AgileLoad

Application Performance Testing Basics

ABSTRACT

Todays the web is playing a critical role in all the business domains such as entertainment, finance, healthcare etc. It is much important to ensure hassle-free and responsive operations of front-end as well as back-end operations. This paper addresses performance testing concepts and importance of performance testing from business perspectives. Performances testing goals, types of performance testing, performance testing process, performance bottlenecks and challenges with performance testing are discussed to familiarize the readers with performance testing, its importance and to remove all the misunderstandings and myths associated with performance testing.

INTRODUCTION

Performance testing is a non-functional type of testing to determine the system responsiveness i.e. speed, stability, reliability and scalability.

Although performance testing is non-functional type of testing, but in this rapidly growing fast technological era, performance has become extremely important and organizations are keenly focusing on it, to gain competitive edge by satisfying their clients in a quick, fast and efficient manner. Today organizations preferred to conduct complete performance testing of their applications before going live, to make sure their application is fast enough in responding the intended user's requests and can handle the expected and unexpected user loads. Performance testing increase client's confident level and ensures the Application Under Test (AUT) behavior as expected under extreme conditions.

> The peace of mind that it will work on go live-day alone justifies the cost of performance testing

Performance testing of business critical applications before its launch is almost as important as its functional testing. Studies reveals user conversation rate of an application start to suffer when its response time exceeds from 5 seconds. Unsatisfied users may never come back again and can also influence others from not trusting the application and can make huge business impacts. Few businesses ignore the performance testing activity due to its associative cost but it can lead to business disaster at the end. The average cost of performance testing is nearly 2.5% of the total development cost. But fixing a poorly performing application when it's in production environment can cost nearly 25% of the development cost.

GOALS OF PERFORMANCE TESTING

There are various factors which are examined but following are some of the main goals of conducting the performance tests on an application.

- Access the Production Readiness
    - Check the system response time during expected load conditions
    - System behavior during unexpected load conditions
    - Check the system scalability
    - Best configuration settings for optimal performance
    - System behavior during spike user loads
    - System stability
- Compare two platforms with the same software to see which performs better
- Compare Performance characteristics of system configurations
- Evaluate System against performance criteria

- Find throughput level
- Discover what parts of the application perform poorly and under what conditions
- Finding the source of performance problems
- Support system tuning

IMPORTANCE OF PERFORMANCE TESTING

Importance of Performance testing can only be realized from the real world examples of business losses due to ignoring the performance testing activity before officially launching an application. Following are some of these performance stats and business losses examples.

- Average user clicks away after 8 seconds of delay
- $45 billion business revenue loss due to poor web applications performance
- In November 2009, a computerized system used by US based airlines to maintain flight plans failed for several hours causing havoc amongst all major airports. This caused huge delays to flight schedules causing inconvenience for thousands of frustrated passengers. Identified as a 'serious efficiency problem' by the Federal Aviation Authority, this was one of the biggest system failures in US Aviation History!
- Aberdeen found that inadequate performance could impact revenue by up to 9%
- Business performance begins to suffer at 5.1 seconds of delay in response times of web applications and 3.9 for critical applications and an additional second of waiting on a website significantly impact customer satisfaction and visitor conversions. Page views, conversions rate and customer satisfaction drops 11%, 7% and 16% respectively!

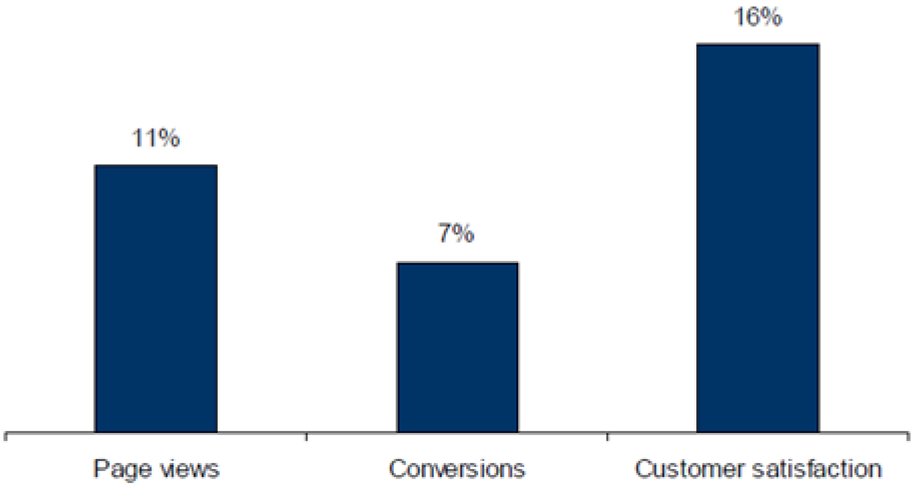

Fig: 1

Based on these case studies one can clearly find the importance of performance testing and risks associated with neglecting the proper performance testing activity.

TYPES OF PERFORMANCE TESTING

“Performance Testing” is a generic term which is used to describe a category or suite of tests. Performance testing can be of many types including:

- Load test
- Stress test
- Soak/Endurance Testing
- Volume Testing
- Scalability test

Load Testing

Load testing is performed to test the application behavior at various load levels within its acceptable limits. Main parameter to focus during Load testing is “Response Time”.

Questions to answer through load testing:

- How well the application behaves under normal load conditions?

Stress Testing

Stress testing is performed to find out the break point where application performance deteriorates. Main parameters to focus during Stress testing are “Response Time” and “Throughput”.

Questions to answer through stress testing:

- What happens if the production load exceeds the anticipated load?
- What kinds of failures should we plan for?
- What indicators should we look for?

Soak/Endurance Testing

Soak testing is performed to determine if the application can handle continuous load for an extended period of time. Main parameter to focus during Soak testing is “Memory”.

Questions to answer through Soak Testing:

- Is there any memory leaks?
- Will performance be consistent over time?
- Are there slowly growing problems that have not yet been detected?

Volume Testing

Volume testing is done to determine the application performance against sudden user increase. Main parameter to focus during Volume testing is “Response Time”.

Questions to answer through Volume Testing:

- Whether the system will be able to handle dramatic change in load?
- What will be application behavior at various DB levels?
- What happens if the production load exceeds the anticipated peak load?

SCALABILITY TESTING

Scalability testing is performed for measuring the system capability to scale up. Main parameters to focus during Scalability testing are “Response Time” and “Transactions per Seconds (TPS)”.

Questions to answer through Scalability testing:

- What is the system MAX TPS?
- Is system capacity meeting business volume under both normal and peak load conditions?

PRE-REQUISITES OF PERFORMANCE TESTING

There are certain pre-conditions which should meet before starting the performance testing activity. Executing the performance testing activity without fulfilling all these conditions can produce really useless results at the end. Following are some of these pre-requisites.

- Stable and defect-free dedicated environment
- Performance testing environment similar to production environment
- No other testing should be performed while Performance Testing
- Performance testing before going live
- Performance testing plan development
- Test Data Preparation
- Performance testing requirements gathering
    - Application architecture
    - Application development technologies
    - Servers information
    - Application usage information
    - Application usage patterns
    - Performance Acceptance criteria

PERFORMANCE MANAGEMENT

There are two approaches of managing the performance testing activities, either after the development of the complete application after the system testing (Reactive Approach) or doing the performance testing throughout the Software Development Life Cycle (Proactive Approach). Both approaches have their advantages and disadvantages listed below.

Reactive Approach

Performance Testing is most often approached in a reactive way where performance testing is only done after the System testing. Following are the advantages and disadvantages of Reactive performance testing.

Advantage

- Cost effective

Disadvantages

- Difficult to resolve the performance bottlenecks after the complete development
- Defect removal cost is exponentially increase
- Whole system can be useless

Proactive Approach

In proactive approach every performance parameter is analyzed and addressed in testing environment before it really impact the production system and fix it before launching the application. Following activities are performed in each phase of the Software Development Life Cycle (SDLC).

- **Non-Functional Requirements Gathering phase** – You should start thinking of performance goals you will set
- **Design phase** – Define system performance matrices and their explicit goals
- **Development phase** - Frequently perform performance testing on prototype or partially complete features to validate defined performance goals
- **Test Execution phase** – Perform detailed performance testing to verify system's performance goals
- **Maintenance** – Run a perform testing cycle after specific interval and also with every new release to validate the system performance

Proactive approach is considered the best among the two approaches and has following advantages and disadvantages over Reactive approach.

Advantages

- Lower performance bottleneck mitigating cost
- All performance bottlenecks are resolved before launching the application
- Complete peace of mind and confidence before launch

Disadvantages

- More cost associated with performance testing activities when they are exercised in each phase of the SDLC

PERFORMANCE TESTING CYCLE

Typical Performance testing cycle consists of following three activities,

1. Run the performance test to determine the application performance parameters
2. Analyze results to examine whether the application is fulfilling the performance goals or not
3. Optimize the application to resolve the performance bottlenecks

This three steps performance testing cycle is repeated unless the performance goals are achieved.

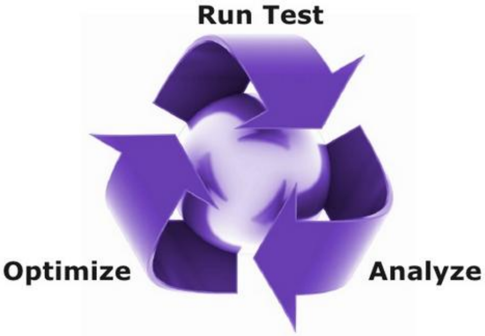

Fig 2

PERFORMACNE TESTING ACTIVITIES

Following activities are performed in performance testing process

1. Identify Test Environment
2. Identify Performance Acceptance Criteria
3. Plan and Design Test
4. Configure Test Environment
5. Implement Test Design
6. Execute Tests
7. Analyze Reports and Retests

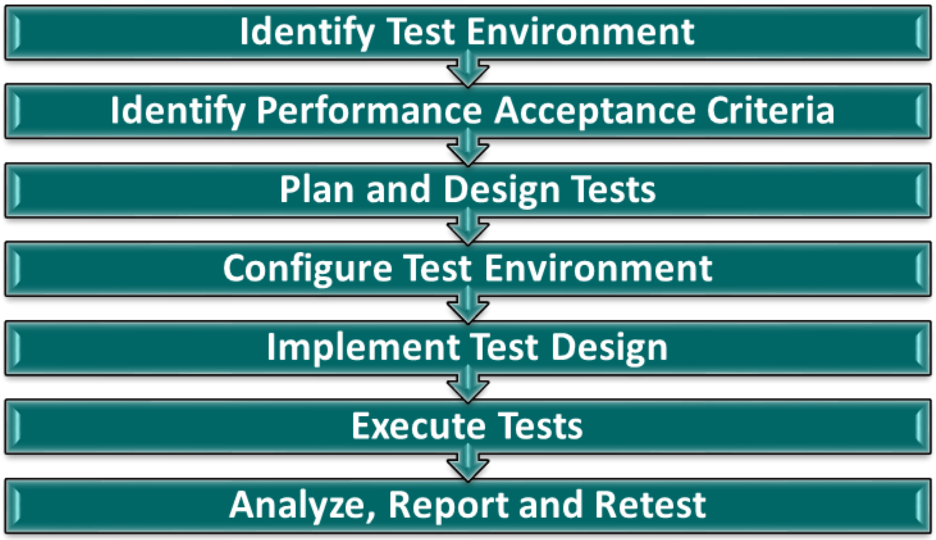
Identify Test Environment
Identify Performance Acceptance Criteria
Plan and Design Tests
Configure Test Environment
Implement Test Design
Execute Tests
Analyze, Report and Retest

Fig 3

Identify Test Environment

For successful performance test, Performance test environment should be exact replica of production environment. Building the accurate test environment very similar to production environment is the fundamental step for successful performance test. Following activities are performed to identify the test environment similar to production environment

- Identify the production environment, its components and their details
- Identify performance test environment
- Identify Hardware and Software resources required for the test

Identify Performance Acceptance Criteria

Performance tests are executed against some quantitative goals and test results are verified against these goals. Setting up the realistic goals is very crucial for a performance test. It can never be fruitful without setting some explicit quantitative goals. Performance testing matrices and acceptable value of each matrix are identified through following activities of this phase

- Identify response time, throughput, resource utilization goals
- **Response time**. For example, the product catalog must be displayed in less than 3 seconds.

- **Throughput**. For example, the system must support 100 transactions per second.
- **Resource utilization**. A frequently overlooked aspect is the amount of resources your application is consuming, in terms of processor, memory, disk input output (I/O), and network I/O.
- **Maximum user load**. This test objective determines how many users can run on a specific hardware configuration.

- **Business related metrics**. This objective is mapped to business volume at normal and peak values; for example, the number of orders or Help desk calls handled at a given time.

PLAN AND DESIGN TEST

Selection of right performance scenarios is another very critical task in a performance test. You can never get the true performance picture of your application unless you select the right set of performance scenarios. Performance test scenarios are identified based on their importance, frequency and performance impact. Following is the criteria of test scenarios selection for performance test

- Business critical scenarios
- Mostly accessed scenarios
- Performance intrusive scenarios

Apart from test scenarios selection, test data is also prepared in this phase of performance testing.

CONFIGURE TEST ENVIRONMENT

Once all the performance test requirements are identified then comes the implementation phase. In this phase test environment is setup as identified in first phase of the performance testing (i.e. Identify test environment).

IMPLEMENT THE TEST DESIGN

After configuring the test environment, scripting of identified performance scenarios is done with the help of a performance testing tool in this phase. Following are the list of activities performed in Test Design Implementation phase

- script the identified tests with a performance testing tool
- Simulate all the user actions
- Design test cases (user distribution)
- Service Level Agreement (SAL)

EXECUTE THE TEST

This phase involves test script execution against different user sets with different system resources configurations and test results are monitored to see the trends.

ANALYZE RESULTS, REPORT AND RETEST

In last phase when performance tests have been executed and their results have been collected, then following activities are performed

- Analyze the test results
- Documented them and share with all stakeholders
- Retest the application with new configurations until you received desired results

PERFORMANCE SYMPTOMS AND ISSUE

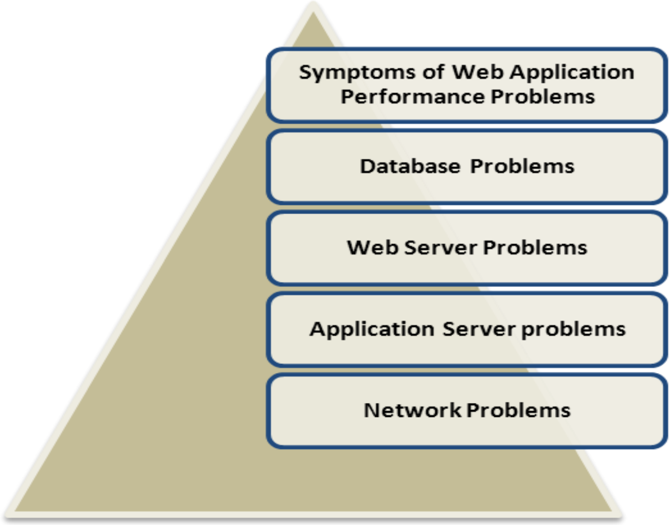
Symptoms of Web Application Performance Problems
Database Problems
Web Server Problems
Application Server problems
Network Problems

Fig 4

In an application performance bottlenecks could appear on various areas of the application infrastructure. Being a performance tester, one should have list of performance symptoms in his/her mind to properly analyze the test results and providing useful insights in mitigating performance bottlenecks. Following is the list of performance symptoms that could appear at various levels of an application.

SYMPTOMS OF WEB APPLICATION PERFORMANCE PROBLEMS

- Long user response time
- Long server response time
- Memory leaks
- High CPU usage
- Too many open connections
- Length queues for requests
- Too many table scans of database
- Database deadlocks
- Erroneous data returned
- HTTP errors
- Pages not available
- Page check errors

TYPICAL DATABASE PROBLEMS

- **Insufficient indexing:** Tune database indexing to improve query processing
- **Fragmented databases:** Place table records into adjacent pages
- **Out-of-date statistics:** Degrade query optimizer performance
- **Faulty application design:** Excessive DB calls, excessive data requests

TYPICAL WEB SERVER PROBLEMS

- **Poor server design:** Inefficient data or page caching
- **Memory problems:** Physical memory constraints

TYPICAL APP SERVER PROBLEMS

- **Poor database tuning :** Application server sending too many DB requests
- **Poor cache management :** Produces high CPU usage, disk access
- **Poor session management :** Produces high CPU usage, disk access, Time-outs
- **Poor security design :** Excessive use of https protocol
- **High CPU usage** : Usage >70% indicates problems

NETWORK PROBLEMS

Potential sources of network problems:

- Firewall throughput
- Internet access throughput
- Load balancers, gateways, routers

TYPICAL ORDER OF FIXES

Once the performance bottlenecks are identified, then there comes the stage of their fixing. Like Functional defect performance bottlenecks are also prioritized and fixed in a certain order. Following is the typical priority list of fixes,

- **Improve current application design:** Algorithms, caching, DB calls, memory use
- **Upgrade hardware:** RAM, CPU, network bandwidth
- **Upgrade software infrastructure:** OS, web server, database (database connection polling)
- **Upgrade system architecture:** Client-server to basic n-tier, basic n-tier to enterprise n-tier, software and hardware changes, Use Apache HTTPD in front of Tomcat to serve static resources, Use hardware load balancing /SSL

CHALLENGED WITH PERFORMANCE TESTING

Conducting a successful performance test is not a piece of cake. There are many challenges that you need to successfully face to conduct a successful performance test. Following is a list of these challenges,

- Test environment setup
- Collection and Analysis of huge data
- Bottleneck root cause identification
- Cooperative effort required (Product vendors, Architects, Developers, Testers, Database administrators, System administrators, Network administrators)
- Obtaining accurate results
- Client involvement
- Testing inside the firewall
- Performance testing of new technologies
- Testing on live environment
- Expensive

PERFORMANCE TESTING BEST PRACTICES

Although there are many challenges associated with successful performance testing activity but a performance tester can overcome some of these challenges by following some of the best practices of conducting the performance test. Here are examples of some of these,

- Use user Ramp up and Ramp down approach
- Ignore the results collected during Ramp up and Ramp down periods
- Run individual tests of performance scenarios before combining them in a single test
- Run a baseline test with single user to validate the script
- Run a benchmark test with 15 to 20 percent of desired load to validate the system matrices at lower load and check the system readiness for high load
- Run the test for at least 10-15 minutes at stable peak load
- Repeat tests at least 3 times to confirm the results
- Run tests at different times

CONCLUSION

Quality attributes and the performance of the application delivers successful product. Having adopted performance test process we believe that it enhances the business growth. In this paper we addressed performance testing concepts and highlighted the outcomes of each activity performed during the performance testing and its business impacts on neglecting the performance testing of an application before its goes to market.

Performance QA Team will get crystal clear understanding about performance testing, its process and symptoms that needs to be addressed for performance optimization and the business owners will get the idea of impacts of performance testing.

Sajid Manzor