ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
Indian Journal of Science and Technology, *Vol 9(41), DOI: 10.17485/ijst/2016/v9i41/100227, November 2016*

A Perspective View of different Techniques Used in Different Phases of Load Testing

Bhawna Jyoti* and A.K. Sharma

Himachal Pradesh University, Shimla, India;
bhawnashkl@gmail.com, sharmaas1@gmail.com

Abstract

Background/Objectives: To determine the quality and system characteristics, load testing has paramount importance in addition to the traditional functional testing. This study will be useful for the researchers to detect performance problems under load. **Methods/Statistical Analysis:** This comprehensive study is focussed on analysing different techniques for designing the load using workload intensity and workload mix in operational profiles as well as use of system models techniques and source code analysis techniques. We compare and contrast various approaches applied in load execution phase and load analysis phase. **Findings**: To deal with the load related problems and for recording system behaviour, realistic load test cases are designed to see functionality. Load execution testing can be done by human beings, load testing tools and also there is a use of special platforms. After completion of load test, performance analyst use tools to perform comparison against a predefined threshold values. **Applications/Improvements:** Although there are many existing techniques which shows correlation between performance metrics and anomalies but in practice, valid performance techniques are often limited. So, there is a need to put attention on these issues as a future work.

Keywords: Load Analysis Phase, Load Testing, Monitoring Test Execution

1. Introduction

In modern era, web based applications are gaining popularity in every sphere of life as they enable people to access greater volumes of data very easily. Large scale systems like Google, Amazon are growing rapidly in size and these systems handle complex services. Load testing is the most integral part of testing as it identifies functional and performance problems that are not being detected in unit and integration testing. The performance measurement of web based applications relies on the client, servers and networks. These application leads to heavy workloads on servers and load testing is needed for finding performance bottlenecks of application under use.

This organization of the paper is as follows:

Section 2 includes a comparative study of performance testing, stress testing and load testing; Section 3 focuses on the techniques that are being used for designing the load test; Section 4 gives a detailed description of load test execution; Section 5 covers the load analysis and Section 6 concludes our load related study and list some open research areas.

2. Comparative Study of Load, Performance and Stress Testing

The Load testing objectives are not clear in the early development stages[38] and are often defined during the

**Author for correspondence*

initial observation period in the load test. The term load testing is interchangeably used with performance testing and stress testing[9,15,25]. Performance testing is the process of determining effectiveness or speed of a system under test[2]. The framework for performance testing may be autonomic and workloads can be determined by measuring the system performance and analytically calculate the performance metrics[1,13,20,41].

Stress testing is focused on determining robustness, availability and reliability of system under test by applying extreme conditions. Bugs can be found by applying stress to a system that make the breaking point of system potentially harmful and these breaking points are identified to determine reliability of a system[15,25,58].

2.1 Performance Testing

It focuses on user satisfaction by analyzing the performance characteristics of the application like speed and scalability. It is used to detect mismatches between desired expectations and actual behavior of system. Sometimes this testing is unable to detect functional anomalies that are only detected under load[6,8,16,28]. Tests are being conducted on the field like environment and so there is always a degree of uncertainty when system is deployed in actual environment[7,17,29,33,36,50].

2.2 Load Testing

It is used for determining throughput and response time of system under test. Concurrency issues and functionality errors under load is being detected. Load testing is focused on feeding the system with largest task to detect the hardware limits of resource utilizations without compromising performance[11,23,30,47].

2.3 Stress Testing

It considers application issues that appear only under extreme conditions when stress is applied on the system[12,15,19,35]. Potential application issues include bandwidth of network, processor and memory cycles and breaking point of the disk capacity due to unpredictable usage patterns mostly found in web applications. Stress testing is very useful to diagnose memory leaks, deadlocks and security vulnerabilities by applying genetic algorithms in real time environments[22,45].

3. Design Phase of Load Test

The goal of the design phase includes to deal with load related problems and record system behavior when exposed and tested with load testing parameters. Based on the load testing objectives, there are two general approaches that are being designed for proper load:

- Designing Realistic Load test cases: These test cases are designed to see the functionality which resembles the actual usage environment once the system is operational in the field. For designing realistic loads dimensions like workload intensity and workload mix have been proposed by various authors[2,15,43,44].

Table 1. Load Test Design Techniques

Techniques for Designing the Load	Test Objectives	Usage Data/Data Sources	References
Load extrapolation based on workload intensity and workload mix	To detect functional and non-functional problems under load.	Past Usage Data And Operational Profiles	2,15,44,55
Loads which are derived using Stochastic Form-oriented models	To simulate realistic user behavior and to detect various load problems	Business Requirements,User Configurations And Operational Profiles	3,43

Table 1 Continued

Load design using Genetic Algorithms	To detect performance problems (high response time)	Resource Usage Per Request	5,19,12,21,22
Use-case based load design technique using UML models	To detect functional and non functional problems	UML Diagrams, UML Activity Diagrams, Operational Profiles.	23
Use-case based load design technique using Markov Models	To detect functional, non functional problems	Past Usage Data	45,59

- Designing load test cases by using various system models techniques or with source code analysis: These load design techniques are based on system's source code to generate load which exercise the load sensitive regions and detect memory related faults in the system[30,46]. In some systems, source code must not be directly accessible. So, various system models[24,40,47,56] are being used to design potential loads for system under test.

4. Execution Phase of Load Test

This phase consists of three steps:

Set up of system deployment in which load test has to be executed; Test cases generation and termination of the test cases after completion; Monitoring of test execution and recording of load related parameters.

For execution of test case, three approaches are being used which are discussed below:

- Testing load of a system by human beings: A group of human testers is being involved for load testing of a particular organization and they manually check the system performance under test[15]. But there is a scaling and timing issues between manual coordination between testers. Load tests that are executed once cannot be reproduced or repeated exactly as they occurred.
- Use of load drivers/load testing tools: For automatic generation of thousand and millions of concurrent requests for a longer period of time, load drivers are used. Based on the analysis of load testing tools, we found that commercial tools such as WebLOAD Professional or Neoload support every key feature supported for load testing[27]. On the other hand, APACHE Jmeter which is an open source java application from the Apache Software foundation being designed for distributed load testing[9,48] and PReWebN tool is used with large datasets[39].
- Use of Special Platforms: It includes service oriented systems in which load can be tested throughout the software development life cycle even before the system is completely developed[32.42].

In the next subsections, we compare and contrast various techniques applied in load execution phase. Section 4.1 includes setup, Section 4.2 discusses about how to generate and terminate the load, Section, 4.3 explains monitoring of test data.

4.1 System Deployment and Load Execution Set Up

It includes configuring the mail server and database server or to make any system under testing operational.

Test executing setup is done by configuring the load testing tools or acquiring group of human testers for load testing. Testers are acquired according to the specific criteria depending on the testing objectives. Two techniques are mainly used to create field like test bases:

(1) Use of raw data from the field data.
(2) Extracting the field database so that certain sensitive information is removed[26,34].

But system deployment for the special platforms is different as compared to group of human testing as discussed below:

- Model driven engineering platforms: These platforms are very useful when there are incomplete system components. Automated code generation is being done using domain specific modeling languages24.
- Use of special platforms to control interleaving threads: When problems like deadlocks and racing condition occurs, special platforms are being used to control such situations.

Some load drivers especially the benchmarks suits like[45,50,54] provides a simple GUI for load test practitioners to specify the rate of requests as well as data durations. Number of researchers has proposed to automatically generate load driver configuration code based on usage models[24]. Store and replay load driver configuration techniques are being used in web based applications[14,24] and distributed telecommunication applications[44,48].

4.2 Generation and Load Termination Techniques

Generation and Load Termination Techniques: Load can be generated statistically or dynamically. When user manually generates load, he conducts a sequence of actions over a fixed period of time[10]. Besides manual load germination, the other two techniques are widely used. These are:

- Measurement based static load identification: A controller component is being used to generate the specific load based on the configuration of system under test14. Load termination techniques are based on continuous, time-driven and counter driven that is being discussed in existing load drivers50,54. When load test is terminated the performance metrics of interest(example: response time, CPU and memory) are statistically drawn30,50.
- Measurement based dynamic load identification: Authors proposed automated approach that gives a comprehensive view of performance counters and execution logs to diagnose memory related issues in measurement based dynamic load execution18,46.

4.3 Monitoring of System Behavior and Recording of Data

Monitoring of system behavior and recording of data: Detailed monitoring of system under test is being done during the course of load test execution[2,17].

Collecting Throughput and Performance Metrics: Once the load test has been executed and being terminated, the data is collected for throughput and number of passed either periodically or recorded once at the end of load test. System resource usage metrics concerning CPU, memory and disk parameters are usually collected and recorded at fixed time interval[15,32,37.]

Recording of Runtime behaviour of the System by Gathering Execution Logs and System Resource Usage in Terms of Performance Counters is done[18,46].

5. Load Analysis Phase

After the completion of load test, performance analyst use tools to perform simple comparisons of the average of metrics against a pre-defined threshold. Recorded data must be analyzed to decide whether the system under test has fulfilled the desired objectives. Various data analysis techniques are used for this purpose[2,17,46]. It involves the following domains of actions to consider:

- Summarize system behavior into one number parameter and compare against a potential threshold value. The usual output is based on pass/fail criteria.

Table 2. Analysis Techniques for Validating load Test Objectives

Techniques for Validating test objectives	Data	Test objectives	References
Straight-forward comparisons	Performance metrics	To check scalability and performance requirements	16,57
Comparing against derived data from previous baseline	Number of pass/ fail requests, past performance data	To detect validations of performance and reliability requirements	2,47,60
Detecting throughput problems	Throughput response time metrics	To detect load related functional problems and see violations in scalability requirements	26,36,37
Detecting anomalous behavior of system by using execution logs	Execution logs	To detect performance and functional problems.	16,47,53,60
Statistical Analysis	Periodic sampling metrics	Detecting unexpected behaviour in performance requirements	2,5,17,47
Empirical performance analysis	Performance metrics	To detect functional problems in system under test	5,6,16,17,26,50
Detecting anomalous behavior of system by using memory leaks	Memory usage metrics	Detecting load related functional and memory problems by seeing memory leaks	30,46
Detecting anomalous behavior of system by using performance counters	Performance counters	To detect potential performance bottlenecks	6,3660

- Examining the systems behavior to locate patterns of known problems.
- Anomaly detection focused on analysis of resource usage data.
- Statistical analysis is done to calculate the performance metrics and report is prepared.

6. Conclusion and Future Work

To ensure the proper functioning of the dynamically composed web services, load testing is required along with traditional functional testing procedures. In this paper, we provides a comprehensive view of load testing

phases and addresses some areas of research which are still needed to be explored. So, load testing evaluates the systems with various performance objectives to meet service level agreement. Here, we have some open research areas which are still need to be explored thoroughly:

In load testing, recorded data (example: metrics and logs) is very large due to which system overheads are maximized[50]. Additional work is needed to find proper system monitoring data that will minimize the system monitoring overhead.

Web sites are increasing their reliability on dynamically composed web services[44]. So, there is a need to collect more data to achieve statistically meaningful results that is affected by variability in parameters measurement.

There are no techniques developed which focus on improvement of the fault-inducing based load design techniques[23,43]. Less work is done to combine the strength of various fault inducing techniques so that the overall testing of load can achieve multiple objectives.

Although there are many existing techniques[52] which shows correlation between performance metrics and anomalies[16] and service level agreements determine occurrence of anomaly but in practice, valid performance techniques are often limited. So, more work is needed to identify metric deviation as there could be phase shift in performance test.

When there is a rise of new requirement then system is updated. Variability in metric values[28] will have adverse effect on the confidence parameter in metric values. There is need to pay attention to modify test suite and studies can be done on sliding window to select test cases in the existing dataset.

7. References

1. Janani V, Krishnamoorthy K. Evaluation of cloud based performance testing for online shopping websites. Indian Journal of Science and Technology. 2015 Dec; 35(8).
2. Kalita M, Khanikar S, Bezboruah T. Investigation on performance testing and evaluation of PReWebN: a Java technique for implementing web application. IET Software. 2011Oct; 5(5):434-44.
3. Avritzer A, Weyuker J. The Automatic Generation of Load Test Suites and the Assessment of the Resulting Software. IEEE Transactions On Software Engineering.1995 Sep; 21(9).
4. Yang CSD, Pollock LL. Towards a structural load testing tool. Proceedings of the 1996 ACM SIGSOFT international symposium on Software testing and analysis (ISSTA).1996; 201-08.
5. Matheus SL. An Genetic Algorithm Approach for Profiling Computational Performance Measures. Peer J. Pre. Prints. 2015 Feb.
6. Malik H, Jiang ZM, Adams B, Hassan E, Flora P, Hamann. Automatic Comparison of Load Tests to Support the Performance Analysis of Large Enterprise Systems. IEEE 14th European Conference on Software Maintenance and Re-engineering. 2010 March.
7. Maplesden D, Tempero E, Hosking J, Grundy JC. Performance Analysis for Object-Oriented Software: A Systematic Mapping. IEEE Transactions on Software Engineering. 2015; 41(7).
8. Sarojadevi H. Performance Testing: Methodologies and Tools. Journal of Information Engineering. 2012; 2(3).
9. Grguri J, Mošmondor M, Lazarevski P. Load testing and performance monitoring tools in use with AJAX based web Applications. MIPRO. Opatija, Croatia. 2010 May; 24-28.
10. Avritzer A, Weyuker EJ. Generating test suites for software load testing. Proceedings of the ACM SIGSOFT International Symposium on Software Testing and Analysis ISSTA. 1994; 44–57.
11. Lim BH, Kim JR, Shim KH. Hierarchical Load Testing Architecture Using Large Scale Virtual Clients. Proceedings of IEEE International Conference on ICME. 2006.
12. Briand LC, Labiche Y, Shousha M. Stress testing real-time systems with genetic algorithms. Proceedings of the Conference on Genetic and Evolutionary Computation (GECCO), 2005, p. 1021-28.
13. Weyuker EJ, Avritzer A. A metric for predicting the performance of an application under a growing workload. IBM System Journal. 2002 Jan; 41(1):45-54.
14. Hill J, Schmidt D, Edmondson J, Gokhale A. Tools for continuously evaluating distributed system qualities. IEEE Software Journal. 27(4); 2010 Jul; 65-71.
15. Barna. C, Litoiu M, Ghanbari H. Autonomic load testing framework. Proceedings of the 8th ACM International Conference on Autonomic Computing (ICAC), 2011.
16. Foo K, Jiang ZM, Adams B, Hassan AE, Zou1 Y, Flora P. Mining Performance Regression Testing Repositories for Automated Performance Analysis. Proceedings IEEE International Conference on Quality Software, 2010, p. 32-41.

17. Byoungju C, Yoon M, Kim H. An Approach to Developing a Performance Test Based on the Tradeoffs from SW Architectures. Journal of Software Engineering and Applications. 2013; 6:184-95.
18. Malik H, Jiang Z, Adams B, Flora P, Hamann G. Automatic comparison of load tests to support the performance analysis of large enterprise systems. Proceedings of the 14th European Conference on Software Maintenance and Reengineering (CSMR). 2010 Mar.
19. Alesio SD, Briand LC, Nejati S, Gotlieb A. Combining Genetic Algorithms And Constraint Programming To Support Stress Testing Of Task Deadlines. ACM Transactions on Software Engineering and Methodology. 25(1); 2015 Dec.
20. Menasce DA, Almeida VAF, Fonseca R, Mendes MA. A methodology for workload characterization of e-commerce sites. Proceedings of the 1st ACM Conference on Electronic Commerce (EC). 1999; 119-28.
21. Afzal W, Torkar R, Feldt R. A systematic review of search-based testing for non-functional system properties. Information and Software Technology. 2009; 957-76.
22. Briand LC, Labiche Y, Shousha M. Using genetic algorithms for early schedule ability analysis and stress testing in real-time systems. Genet Program Evolvable. 2006 Mar; 145-70. DOI 10.1007/s10710-006-9003-9.
23. Garousi V, Briand LC, Labiche Y. Traffic-aware Stress Testing of Distributed Systems Based on UML Models. Proceedings - International Conference on Software Engineering, 2006 Jan.
24. Wang X, Zhou B, Li W. Model Based Load Testing of Web Applications. IEEE International Symposium on Parallel and Distributed Processing with Application, 2010. 978-0-7695-4190-7.
25. Lim BH, Kim JR, Shim KS. A Load Testing Architecture For Networked Virtual Environment. 8th IEEE International Conference Advanced Communication Technology, 2006.
26. Bora A, Bezboruah T. Investigation on Security Implementation and Performance Aspects of MedWS: a Hierarchical SOAP based Web Service. International Journal of Database Theory and Application. 7(14); 2014:169-88.
27. Ho CW, Williams L. Deriving Performance Requirements and Test Cases with the Performance Refinement and Evolution Model (PREM). Proceedings of 15th IEEE international Requirement Engineering Conference, 2007, p. 79-88.
28. Malik H, Adams B, Hassan AE, Flora P, Hamann G. Using Load Tests To Automatically Compare the Subsystems of a Large Enterprise System. Proceedings of the 14th European Conference on Software Maintenance and Reengineering (CSMR), Mar 2010.
29. Luo Q, Nair A, Grechanik M, Poshyvanyk D. FOREPOST: Finding Performance Problems Automatically with Feedback-Directed Learning Software Testing. Empirical Software Engineering Journal, 1382-3256, Springer, US.
30. Mohamad S, Joao B, Cangussu W. Automatic Stress and Load Testing for Embedded Systems. Proceedings of COMPSAC. 2006 Sep; 2:229-33.
31. Grechanik M, Csallner C, Fu C, Xie Q. Is Data Privacy Always Good For Software Testing? Proceedings of the 2010 IEEE 21st International Symposium on Software Reliability Engineering (ISSRE), 2010 Nov, p. 368-77.
32. Bora A, Bezboruah T. Testing and Evaluation of a Hierarchical SOAP based Medical Web Service. International Journal of Database Theory and Application.; 2014; 7(5):145-60.
33. Malik H, Adams B, Hassan AE. Pinpointing the subsystems responsible for the performance deviations in a load test. Proceedings of the IEEE 21st International Symposium on Software Reliability Engineering (ISSRE 2010), 2010 Nov.
34. Barros MD, Shiau J, Shang C, Gidewall K, Shi H, Forsmann J. Web Services Wind Tunnel: On Performance Testing Large-scale Stateful Web Services. Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2007, 612-17.
35. Meira J, Almeida ECD, Suny G, Traon YL, Valduriez P. Stress testing of transactional database systems. Journal of Information and Data Management (JIDM). 2013; 4(3):279-94.
36. Snellman N, Ashraf A, Porres I. Towards Automatic Performance and Scalability Testing of Rich Internet Applications in the Cloud. Proceedings of the 37th IEEE EUROMICRO Conference on Software Engineering and Advance applications, 2011.
37. Duttagupta S, Nambiar M. Performance Extrapolation for Load Testing Results of Mixture of Applications. Proceedings of the Fifth UK Sim European Symposium on Computer Modelling and Simulation (EMS), 2011 Nov.
38. Denaro G, Polini A, Emmerich W. Early Performance Testing of Distributed Software Applications. ACM SIGSOFT Software Engineering, 2004.
39. Yi T. Comparison Research of two typical UML Class Diagram Metrics Experimental Software Engineering. Proceedings of the IEEE International Conference on Computer Application and System Modelling, 2010.

40. Westermann D, Happe J. Towards Performance Prediction of Large Enterprise Applications Based on Systematic Measurements. In: Proceedings of the Fifteenth International Workshop on Component-Oriented Programming (WCOP), 2010.
41. Che X, Maag S. Passive Testing on Performance Requirements of Network Protocols. 27th IEEE International Conference on WAINA, 2013 Mar.
42. Sharma AK. Empirical Analysis of web service testing tools. IJETMAS, 2015.
43. Draheim D, Grundy J, Hosking J. Realistic Load Testing of Web Applications. Proceedings of the Conference on Software Maintenance and Reengineering (CSMR); 2006, 57-70.
44. Menasce DA. Scaling the web load testing of web sites. IEEE Internet Computing, 2002, 1089-7801/02.
45. Casale G, Kalbasi A, Krishnamurthy D, Rolia J. Automatic Stress Testing of Multi-Tier Systems by Dynamic Bottleneck Switch Generation. Proceedings of the ACM 10th International Conference on Middlewar, 2009, Springer-Verlag.
46. Syer MD, Jiang ZM, Nagappan M, Hassan AE, Nasser M, Flora P. Leveraging Performance Counters and Execution Logs to Diagnose Memory-Related Performance Issues. Proceedings of the 2013 IEEE International Conference on Software Maintenance (ICSM), 2013.
47. Baone CA, Veda S, Pan Y, Premerlani W, Dai J, Johnson A. Measurement based Static Load Model Identification. IEEE Poer and Energy Society General Meeting, 2015; 978-1-4673-8040-9.
48. Karr AF, Porter AA. Distributed Performance Testing using Statistical Modelling. In: ACM SIGSOFT Software Engineering Notes, 2005.
49. Harrold MJ. Testing A Roadmap. In Future of Software Engineering, Ireland, ACM Publications, 2000.
50. Dujmovic JJ. Universal Benchmark Suites. 7th IEEE International Symposium of Modeling Analysis and Simulation of Computer and Telecommunication Systems, 1999.
51. Bernald DJ, Watkins. Investing The Performance of Genetic Algorithm Based Software Case Generation. IEEE International Symposium of High Assurance System Engineering, 2004.
52. Jiang ZM, Hassan AE. A Survey on Load Testing of Large Scale Software Systems. IEEE Transactions on Software Engineering, 2015 Jun.
53. Jiang ZM, Hassan AE, Hamann G, Flora P. Automatic Identification of Load Testing Problems. In: Proceedings of the 24th IEEE International Conference on Software Maintenance (ICSM), 2008.
54. Kalibera T, Tuma P. Precise Regression Benchmarking with Random Effects: Improving Mono Benchmark Results. In: Proceedings of 3rd European Performance Engineering Workshop, EPEW, Budapest, Hungary, Springer Berlin Heidelberg, 2006 Jun.
55. Calzarossa M, Serazzi G.Workload Characterization: A Survey. In: Proceedings of IEEE, 1993.
56. Dillenseger B. CLIF, a framework based on Fractal for flexible, distributed load testing. in Annual telecommunications; Springer-Verlag, France, 2008.
57. Yan M, Sun H, Liu X, Deng T, Wang X. Delivering Web service load testing as a service with a global cloud. Concurrency and Computation: Practice and Experience, 2014.
58. Krishnamurthy D, Rolia J, Majumdar S. SWAT: A Tool for Stress Testing Session-based Web Applications. International CMG Conference ,2003.
59. Avritzer A, Weyuker EJ. The Automatic Generation of Load Test Suites and the Assessment of the Resulting Software. IEEE Transactions on Software Engineering. 1995 Sep; 21(9).
60. Jiang ZM, Hassan AE, Hamann G, Flora P. Automated Performance Analysis of Load Tests. IEEE Conference on Software Maintenance, 2009.