Web Application Performance Analysis Based on Component Load Testing

Charu Babbar¹, Neha Bajpai² and Dipti Kapoor Sarmah³
¹Center for Development of AdvanceComputing ,Noida
²Center for Development of AdvanceComputing ,Noida
³Maharashtra Academy of Engineering,Pune
¹babbar.charu@gmail.com, ²nehakapoor@cdacnoida.in and ³dkapoor@maepune.com

ABSTRACT
Performance of many web sites depends on the load on the site at peak time under varying conditions. Performance testing is normally conducted in a reasonably simulated environment with the help of performance testing tools. However, performance of a website depends on various parameters and each parameter must be tested under varying stress levels. It is not possible to draw a common denominator for performance parameters to test the websites due to complexity of websites. Different parts of the websites must be tested with different parameters under varying condition and stress level. In such circumstances, it is necessary to decompose the websites into many components, which represents the behavior of various business components. These business components are mapped to various objects that truly represent the behavior and structure of the part of the website. These objects are subjected to performance testing with different parameters and stress levels. This paper addresses the new testing process, which uses the concept of decomposing the behavior of the web Application into testable objects. These testable objects are subjected to performance testing under varied performance parameters and stress levels.

1.1 SETTING PERFORMANCE GOALS
Performance goals will differ depending on the application technology and purpose of application however they should always include some of the following:

- **Concurrency/Throughput**
  If an application identifies end-users by some form of login procedure then a concurrency goal is highly desirable. By definition, this is the largest number of concurrent application users that the application expected to support at any given moment. The workflow of the scripted transaction may impact true application concurrency especially if the iterative part contains the Login/Logout activity. If the application has no concept of end-users then the performance goal is likely to be based on a maximum throughput or transaction.

- **Server Response Time**
  This refers to the time taken for one application node to the request of another. A simple example is HTTP ‘GET’ request from browser client to web server in terms of response time which is measured by load testing tools. It may be relevant to set server response time goals between all nodes of the application landscape.

- **Render Response Time**
  A difficult thing for load testing tools to deals with as they generally have no concept of what happens within a node apart from recognizing a period of time where there is no activity ‘on the wire’. To measure render response time it is generally necessary to include functional test scripts as part of the performance test scenario which is a feature not offered by many load testing tool. [2]

1.2 PREREQUISITES FOR PERFORMANCE TESTING
A stable build application which must resemble the production environment as close to possible. The performance testing environment should not be clubbed with user acceptance testing environment (UAT). This is dangerous as if an UAT or integration testing or other testing is going on the same environment, then the results obtained from the performance testing may not be reliable. So, it is always advisable to have a separate performance testing environment resembling the production environment as much as possible.

1.3 MYTHS OF PERFORMANCE TESTING
Some of the very common myths of Performance Testing are given below:

- Performance Testing [1] is done to break the system.
- Performance Testing should only be done to break the system.
- Performance Testing only involves creation of scripts and any application changes would cause simple refactoring scripts.

2. WHAT IS WEB TESTING?
Conventional application software testing is well defined with a set of methods and established over the years. Each method is implemented with a proper strategy. However, one can use some heuristics to test the specific parts, which cannot be done by the available methods due to many practical aspects of the software. But, the conventional methods may not walk with the web based applications. These applications vary with functionality, presentation, and target users. Thus web applications are dynamic in nature and require strong testing methodology. Web testing mainly involves testing functionality of the system like in conventional system, presentation layer and performance. During testing of the web applications, many issues must be addressed as the website is subjected to many unknown and uncertainties.

3. WHAT IS PERFORMANCE TESTING?
Modern computer systems are becoming more complex and dependent on many factors such as the network technologies on the internet. Computing is distributed between various processors such as in the client server paradigm or clients and web servers. Deploying applications that rely on webservers, intranets and client server technologies is a challenge both in assuring that the functionality will be maintained and in guaranteeing that the functionality will be delivered with an acceptable performance. Performance problems can bring all sorts of undesired consequences, including financial and sales loss, decreased productivity. Performance prediction may be accomplished through performance models.

Performance testing is directly reflecting the behavior of the complete website. Visitors expect the fast response within a short period of time. Therefore, a rigorous performance testing must be carried out on each site. Performance testing can be viewed as a “Black Box” testing, which focuses on application and system behavior from outside, with no knowledge of the program code that supports the system. The main objectives of the performance testing is:

- Maximum number of concurrent users that can be supported while offering “acceptable performance”.
- Maximum number of concurrent users that can be supported prior to causing the system failure.
- Location of bottlenecks within the application architecture.
- Impact of software or hardware change on the overall performance of the application.
- Scalability issues.

The performance testing is done by recording or scripting the actions that real users perform, and then playing those actions back against the System Under Test (SUT), in an automated and controlled manner. Performance testing also depends on certain issues like graphics load time and connection methods.

4. ASPECTS OF PERFORMANCE TESTING
Several aspects that can affect the performance includes the following:
- High activity and volume at launch.
- Time of the day.
- Activity spikes due to marketing promotions.
- Bottlenecks due to hundreds of users on a network.
- Download time.
- Usage pattern.
- Think Time.
- User arrival rates.
- Client platforms.
- Internet access speed.

5. TYPES OF PERFORMANCE TESTING
The size of each page and the number of images used on the websites can also affect the performance of the application. It can be divided into three parts:

(1) Scalability testing
(2) Load testing
(3) Stress testing

Scalability testing: Scalability concerns the website’s ability to handle the volumes and types of activities that can occur after launch.

Load testing: The purpose of Load testing is to model real world experiences, typically by generating many simultaneous users accessing the website. Automation increases the ability to conduct a valid load test, because we can emulate thousands of users by sending simultaneous requests to the application or the server. In order to create adequate load and scripts, the tester uses information from daily usage logs to mimic a realistic user load.

Preparing for Load test: Load testing must be carefully planned to successfully complete the load test the first time, thus shortening the amount of test time. In reality, even with the best planning, load testing may need to be repeated at least once, and usage more time than initially thought due to the time necessary to setup the load test.

Some good points for preparing the load test are:

1. Understand the load requirements of the system.
   Investigate the total and concurrent number of users the website may need to support. Newly released applications will not have history statistics available. The test team may have to rely on the requirement specification to collect specific targets, such as:
   i. Number of users in either unique hits per day, per week, or per month;
ii. Total concurrent users: worst-case scenario at peak time;
iii. Peak request rate: number of pages to be served per second.

(2) Identify the tools, such as test tools and monitoring tools, to be used to conduct the load test.

(3) Generate enough users and transactions to access capacity and performance that will hold up to a live environment.

(4) Establish a baseline: create scripts to simulate a single user with single browser.

(5) Create test scripts to simulate multiple session played on multiple browsers.

(6) Identify any other applications that are running on the application server to capture the correct system activity.

(7) Execute the test(s) multiple times.

(8) Identify the players who will be monitoring the system performance during the test.

(9) Conduct formal inspections on the test scripts.

5.1 PARAMETERS FOR PERFORMANCE TESTING:
1) Response time during data retrieval.
2) Calculations.
3) Turnaround time.
4) Bandwidth.
5) Webpage load time during multiple user logins.
6) Capacity.
7) Volume.
8) Load

5.2 TIME PARAMETERS:
1) Elapsed time
2) Response time
3) Hits per second
4) Throughput
5) Page download time
6) Transaction per second
7) Request
8) Response

All these issues result in the need for an efficient web-based load testing tool for testing web applications with individual components performance.

7. IMPLEMENTATION

7.1 OUTPUT 1
7.2 OUTPUT 2

Connecting to the web page.
Connected to web page.
Retrieving contents from perfect.html
sun.net www.protocol like File://localhost/filenamexxx/perfect.html
Request time for images is: 1220381.00 milliseconds.
Number of Occurrence of images = 1
Response time for images is: 1220381.00 milliseconds.
Load time for images is: 15 milliseconds.
Request time for html files is: 1220381.00 milliseconds.
Number of Occurrence of html pages = 1.
Response time for html files is: 1220381.00 milliseconds.
Load time for html files is: 15 milliseconds.
Request time for css sheets is: 1220381.00 milliseconds.
Number of Occurrence of css sheets = 0.
Response time for css sheets is: 1220381.00 milliseconds.
Load time for css sheets is: 15 milliseconds.
BUILD SUCCESSFUL, total time: 0.000 seconds.

7.3 OUTPUT 3

Connecting to the web page.
Connected to web page.
Retrieving contents from perfect.html
sun.net www.protocol like File://localhost/filenamexxx/perfect.html
Request time for images is: 1220381.00 milliseconds.
Number of Occurrence of images = 1
Response time for images is: 1220381.00 milliseconds.
Load time for images is: 15 milliseconds.
Request time for html files is: 1220381.00 milliseconds.
Number of Occurrence of html pages = 1.
Response time for html files is: 1220381.00 milliseconds.
Load time for html files is: 15 milliseconds.
Request time for css sheets is: 1220381.00 milliseconds.
Number of Occurrence of css sheets = 0.
Response time for css sheets is: 1220381.00 milliseconds.
Load time for css sheets is: 15 milliseconds.
BUILD SUCCESSFUL, total time: 0.000 seconds.

8. BENEFITS OF PROPOSED SOLUTION

1. Measures can be obtained for the users’ effectiveness, efficiency and satisfaction.
2. Reports generated are useful for checking particular component performance.

CONCLUSION
Web based applications are more complex compared to the conventional client-server applications due to many unknowns and uncertainties. Among many complexities, performance testing is one of the difficult activities which has to be tackled with more vigor and aggression.
Performance testing, which enhances the customer confidence on the web applications, is based on many approaches and
strategies. Testing the site as a whole is cumbersome and tedious due to many complexities including the behavior. The sub-behaviors like, a query, a downloadable objects and meaningful business entities is mapped on the web application by properly organizing the application. The behavior of each component, which is available in all performance testing tools. The scenario is then for a specific test session for different stress levels. This methodical approach not only ensures the structured way of testing but also provides step by step enhancement of the quality of the site. Experimental results shows that the whole web application need not be subjected to rigorous performance testing with all performance indicators set for each subbehavior. The proposed idea will be helpful as it will provide the complete loading information about every component used in the application like request time, response time, load time, etc.

ACKNOWLEDGEMENT
I sincerely thank to Mrs. Neha Bajpai(Project guide) and to my friend who helped me in conducting the approach and calculating the results during performance analysis.

REFERENCES