pumping in new data even while the application is being tested [Link: http://sourceforge.net/projects/loadui/files/]. This tool being a open source application, it is available for free and everyone can have the easy access to its full source code.

3.5 WAPT (Web Application Testing): Performance testing tool for web sites and intranet applications: WAPT refers to the Web Application Performance tool. These are scales or analyzing tools for measuring the performance and output of any web application or web related interfaces. These tools help us to measure the performance of any web services, web applications or for any other web interfaces. With this tool we have the advantage of testing the web application performances in various different environment and different load conditions. WAPT provides detailed information about the virtual users and its output to its users during the load testing. This is considered to be the best cost effective tool for analyzing the performance of the web services. The WAPT tools can tests the web application on its compatibility with browser and operating system. It is also used for testing the compatibility with the windows application in certain cases [Link: http://www.loadtestingtool.com/wapt.exe]. WAPT System Requirement: Windows OS is required for this testing tool.

3.6 Rational Performance Tester: The Rational performance tester is an automated performance testing tool which can be used for a web application or a server based application where there is a process of input and output is involved. This tool creates a demo of the original transaction process between the user and the web service. By the end of it all the statistical information are gathered and they are analyzed to increase the efficiency. Any leakage in the website or the server can be identified and rectified immediately with the help of this tool. This tool can be the best option in building a effective and error free cloud computing service. This Rational Performance tester

was developed by IBM (Rational software division). They have come up with many versions of this automated testing tool.[5] [Link: http://www.ibm.com/developerworks/downloads/r/rpt/] . Rational Performance Tester System Requirement: Microsoft Windows and Linux AIX good enough for this performance testing tool.[5]

3.7 Testing Anywhere: Test anywhere is a automated testing tool which can be employed for testing the performance of any web sites, web applications or any other objects. Many developers and testers make use if this tool to find out any bottlenecks in their web application and rectify them accordingly. It is a powerful tool which can test any application automatically. This testing tool comes along with a built in editor which allows the users to edit the testing criteria according to their needs. The testing anywhere tool involves 5 simple steps to create a test. [5] [Link: http://www.automationanywhere.com/Testing/Downloads/Testing-Anywhere-Setup750.exe]. This tool is compatible with all versions of Windows OS.

3.8 QEngine (ManageEngine): QEngine (ManageEngine) is a most common and easy-to-use automated testing tool helping in performance testing and load testing of web applications. Many developers find it to be the most simple and easy tool to use for finding out any leakage in their web services or websites. The key important feature of this testing tool is its ability to perform remote testing of web services from any geographical location. This automated testing tool has the capacity to generate and simulate lot if users so that the performance can be well analyzed during the maximum load. This is free software available for the users online [Link: http://www.manageengine.com/products/qengine/]. This tool works with the Microsoft Windows and Linux.

3.9 Httperf: Httperf is a high performance testing tool for measuring and analyzing the performance of any web services and web applications. This is mainly used to the test the HTTP servers and its performance. The main objective of this testing tool would be to count the number of responses generated from this particular server. This generates the HTTP GET requests from the server which helps in summarizing the overall performance of the server. Through this tool, you will be able to conclude the rate at which the response is sent from each server and thereby the efficiency can be calculated. This is a Hewlett Packard product. [5] [Link: http://www.hpl.hp.com/research/linux/httperf/download.php]. Httperf System Requirement: Windows and Linux.

3.9 Selenium is a portable software testing framework for web applications. Selenium provides a record/playback tool for authoring tests without learning a test scripting language (Selenium IDE). It also provides a test domain-specific language (Selenese) [1] to write tests in a number of popular programming languages, including Java, C#, Groovy, Perl, PHP, Python and Ruby. The tests can then be run against most modern web browsers. Selenium deploys on Windows, Linux, and Macintosh platforms. [Link : http://www.seleniumhq.org[5]

4. The Major Challenges in Testing Web based Application

Figure 2: Problems in Web Application Testing

The performance testing is today an unavoidable part of quality assurance programmes. Organizations understand the adverse impact that poorly performing IT systems can have on their revenues and reputation, and want to guard against such a scenario. However Quality Assurance (QA) teams do face challenges when it comes to performance testing. Performance testing is indispensable for managing certain significant business challenges. For example, if your Web site cannot handle the volume of traffic it receives, your customers will shop somewhere else. Beyond identifying the obvious risks, performance testing can be a useful way of detecting many other potential problems. Many businesses and performance testers find it valuable to think of the risks that performance testing can address in terms of three categories: speed, scalability, and stability.

The primary purpose of performance testing is to ensure optimization of databases, response to data inputs, and CPU and memory utilization of servers. Further certain characteristics of applications make them more conducive to this alternative than others:

a. Number of users accessing the application simultaneously is not too high (~300)
b. Overlap between functional and load test scenarios is high
c. Applications developed are based on object oriented concept which allows functional test tools to recognize screen controls

Some of the typical challenges faced by QA teams during performance testing are

a. Procurement of expensive tool licenses
b. Cost and technical issues faced while upgrading to newer versions of performance testing tools
c. Unavailability of compatible performance testing tool versions
d. Stringent project timelines

To test the applications in various browsers as the end user prospective and also the user can use the Web application using various or a wide range of connection speeds. To do this we have various tools like selenium etc. in the market. In

web-based applications the business environment plays a key role in the testing, especially in the case of e-commerce web applications. In those applications the issues like tax calculation, shipping costs determination, completing and executing financial transactions, tracking of customer profiles.

Another major challenge is testing under the various Testing Environments. Web-based application testing is very expensive and time consuming because we require the same or a duplicate production environment which are web servers, application servers, and database servers that are basically required to ensure the quality and test the web applications.

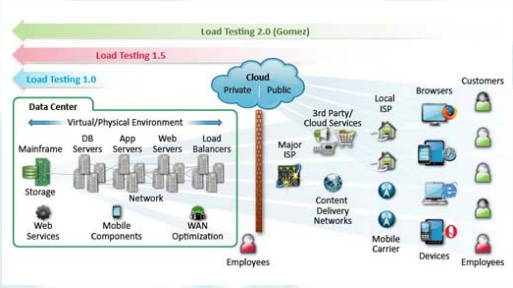

Figure 3: Various levels of web accessing

In the web application security is one of the major challenges that an application faces because the web-based application is running on the Internet or Intranet and it is open to the entire world. Therefore protection from unauthorized access to the application is very crucial and it will help the application to be protected from hackers. Web components in an application are developed by various companies and are integrated to the applications based on business need. The integration of these components can result in the malfunctioning in an application, therefore integration testing of these components are very important and the very biggest challenges for the web testers. Another challenge to testing web applications is compatibility with various browsers because today many browsers are being used in the market and each has its own function and behavior. Sometimes it is possible for an application to change its look in various browsers. So to ensure the application can work or look as the end user requires, testers need to do compatibility testing. The Ranorex Test Suit Runner is a popular tool also used in various browsers for this type of testing.

4.1 Factors effecting Performance Testing of Web Applications

- Numerous Application Usage (Entry – Exit) Paths are possible
- People with varying backgrounds & technical skills may use the application
- Intranet versus Internet based Applications
- The end users may use different types of browsers to access the app
- Even on similar browsers application may be rendered differently based on the Screen
- Resolution/Hardware/Software Configuration
- Network speeds
- ADA (Americans with Disabilities Act)
- Other Regulatory Compliance/Standards
- Firewalls
- Speed related challenge [8]

If applied functionality is multithreaded and synchronized methods, testing the performance with the other modules by using the tool is a challenge. Performance testing is a complex and time consuming activity. The process should start from the requirements collection phase itself. Performance Testing requires simulating several hundred concurrent users. This requires automated tools that are expensive. A proper environment like bandwidth, system configuration and concurrent users hinders in providing adequate performance results. Production environment cannot be simulated as it requires investments. However, testing the system for an optimal performance is a challenge and an opportunity [7].

4.2 Future Scope

Above mentioned tools can be studied independently and comparative study repot can be generated with statistical graph, which gives clear idea about the tool performance.

5. Conclusions

The World Wide Web is an irresistible entity when it comes to business and information access in the modern world. All business systems utilize WWW to perform effectively in their own domains of business. A world wide survey shows that business systems are unacceptable if the system is not high performance conscious. Ensuring high performance is the main criteria for Web users to repeatedly use the same site for their business. The performance of any system is attributed to many parameters like response time to a user query, high throughput from the system and availability at all times. To ensure this each system must undergo performance testing before its development. Performance Testing could be conducted in many ways such as load, endurance, stress and spike testing. All applications must follow their own lifecycles like functional testing. If the Performance Testing life cycle is followed systematically, one can ensure the performance of the web site that satisfies the customer.

References

[1] Qian, Z., H. Miao and H. Zeng, 2007. A practical web testing model for web application testing. Proceeding of the Third International IEEE Conference on Signal-Image Technologies and Internet-Based Systems. Dec. 16-18, Shanghai, pp: 434-44. 10.1109/SITIS.2007.16

[2] Elbaum, S., Karre, S., Rothermel,G., Improving web application testing with user session data. In International conference of software Engineering, pages 49-59, 2003

[3] Performance Testing Guidance for Web Applications , patterns & practices By J.D. Meier ,Carlos Farre,

Prashant Bansode, Scott Barber, Dennis Rea, Microsoft Developer Network page 16 2007
[4] http://en.wikipedia.org/wiki/Software_performance_testing
[5] http://www.softwaretestinghelp.com/performance-testing-tools-load-testing-tools/
[6] Performance Testing With JMeter 2.9 By Bayo Erinle
[7] Integrated Approach to Web Performance Testing edited by B. M. Subraya
[8] The Art of Application Performance Testing: Help for Programmers and Quality, By Ian Molyneaux
[9] Web-Based Systems and Performance Testing by Subraya
[10] eValid, a web testing tool: http://www.soft.com/eValid/evindex.html

Author Profile

Mr. Vinayak Hegde received the M.Sc Degree in Software and Information Systems from Bharthiar University Coimbatore in 2004. Currently he is working as an Assistant professor, Computer Science Department, Amrita Vishwa Vidyapeetham Mysore Campus, Karnataka, India. His Interested areas are Web Testing, Semantic Web

Ms. Pallavi M S received the MS Degree in Computer Science from University of Mysore in 2007. Currently she is working as a Lecturer in Computer Science Department, Amrita Vishwa Vidyapeetham Mysore Campus. , Karnataka India Her Interested areas are Web Testing and Web Analytics.