Journal of Information Engineering and Applications
ISSN 2224-5758 (print) ISSN 2224-896X (online)
Vol 1, No.5, 2011

Performance Testing: Methodologies and Tools

H. Sarojadevi*

Department of Computer Science and Engineering, Nitte Meenakshi Institute of Technology,
PO box 6429, Yelahanka, Bengaluru -64

* E-mail of the corresponding author: hsarojadevi@gmail.com

Abstract

Performance testing is important for all types of applications and systems, especially for life critical applications in Healthcare, Medical, Biotech and Drug discovery systems, and also mission critical applications such as Automotives, Flight control, defense, etc. This paper presents performance testing concepts, methodologies and commonly used tools for a variety of existing and emerging applications. Scalable virtual distributed applications in the cloud pose more challenges for performance testing, for which solutions are rare, but available; one of the major providers is HP Loadrunner.

Keywords: Performance testing, Application performance, Cloud computing

1. Introduction

Building a successful product hinges on two fundamental ingredients — functionality and performance. 'Functionality' refers to what the application lets its users accomplish, including the transactions it enables and the information it renders accessible. 'Performance' refers to the system's ability to complete transactions and to furnish information rapidly and accurately despite high multi-user interaction or constrained hardware resources.

Application failure due to performance-related problems is preventable with pre-deployment performance testing. However, most teams struggle because of lack of professional performance testing methods, and guaranteeing problems with regard to availability, reliability and scalability, when deploying their application on to the "real world".

Performance testing is important for all types of applications and systems, especially for life critical applications in healthcare, medical, biotech and drug discovery systems, and mission critical situations such as automotives, flight, defense, and many others. In this paper a study of performance testing concepts and tools are presented, which are used for a variety of enterprise and scientific applications.

2. Performance/Load Testing Concepts

Performance Testing is a process of exercising an application by emulating actual users with a load-generating tool for the purpose of finding system bottlenecks. Often it is also termed Load testing. Main goal is testing for scalability, availability, and performance from the point of hardware as well as software. Resource aspects such as CPU usage, memory usage, cache coherence, data consistency (with regard to main memory, virtual memory pages, and disk), and power consumption, network bandwidth usage are also monitored and reported as part of performance testing. Further, response time, and usage related to router, web server, *appserver* (application server) are also considered in performance testing. Analysis for performance needs to be applied at each stage of the product development (Collofello 1988). Put together, system performance is perceived as a figure of merit from the point of response time, throughput, availability, reliability, security, scalability and extensibility.

2.1 *Why performance testing?*

In today's world of e-business, customers and business partners want their web sites and the web based

services to be competitive and many are moving to Cloud platform for the same reason. To sustain the competition, a website should satisfy the following criteria: pages need to download immediately; web pages must support efficient and accurate online transactions and near zero downtime. Any downtime could be very expensive. According to Gartner report, an average cost of unplanned downtime of a mission critical application is around $100,000 per hour.

Online consumer and B2B marketplaces are becoming more and more competitive. Companies must take care that their web-based applications accommodate multiple, simultaneous users who are connecting themselves to a web site or engaging in multiple online transactions. To ensure that such a service level is guaranteed, the service provider enterprises need to use application load-testing tool. According to a Jupitor media metrix consumer survey (**Error! Reference source not found.**), technical and performance problems of websites lead to abandoning of sites by more than 46% of the users.

The need for performance testing is more in life critical situations, with the systems used in heart surgery or angioplasty etc. There is a critical moment in an angioplasty procedure when the balloon is inflated inside the artery (Kelley 2006). During the next 60 seconds the balloon obstructs the artery that may cause another fatal heart attack. Other example of a critical application is genome based one, for instance a drug discovery system, the end product of which is going to act on the gene. Any malfunction in the process cannot be compromised as it would affect generations.

2.2 Key Features of a Load-testing Tool

A load-testing tool simulates the behavior of real users with "virtual" users. The load-testing tool can then record the behavior of the site under the load and give information on the virtual users' experiences. Load-testing software is often distributive in nature. It is deployed on multiple servers running simultaneously, with each server simulating multiple virtual users. In many cases, the testing tool vendor company develops its own proprietary browser that can be combined with a set of instructions tailored to the testing of each client business. Besides, ongoing records of the virtual users' experiences at the test site, including response times and errors, are also maintained for postproduction analysis.

Many testing companies also monitor the client web site remotely to help diagnose connectivity problems. The actual error messages experienced by the virtual users may be recorded for later review. A set of logs can be created that document each of the user experiences and this information can later be compared with the CPU and database testing information obtained during the test to diagnose the problem.

One feature that load-testing tools often provide is testing the Web-based application externally from multiple points of presence to find out if the quality of service provider's connectivity becomes the cause of system slowdowns. For example, if the client network was expected to have 10 Mbps of bandwidth but consistently experiences network slowdowns at 6 Mbps, this may signify that the network is not getting its expected bandwidth, possibly due to overloading, underutilization, or wrong usage.

A very useful feature of load/performance testing tool is that it provides information about the performance of the infrastructure of the client network itself. Firewalls, routers and load balancers may all be linked in a network. This may sometime create bottleneck, because a firewall may not have sufficient throughput to withstand the number of simultaneous users. A load-testing tool essentially simulates real user activity at the site and focuses more on performance of the application and the database under stress.

2.3 Factors for Successful Load Testing

Parameters complicating and thereby lengthening the task of performance testing are the following - complex tools, lack of experience, and lack of flexibility in the testing tools. Successful load testing requires that the Testers be trained well on tools, educated about the application domain, design and architecture to some extent, influence of technologies, and on performance testing methodologies.

Key steps involved in preparing for performance test are – 1. Prepare script, 2. Model and schedule workload, and 3. Execute Script. The support provided by a load-testing tool plays a major role in determining the cost-effectiveness, and the success of load-testing activity. The following are the

factors for successful load-testing.

- Testing at different speeds: It is the only way that allows us to see if slower connections use more resources. However this may reduce the number of virtual users who may simultaneously visit a web site.
- Testing on different browsers: Load testing on just one browser is not sufficient. For obtaining insight into the error-free performance of the web-based applications, it is required to load-test on different browsers.
- The ability to draw complex scenario to simulate user experiences: To simulate a real user experience, the company that load tests needs to create a scenario where the information to do the test are provided to the testing browsers. The scenarios need to be closely alike the transactions performed by the real users of the Web site.
- Good amount of scripting possibilities: Maximum scripts are needed to exercise full test scenario.
- Clear reporting: Reporting of errors, time response, throughput information, resource utilization, network monitoring, etc. that help optimize the system performance is necessary.
- User friendly and intuitive tools: The tools thus developed must be user friendly and intuitive to use such that the load testing effort becomes less costly, yet effective.
- Performance prediction: With an advanced capacity planning process, a system behavior can be modeled and workload characteristics are forecast. However, the real-world performance needs to be predicted with high scale up factor.

2.4 Application Performance

Typical performance objectives relate to response time and throughput under certain workload and configuration. Testing for performance early in the life cycle and validation at various stages, starting from the software requirements is a good practice. In fact performance analysis should be done during each of the verification and validation activities of a product (Myers 1969) in phases shown in Figure 1.

Performance testing is considered as non-functional testing. During the requirement specification stage performance objectives must be analyzed to ensure completeness, feasibility, and testability. Feasibility could be ensured by prototyping, simulation, or other modeling approaches. In the design phase, performance requirements must be applied to individual design components. Analysis of these components can help determine if the assigned requirements can be met. Simulation, prototyping, and modeling are the approaches applicable to this task. During the development and deployment, performance analysis must be done at every level of testing, with careful construction of test data suitable to the performance testing scenarios. Profile guided performance optimizations are used for tuning the performance in modern microprocessors.

Memory is a major resource that is critical to the system and the application as well. The method of reclaiming memory after its usage decides the overall performance. The technologies like COM/DCOM, J2EE or .NET have inbuilt garbage collector routines for reclaiming memory, which are based on popular algorithms such as simple mark-and-sweep, generation based or even advanced *train* algorithm (Hudson and Moss 1992). Since memory related problems such as leaks, overflow, byte order violation etc. cause major performance problems, often resulting in functional misbehavior as well as security breach, these must be monitored and reported in the form of diagnostic reports.

With the advent of distributed and concurrent systems, and the commercial multi-cores as well as Network-On-Chip kind of systems, memory related issues such as cache coherence and consistency need to be considered as vital performance parameters. These have to be addressed and enough measures need to be taken to combat any problems in this direction. Automated tracking of such problems using a hardware-software co-design will be a great advantage (Sarojadevi and Nandy 2011). Application performance is a cumulative result of all these aspects and tests need to be planned accordingly.

2.5 Benefits of performance testing

Performance testing brings in many advantages at various levels, be it business, project, process or product level. The following are a few benefits of performance testing.

- Better reliability: Performance testing helps avoid deadlocks, improve response time; helps provide scalability, fault tolerance, recovery from failures, etc.
- Shorter time to market: Performance testing reduces time to market greatly for large enterprise applications. In general if 98% of the high priority requirements are successfully tested it is then considered time to release to market. By treating the performance requirements that are treated non-functional, as part of high priority requirements, we can improve time-to-market, with considerable reduction in the test cycle, which results because of reduced defect rate.
- Helps put a tab on memory problems: Memory leak, overflow, data inconsistency, byte order violation -are a few major problems that can be monitored, measured and controlled.
- Secure software: Performance testing helps ensure secure software by detecting memory overflows, and other resource vulnerabilities, for web based as well as desktop-based application.
- Benchmarking: This can allow testing Quality of Service of various architectural options.
- Future expansions very easy: Performance testing helps accurately predict the required system and network capacity, which in turn helps in planning future expansions.
- Service level tests: Performance testing helps test various service levels to meet the challenges after deploying the product; thereby supports acceptance testing.

2.6 Sources of performance problems – in a nutshell

A tester needs to be aware of primary sources of performance problems for effective and efficient testing. The following are a few sources of performance problems that relate to various aspects of the system and its architecture.

- Technologies: J2EE is considered the most scalable and high performance architecture. However intense use of threading, use of rich environments, heavy transactions can cause performance overheads in Java/J2EE environments. The session affinity feature of J2EE provides ability to direct all the requests for a particular session to the same WebServer/J2EE container. Most commercially available J2EE containers support session affinity, via Apache or IIS plug-in, thereby ensuring server load balancing to guarantee high availability. Whereas, .NET, and COM/DCOM are constrained by heavy memory footprint, tightly coupled nature, heavy transaction entities between modules, and poor load balancing support.
- XML is used widely for interoperability support. Storage, retrieval and processing of XML add delay if used in the critical path. For navigation through XML structure, use of Document Object Model (DOM) incurs heavy memory footprint, whereas XPath navigator is lighter and faster for normal usage.
- Database: Having a Database server on web server can lead to severe performance problems besides adding security threats. Further, use of stored procedure (precompiled SQL statements) can reduce network traffic.
- Languages: Use of Java is supposed to be highly performance oriented. However Java uses synchronization statements in-between threads, locking resources in one place, which may potentially cause deadlock that leads to system breakdown. A break at any point in the system implies that the customers are not getting the service. The moment “Page not available” error appears, the customer moves on to some other page that can provide similar support. In fact all servlets contain several threads, a key source for memory leak and dead-locks.
- Network/interconnection: Network traffic and communication delay in the network are the most common performance problems. Network round trip time is usually long for a distributed application. Use of High Speed Ethernet (HSE) combining with high speed H1 field bus protocol at 31.25 KBits/sec provide complete solution. HSE can also connect to fiber optic media. Thus it is well suited for mission critical monitoring and process control applications, providing interoperability with any other connection technology, thereby having an edge over TCP/IP and normal Ethernet.

- Regarding protocols: Usage of network protocols need to be leveraged intelligently. For instance, SOAP protocol can be slower, as well as heavier than HTTP.
- Wireless protocols: Bluetooth wireless technology for personal area networking or to connect to ad-hoc networks operates over a short distance, with a high data transfer speed of 700Kbps. Bluetooth devices use radio transmission at 2.4GHz, which enable computers, mobile phones, printers, keyboards, mice, PDA, and other devices to communicate with each other without cables. A Bluetooth device can transmit through walls, pockets, and briefcases. Major performance problems like connection failure may happen, due to too many connections at a time (having ignored scalability issue while architecting), or interference with other standards such as wireless Wi-Fi 802.11 used for LAN, WAN, or the Internet.
- No batch processing: Using batch processing wherever possible, use of normalized data, disconnected data objects as DataSet in .NET, can significantly improve network performance, which is often ignored.
- Security features: Firewall, encryption or decryption of data in the database or outside, can cause performance problems such as increased access delay.
- Platforms/Servers: Servers such as DELL servers or HP integrity servers optimize for performance. These kinds of efficient servers must preferably be used always, as inefficient servers lead to poor performance.
- Algorithms: Consider imaging algorithms, especially the ones used in Medical imaging. – the techniques used for texture mapping, setting luminance values, packing and unpacking of pixel data values (OpenGL 2012), are performance critical as well as function critical. For instance, performing scaled normalization is necessary to get improved lighting condition that gives subtle clues to the viewer about the curvature and orientation of surfaces in your scene. However such operation includes complex arithmetic thereby slowing down the display. In striking a trade-off between performance and correct visualization effects, resource problems such as stack overflow[1], delay, byte order violation in memory store/retrieval etc. may occur, causing poor visualization (too bright/dull, unlit) of images leading to wrong diagnosis, fatality, data inconsistency routing wrong/stale data, and even leading to system crash.
- Internationalization (i18N): Handling resource bundles in local language (CJKV[2]) often requires more storage. Conversion from/to the local language may lead to memory overflow/underflow and byte order violations. Wrapping up dates, data sorting issues, Bi-directional (BiDi) support for Hebrew, Arabic, and Farsi, etc. need to be carefully designed to optimize on performance and scalability. Thus testing for performance problems in local environment is a must.

2.7 Ways to tackle

Strategic ways to combat the performance problems are as follows -

a. Building a test lab with the entire environment set up as with the actual deployment including gigabytes of data storage, and carrying out the tests - This is close to real scenario, still not a perfect practical one, since it is really hard to model or duplicate the geologically spread nature of the application such as ATM banking, SAP, and wireless mobile network applications.

b. Enhancing the infrastructure such as server upgrade, operating system upgrade from Windows 2000 to Windows 2003 – A case study of doing this in one of the applications increased server utilization by 36% for average 25-30 users; No delay has been observed.

c. Using automation tools for performance, load and stress testing – Use of standard performance/load test automation tools during the testing phase or earlier can significantly bring down the performance problems. Sometimes functional test

[1] Simulating hardware stack in software can let overflow go undetected since it then writes into memory locations without raising exception if not taken care.
[2] Chinese, Japanese, Korean & Vietnamese Computing

automation tools can also be enhanced to measure performance parameters such as network delay and response time.

3. Popular Automation tools for Load Testing

A number of open source and vendor specific tools are available for load and performance testing (Jay Philips 2010). A few providers of commonly used performance testing tools are given below.

The load testing software from Mercury Interactive (presently HP) is LoadRunner for predicting web server behavior and performance. The IBM Rational performance tester (RPT) is another major player. A performance testing tool, called SilkPerformer (Borland 2006) from Borland/Segue has many attractive features combined with ease of use, flexibility, and good reporting facility, and useful metrics. Many companies use IBM Tivoli performance monitoring tools.

QALoad is the Compuware performance testing tool, also providing performance monitoring services with QACenter performance edition.

A few other tools are – forecastweb from facilita; E-Load from Empirix for web applications; NeoLoad from Neotys, QuotiumPRO from Quotium -for load testing mid-range projects; ApacheBench from Apache, HttpPerf, OpenLoad from Sourceforge for small projects.

Of the newcomers to the testing tool market an impressive one is Facilita that provides forecast, a non-intrusive performance testing tool for system load testing, performance measurement and multi-user functional testing. Sun Microsystems has a performance monitoring and diagnostics tool, called Validation Test Suite (VTS) for monitoring performance of various hardware units such as cache, memory, processor pipeline, DISK, I/O, etc. for data consistency, correctness, and power consumption.

Parasoft Webking is an Automated Web application testing, which performs Web site risk analysis, functional testing, load and performance testing, and security analysis, thus ensuring that Web sites and Web applications meet their reliability, security, and performance goals.

Besides this, there are freeware such as Apache JMeter that can be used for web page performance testing such as SAP applications. It does not have wide feature set, nor can support multiple platform. OpenSTA is another open source tool for web performance, load and stress testing.

4. Cloud Performance Testing

Cloud is a technology that provides a scalable virtual distributed environment to the application. Cloud is deployed as services – including storage, middle layer and web deployments from which seamless flexibility, availability and load balancing can be extracted by the applications. The rise of cloud computing has brought the promise of infinite scalability for applications, but it has also brought a new set of challenges for developers and performance testers. With HP's LoadRunner in the Cloud (HP 2010 Cloud), businesses can test, tune, analyze and optimize applications for the cloud.

HP LoadRunner, the industry's best-selling load testing software is available in the form of Amazon Elastic Compute Cloud (*Amazon EC2*) for cloud applications, making performance testing accessible to businesses of all sizes. This on-demand software gives clients a flexible "pay as you go" approach for performance testing of mission-critical applications and websites.

HP also offers testing services delivered via Software as a Service (SaaS) to help IT organizations further reduce costs and improve business results. The following are the two possible flavors.

- HP Elastic Test enables IT organizations to take advantage of cloud elasticity to instantly expand testing capacity cost-effectively. Specifically designed for spike load testing, HP Elastic Test provides the ability to scale up to very large loads in a utility-based fashion.
- HP Cloud Assure takes advantage of the speed, flexibility, scalability and cost-effectiveness of cloud services. Based on 10 years of HP's SaaS expertise and advanced service-level performance, it delivers four of the following attributes, which are key to reliable cloud computing – security, performance, availability and cost control.

5. Challenges for Performance Test Automation

The following are the challenges for any automated performance/load testing.

- Traceability of Requirements into the performance testing tool
- Interface to test management tool
- Connection to defect management tool
- Support for internationalization
- Testing for availability, reliability, and recovery, load balancing, fault tolerance.
- Metrics for availability, reliability, failover cases and bandwidth usage. Silk performer has a rich set of metrics.
- Report Generation in html, graph plots, XML, MS Excel, MS word and PDF forms is desirable for the portability, security, and convertibility reasons.
- Support for post production analysis
- Monitoring Power consumption at various parts – desirable for power aware systems.

Any design of a performance test framework or tool needs to consider the above factors. Whenever a test plan is written, the above aspects need to be considered for preparing non-functional test requirement.

6. Conclusion

Performance testing is a more serious task than before, in the wake of emerging applications in medical, healthcare, real-time, and mission-critical fields. While the common criteria of performance such as response time and throughput seem trivial under normal conditions, they pose major challenges to mission-critical applications and advanced technologies such as .NET, J2EE, XML. To ensure quality, the process, and the third party tools used for development or testing need to be compliant with standards - such as CMMi, or FDA (Food and Drug Administration) - a standard used for medical and health applications, or standard such as MISRA - specific to communication, automotive, aerospace and other real time application. Performance testing needs to be started in the early stages of the product development cycle for better quality.

References

OpenGL (2012), “OpenGL - The Industry Standard for High Performance Graphics”, www.opengl.org.

H. Sarojadevi and S. K. Nandy (2011), "Processor-Directed Cache Coherence Mechanism – A Performance Study", International Journal on Computer Science and Engineering, Volume 3, issue 9, 3202-3206.

HP (2010), “HP Brings Affordable Performance Testing to the Cloud”, HP white paper, online - http://www.hp.com/go/loadrunnercloud.

Jay Philips (2010), "Words from a Purple Mind".

Borland (2006), “Choosing a Load Testing Strategy” , A Borland whitepaper.

Tom Kelley (2006), “The Art of Innovation”.

R.L. Hudson and E.B. Moss (1992), “Incremental collection of Mature Objects”, *Proceedings of the International Workshop on Memory Management*, Springer-Verlag, 388-403

James S. Collofello (1988), “Introduction to software verification and validation”, SEI curriculum module, CMU.

Glenford Myers (1969), “The Art of Software Testing”, John Wiley.

H. Sarojadevi is born in Udupi, Karnataka, India, and obtained PhD in Engineering from the Indian Institute of Science in 2003. The author has more than 20 years of experience that spans software development industry, research and education. The author’s major field of study is computer architecture and parallel processing.

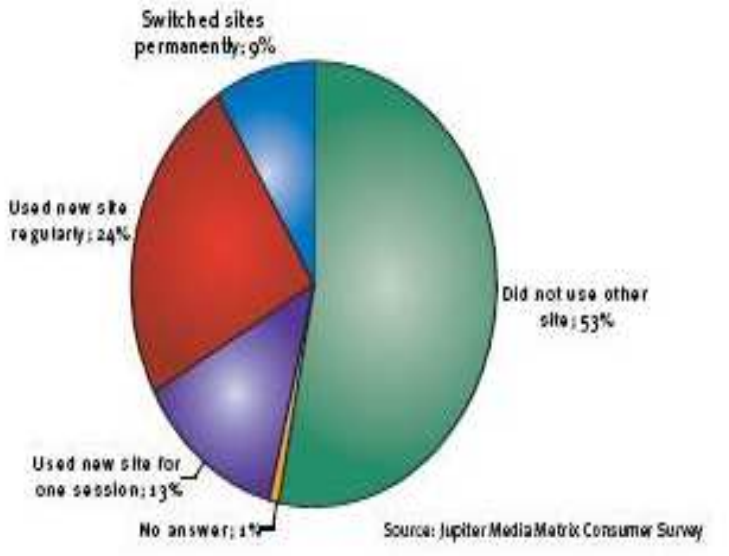

Figure 1. Jupitor media metrix report - 46% users abandon sites possibly due to performance problems

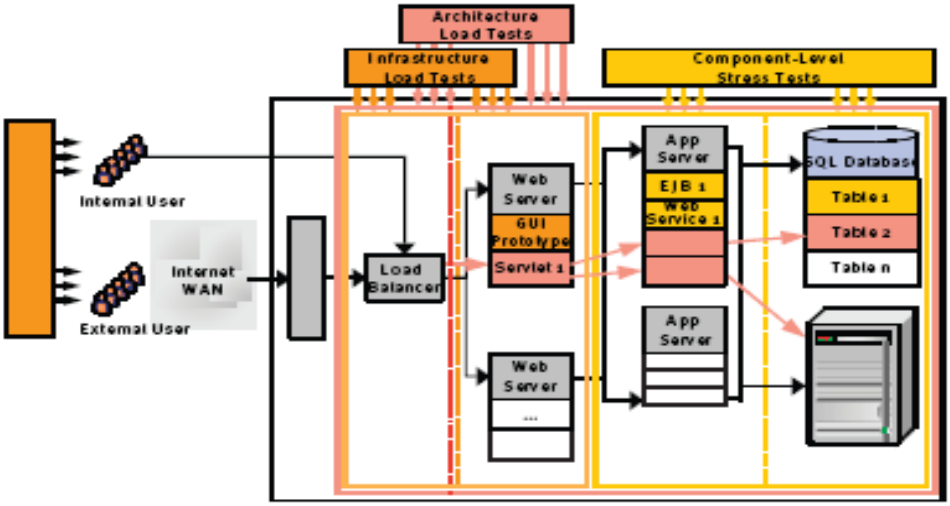

Figure 1: Product development phases in which load testing activities must be leveraged