VOL. 5, NO. 11, November 2015 ISSN 2225-7217
ARPN Journal of Science and Technology

http://www.ejournalofscience.org

A Survey of Most Common Referred Automated Performance Testing Tools

[1] Muhammad Sadiq, [2] Muhammad Shahid Iqbal,
[3] Amizah Malip, [4] Wan Ainun Mior Othman
[1] Faculty of Computing (RIU) Islamabad
[2] School of Computer Science, Anhui University, Hefei, China
[3,4] Institute of Mathematical Science, University of Malaya
[1] sadiq paec@riu.edu.pk, [2] nawabishahid@yahoo.com, [3] amizha.mal ip@um.edu.my, [4] wanainun@um.edu.my

ABSTRACT

System performance is one of the key accounts in any software quality. Automated software performance can be measured with the help of testing based upon some predefine characteristics. Automated performance testing is very important for large scale and distributed applications. Unsatisfactory performance may create functional and non-functional problems which must lead to inference in terms of time and resources. In standards regulation authorities have propose matrix for automated software performance testing. In this discuss some software quality standards and their purposed matrix. We review some literatures for performance measurement based upon some characteristics or matrix. We then combined all characteristics in the form of matrix. We analyze some most commonly referred automated software performance testing tools to verify the practical implementation of purpose matrix.

Keywords: *Automated, performance testing, response time, parameters*

1. INTRODUCTION

Testing can be divided into two types which are functional and non-functional [11], [122]. 'Functional tests can determine whether an application achieves what it is expected to from a business perspective'. It is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.

Functionality testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

Meanwhile performance elaborates non-functional [122].

Software product testing has great importance in an error detection of a software development it reflects directly on the software quality enhancement before its wrong implementation. Testing is widely used in industry for quality assurance [32]. Software testing is a formal process carried out by a specialized testing team. The team will examine software unit, several integrated units or an entire software package by running the programs on a computer. All the associated will be performed according to approved test procedures [76]. In software development life cycle, software testing is highly recommended to assure the quality of the software process and product. 'Researchers and practitioners' came-up with a variety of different software tools to automate the testing process.

Literature survey shows that two appropriate methods are commonly used to measure for performance testing. Which are Load testing and stress testing. Load testing assesses how a system performs under a given "load." The rate at which transactions are submitted to the system is called the load. One of load testing objectives is to determine the maximum sustainable load the system can handle [121]. Meanwhile web application load is defined in terms of concurrent users or HTTP connections. "Stress testing refers to subject a system to an unreasonable load with the intention of breaking it. A stress test denies a system the resources (e.g., RAM, disk, interrupts, etc.) needed to process a certain load. It is designed to cause a failure. It tests the system's fault recovery capability [121]. Trying to break the system under test by overwhelming its resources or by taking resources away from it. The purpose is to make sure that the system fails and recovers gracefully as double the baseline number of concurrent Users/HTTP connections.

Randomly shut down and restart ports on the network. If you cannot measure it, you cannot improve it [26]. We analyze our literature with this concept. We will find commonly used approaches to measure the performance. 'Section two describe, the existing methodologies or standards and they proposed matric, meanwhile section three discuss literature review. On section four presented most common refers tools, their features and implementation description. In section five we describe proposed metrics and guidelines for the development of automated performance testing tools.

2. EXISTING STATE OF THE ART/METHODOLOGY

There are number of quality monitoring organization and agencies which monitor guides and regular standards to measure the parameters. In this regard we check the available standards of software performance which is IEEE STD 1061 -1992. In 1998 another revision some revision were make but parameters for performance remains same. Another revision was 2009 however the parameters for performance remain as per previous IEEE Standard for a Software Quality Metrics Methodology describes the performance base on these parameters.

Table 1

Parameters	Description
Efficiency	Relative extent to which a resource is utilized (for example, storage space, processing time, or communication time).
Integrity	Extent to which the software will perform without failures due to unauthorized access to the code or data within a specified time period.
Reliability	Extent to which the software will perform without failures within a specified time period.
Survivability	Extent to which the software will perform and support critical functions without failures within a specified time period when a portion of the system is inoperable
Usability	Relative effort for training or software operation (for example, familiarization, input preparation, execution, or output interpretation).

As per ISO/IEC 9126 performance is measure in terms of efficiency meanwhile the parameters for measure are time behavior, resource utilization and efficiency compliance. In 2010 ISO/IEC FDIS 25010:2010(E) slandered efficiency changes to performance efficiency and the parameters are time behavior, resource utilization and capacity. Time behaviors refer to response time and throughput.

Performance efficiency as per ISO/IEC FDIS 25010:2010(E).

Performance relative to the amount of resources used under stated conditions.

Table 2

Parameters	Description
Time behavior	Degree to which the response and processing times and throughput rates of a product or system.
Resource utilization	Degree to which the amounts and types of resources used by a product or system when performing its functions meet requirements.
Capacity	Degree to which the maximum limits of a product or system parameter meet requirements.

3. RELATED WORK

A large amount of research has been done over the last few decades to address the problem of analyzing the performance of software systems. In general, much of the work describes methodologies that include modeling the software systems, capturing the performance characteristics and transforming the software specification models into performance models. Gathering data to assess the performance is an essential aspect of software performance engineering approach. In this section, we review some of the methodologies and tools used for software performance assessment and techniques used for gathering and capturing performance parameters (data), which are highly dynamic and uncertain. Flippes, I. vokolos et al. [1998] uses response time, throughput, latency and resource utilization for software performance testing [22] and in (2000) introduces a new approach with same parameters [20]. Huebner [2001] discussed performance testing for IP services and systems via load testing and stress testing based on response time and resource utilization [121]. Lundberg et al. [2001] discussed the conflicts and Trade-Offs between software performance and maintainability and the focus performance matrix were response time, throughput, and resource utilization and compare the real time system and normal routine systems. Ulrich Herzog (2001) present a new technique of performance validation for software/hardware systems based on response time, throughput, latency and reliability [12]. Giovanni Denaro and Andrea Polini (2004) applies performance testing on distributed applications based on response time, throughput, scalability and latency parameters [23].

Nenad Stankovic (2006) discussed some patterns and tools for performance testing the tool address functional aspect of test case implementation and execution and behavioral and automation issues [3].

Vibhu Saujanya Sharma (2006) quantify software performance base on response time, latency, throughput resource utilization, scalability, reliability and security and discuss the trade Offs between various attributes and the importance of optimize system configuration [1].

Murray Woodside and Greg Franks (2000), (2007) describes software performance in terms of time lines and capacity [7],[17]. Vittorio Cotellessa and Pierluigi Pierini (2007) integrate different models for performance analysis based upon time lines and resource utilization [24]. Chih-Wei Ho and Laurie Williams (2007) present a software performance model based on response time and throughput [89]. Yan Jin and Jun Han (2007) discussed software performance for legacy information system based upon response time, throughput and resource utilization [21] George Din (2008) present a test design process for performance testing the target was high response time, scalability and resource utilization[8].

Gwang-hun Kim and Hui-choun Moon (2009) discussed the software performance scheme using virtualization technology based on time behavior, scalability and resource utilization [86]. Hangjung Zo (2010) discussed the security and performance in service oriented applications based on response time, latency, throughput and security and also discussed the tradeoff among these parameters [14]. Li Yuanyuan (2010) presents a method to test Linux software performance the focus parameters was time behavior, scalability, and resource utilization [18]. H. Sarojadevi (2011) discussed

the methodologies and tools for software performance testing the focus parameters was response time, throughput, availability, scalability and also discussed some familiar tools [127].

D. Evangelin Geetha and T.V Suresh Kumar (2011) present a technique for predicting the software performance during feasibility study based upon aspect response time, latency, and throughput and resource utilization [4]. Ibidokum Emmanuel Tope and Pavol Zavarsky (2011) present performance evaluation of Oracle VM server Virtualization based upon response time, latency, scalability and resource utilization[19].

Jovica Durkovic and Jelica Trninic (2011) discussed software performance based upon response time, throughput, latency, security, and scalability and perform the analysis via stress testing and load testing [11]. Donghun Lee (2012) present a new methodology for software performance monitoring Using aggregated performance Metrics based upon time behavior of overall software[9]. Jimoh, R.g & Abikoye (2012) present a software performance evaluation algorithm experiment for in-house software the parameters was speed, reliability, availability and customer objectives [90]. Jing XU (2012) discussed software performance diagnosis and improvements and the followed metrics was response time, throughput, latency and resource utilization [13].

4. MOST COMMON REFERRED PERFORMANCE TESTING TOOLS

4.1 Grinder: [55] ,[33] ,[56], [38], [49], [58], [59], [60].

The Grinder works on any hardware platform and any operating system that supports J2SE 1.4 and above. It can simulate web browsers and other devices that use HTTP, HTTPS, and it can be used to test Web Service interfaces using protocols such as SOAP and XML-RPC, databases using JDBC. MOM based systems using different types of protocols such as IIOP, RMI/IIOP, RMI/JRMP, and JMS. It can be used to test systems that utilize other protocols such as POP3, SMTP, FTP, and LDAP [94].

4.1.1 Response Time, Throughput, Latency

On the Grinder's console there are two tabs which display information about The Grinder and its tests.

1st are graphs and 2nd are results. Grinder analyzer parses the data from grinder and generates Graph for response time, throughput, and latency. When user perform the test grinder shows Statistics, Reports, Charts to represent the response time, throughput and latency for tests [96], [97], [95] , [98].

4.1.2 Scalability

After performing test the grinder shows the bandwidth used by different test and different scenarios at different load level like response time at different loads etc. [96].

4.1.3 Resource Utilization

Grinder generates different Graphs at different loads which shows the bandwidth usage and resource utilization like memory, bandwidth and CPU utilization [96], [97], [95], [98].

4.1.4 Security

Grinder provides HTTP/J2EE form based authentication which provides a more complex HTTP example based on an authentication conversation with the server. This HTTP script describes the path based response and J2EE Servlets describes common model for form based authentication. When user tries to access a secured resource J2EE servlets challenges with logon page and logon page POSTs to J security check page. [96], [97], [95], [98].

4.2 Apache JMeter: [43], [40], [51], [67], [68], [55], [73] [69], [70], [38], [57], [71], [72], [60], [67].

Apache JMeter is a powerful desktop performance testing tool from the Apache Jakarta project, written in Java, for load-testing web pages, web applications, and other static and dynamic resources including databases, files, Servlets, Perl scripts, Java Objects, FTP Servers, and more. The main component in JMeter is the 'Java Swing-based Graphical User Interface' which can be used for both Scripting and Execution [99], [100], [101], [102], [103], [104].

4.2.1 Response Time

Apache JMeter provides the response time of tests at different load point and also provides the average response time of each test [99], [100], [101],[102], [103], [104].

4.2.2 Throughput

JMeter have an option of summary reports which creates a table row for each request. The throughput can be calculated from the point of view of the sampler target (e.g. the remote server in the case of HTTP samples).

JMeter stores the total time which the request have taken in his account. So if other samplers and timers are belonging from the same thread, this will increase automatically the total time, so it will reduce the throughput value. With this scenario we can say that two identical samplers with different names will have half the throughput of two samplers with the same name. So it is very important to choose the sampler labels correctly which will help to get the best results from the Report [99], [100], [101], [102], [103], [104].

4.2.3 Latency

The summery report of apache JMeter show the latency of each request at different loads level [99], [100], [101], [102], [103], [104].

4.2.4 Scalability

Apache JMeter is used for Load testing and stress testing. With the help of these tests we can measure the scalability of our product because we can test our

product with multiple users at a time. [99], [100], [101], [102], [103], [104].

4.2.5 Resource Utilization

Report analyzer tells us about the CPU and other resource usage at different load level [99], [100], [101], [102], [103], [104].

4.3 Silk performer: [85], [52], [86], [57], [87], [58], [88].

4.3.1 Response Time

In reports The Transactions section contains summary measurements in a tabular form, that is, aggregate measurements for all transactions of the specific user. For every transaction, the transaction response time and the transaction busy time are displayed. The transaction response time is measured from the beginning to the end of the transaction. [105], [106], [107], [108].

4.3.2 Throughput

Summary table for each user group provides detailed measurements in tabular form. The measurements include transaction response times, individual timers, counters, and response time and throughput measurements related to the type of application that was tested (Web, database, CORBA, or TUXEDO). In addition, errors and warnings for all user groups [105], [106], [107], [108].

4.3.3 Scalability

Silk performer Ensure the scalability, performance, and reliability of your enterprise applications. Silk Performer ensures the quality of your enterprise applications by measuring their performance from the end-user perspective, as well as internally, in a variety of workload scenarios and dynamic load conditions [105], [106], [107], [108].

4.3.4 Resource Utilization

Log file for each test is created which shows the report regarding the tests including resource utilization [105], [106], [107], [108].

4.4 Mercury Interactive Load Runner:[42] ,[78] ,[44] [46] ,[48] ,[49], [39], [41] ,[65].

4.4.1 Response Time

The Transaction monitor displays the transaction rate and response time during scenario or session step execution. Transaction Response Time - Whole Scenario graph, lets you monitor the Amount of time it takes for each transaction to be completed [109], [110], [111].

4.4.2 Throughput

The Throughput graph shows the amount of data (measured in bytes) that the Users receive from the server at any given second. You can compare this graph with the Transaction Response Time graph to see how throughput affects transaction performance [109], [110], [111].

4.4.3 Latency

FTP Pass Latency shows the Interval between transmitting a FTP PASS packet and receiving a response in msec [109], [110], [111].

4.4.4 Scalability

Running Users - Whole Scenario graph displays the number of Users running and the behavior of Users at a given time [109], [110], [111].

4.4.5 Resource Utilization

Windows Resources graph displays the Windows resources measured during a scenario [109], [110], [111].

4.4.6 Security

When you run certain security scripts, you can use Load runner's security graphs to view information about the simulated attacks on the server [109], [110], [111].

4.5 IBM Rational Performance Tester: [61][62][63][65][66]

4.5.1 Response Time

Select the Page Performance tab. This tab presents a bar graph of the average page response time for the 10 pages with the highest times. The Summary page displays the following Page Summary information. The average response time for all pages. The maximum response time for all pages, the minimum response time for all pages [112], [113], [114], [115].

4.5.2 Throughput

Byte counters provide throughput information regarding the rate and the number of bytes sent and received during a sample interval and during a run [112], [113], [114], [115].

4.5.3 Resource Utilization

There are reports to measure network and CPU utilization, to identify the presence of hardware bottlenecks. You can monitor your resources via reports [112], [113], [114], [115].

4.5.4 Scalability

During load testing the no of users and behavior of application at different loads enable to scale the product for optimize use [112], [113], [114], [115].

4.5.5 Security

In IBM Rational Performance tester user can define his security algorithms by implementing custom security Java™ interfaces which can be used in the security editor. With customize security algorithms. User can implement proprietary security algorithms that transform the XML before sending to and after receiving from the server [112], [113], [114], [115].

4.6 Open STA. [123][124][125][74][75][77]

4.6.1 Response Time

Open STA provides different types of data collection and monitoring functions. When user runs a Test it generates wide range of results data which is collected automatically tool. It includes response times of virtual users and resource utilization of r test. User can also create and reference Collectors in for his Tests to enhance the Test-run monitoring. Data collection options are also available [117], [118], [119], [120].

4.6.2 Throughput

Its NT Performance Collectors collects the performance data during the Execution of test from performance objects. Each performance object has set of performance counters which are associated with performance objects. It helps to measure throughput [117], [118], [119], [120].

4.6.3 Latency

Min Request Latency indicates that the minimum length of time elapsed in milliseconds between sending of an HTTP request and receiving the results of request [117], [118], [119], [120].

4.6.4 Scalability

The behavior of application at different loads helps us to measure the scalability of application. The resultant graphs show the total and average behavior and bandwidth used [117], [118], [119], [120].

4.6.5 Security

A proxy server acts as a security barrier between your internal networks (Intranet) and the Internet, keeping unauthorized external users from gaining access to confidential information on your internal network. This is a function that is often combined with a firewall [117], [118], [119], [120].

4.7 HP Load Runner: [44], [52], [55], [39], [73], [74], [84], [75], [65], [60], [76],[77]

Load Runner is a software testing tool which enables you to test a range of applications including mobile, Ajax, Flex, HTML 5, .NET, Java, GWT, Silverlight, SOAP, Citrix, ERP and legacy. It isolated performance bottlenecks with system and end-user monitoring and a wide range of analysis components.[91]

4.7.1 Response Time

A view of HP Load Runner Analysis cross-results trending capabilities, showing a comparison of increased System scalability and optimized response time performance. We can monitor the response time of requests with the help of resultant graph. Transaction Response Time Whole Scenario graphs. Shows the amount of time it takes for each transaction to be completed. [92], [93]

4.7.2 Throughput

The Throughput graph shows the amount of data (measured in bytes) that the users receive from the server at any given second. You can compare this graph with the Transaction Response Time graph to see how throughput affects transaction performance. If the throughput scales upward as time progresses and the number of users increase, this indicates that the bandwidth is sufficient. If the graph were to remain relatively flat as the number of users increased, it would be reasonable to conclude that the bandwidth is constraining the volume of data delivered [93].

4.7.3 Latency

Hits per Second - Whole Scenario graph. Displays the number of hits (HTTP requests) made to the Web server by users during each second of the scenario run. And also shows the delays [93].

4.7.4 Scalability

Running users - Whole Scenario graph displays the number of users running at a given time [93].

4.7.5 Resource Utilization

Windows Resources graph displays the Windows resources measured during a scenario [93].

5. PURPOSED STATE OF THE ART/METHODOLOGY

After literature review we find some common points to establish slandered criteria for performance measure. To measure the software performance of any kind of software all these parameters are necessary but the level of assurance depends on user requirements.

5.1 Response Time

Response time which is defined as the duration from when a client makes a request until the entire file is received by the client [8].

5.2 Throughput

Throughput denotes the number of operation that can be completed in a given period of time [23].

5.3 Latency

Latency Typically Describes the Delay between request and completion of an event [23].

5.4 Scalability

Scalability is the ability of a system to continue to meet its response time or Throughput objectives as the demand for the software function increases"[27] "Scalability identifies the dependency between the no of distributed system resources that can be used by an application (typically no of hosts) [23].

5.5 Resources Utilization

Typical Resources that considered include network bandwidth requirement, CPU cycle, disk space, disk access operation, and memory usage [10], [11].

5.6 Security

Ensure that the security requirements are meeting [7].

One of the main objectives of performance testing is to help maintain the system with High response time, low latency, high throughput, low resource utilization, High Scalability and high security.

Table1: Common Referred tools and feature mapping

X = Not implementing
✓ = Implementing Properly
P = Implementing Partially

S.No	Name of Tools	Parameters					
		Response Time	Latency	Throughput	Scalability	Resource Utilization	Security
1	HP Load runner	✓	✓	✓	P	✓	X
2	Grinder	✓	✓	✓	X	✓	P
3	Apache JMeter	✓	✓	✓	✓	P	X
4	Silk performer	✓	✓	✓	P	P	X
5	Mercury Interactive Load runner	✓	✓	✓	X	✓	✓
6	IBM rational Performance	✓	✓	✓	P	✓	P
7	Open STA	✓	✓	✓	X	X	✓

In this table 1 we can easily identify tools that are not implementing parameters of performance testing. Some tools archives partial implementation such as web application. While some are implementing a single aspect of parameter and some are totally neglecting the standard defined parameters. Some tools are implementing some extra features so it is recommended to add such type of features in standards due to its wide utilization in industries. It can easily noticed that response time, latency and throughput is implemented by all most common referred tools but scalability, resource implementation and security is not implemented by all common referred tools.

Security is very key component of performance testing with respect to current era.

Figure1 shows the detail overview of tools with respect to features implementations. We can say figure 1 is graphical representation of table 1. Mercury Interactive load runner is comparative better because it is implementing more performance parameters then other tools.

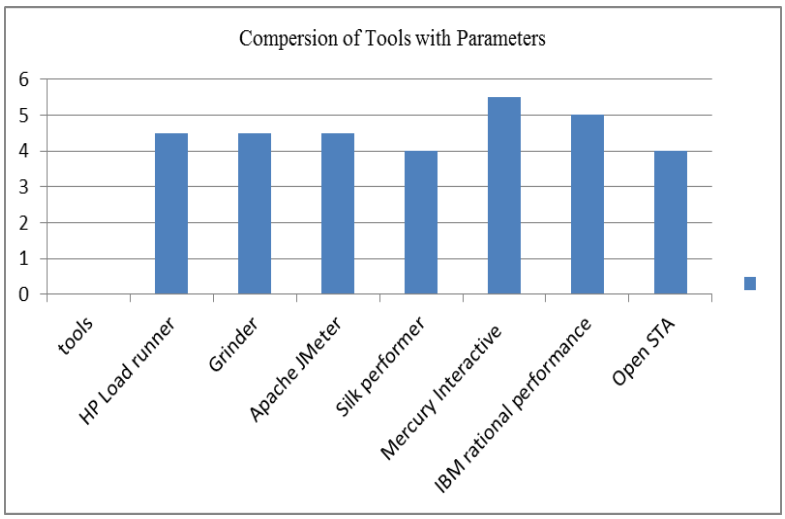

Figure1

The Figure shows tools and implementation of parameters. HP Load runner, Grinder and Apache JMeter implement four parameters completely and one partially. Silk performer and Open STA are least implement parameters. IBM rational Performance implements four parameters and two partially. Mercury Interactive Load runner implements complete five parameters.

6. CONCLUSION

There are so many tools are available but not a single which are completely implement the parameters of performance.

In this paper we discuss tools base performance process and their parameters. In this paper no signal tool have completely implement the whole parameters. Also in this paper we identifies the parameters which are implemented any automated performance testing tools.

This paper helps for the development of performance testing tools. Analysis some most common referred automated software performance testing tools from literature to verify the practical implementation of purposed matrix. In this way we can identify gaps of other testing tools like static testing, functional testing, test case management etc. In this way we can identify the gap and provide guidelines to develop tools in future in better way.

REFERENCES

[1] Vibhu Saujanya Sharma and Kishor S. Trivedi. 2007. Quantifying software performance, reliability and security: An architecture-based approach. J. Syst. Softw. 80, 4 (April 2007), 493-509. DOI=10.1016/j.jss.2006.07.021 http://dx.doi.org/10.1016/j.jss.2006.07.021

[2] H. Sarojadevi" Performance Testing: Methodologies and Tools"

[3] Stankovic, N.: Patterns and tools for performance testing. In: Electro/information Technology, 2006 IEEE International Conference, pp. 152-157 (2006).

[4] D. Evangelin Geetha T.V. Suresh Kumar K. Rajani Kanth, "Predicting the software performance during feasibility study", IET Softw., 2011, Vol. 5, Iss. 2, pp. 201–215.

[5] Lars Lundberg, Daniel H\&\#228; ggander, and Wolfgang Diestelkamp. 2001. Conflicts and Trade-Offs between Software Performance and Maintainability. In Performance Engineering, State of the Art and Current Trends, Reiner R. Dumke, Claus Rautenstrauch, Andreas Schmietendorf, and Andr\&\#233; Scholz (Eds.). Springer-Verlag, London, UK, 56-67.

[6] Frank Huebner, Kathleen S. Meier-Hellstern, and Paul Reeser. 2001. Performance Testing for IP Services and Systems. In Performance Engineering, State of the Art and Current Trends, Reiner R. Dumke, Claus Rautenstrauch, Andreas Schmietendorf, and Andr\&\#233; Scholz (Eds.). Springer-Verlag, London, UK, 283-299.

[7] C. Murray Woodside. 2000. Software Performance Evaluation by Models. In Performance Evaluation: Origins and Directions, G\&\#252;nter Haring, Christoph Lindemann, and Martin Reiser (Eds.). Springer-Verlag, London, UK, 283-304.

[8] George Din, Ina Schieferdecker, and Razvan Petre. 2008. Performance Test Design Process and Its Implementation Patterns for Multi-services Systems. In Proceedings of the 20th IFIP TC 6/WG 6.1 international conference on Testing of Software and Communicating Systems: 8th International Workshop (TestCom '08 / FATES '08), Kenji Suzuki, Teruo Higashino, Andreas Ulrich, and Toru Hasegawa (Eds.). Springer-Verlag, Berlin, Heidelberg, 135-152.

[9] Donghun Lee1 and Jong-Jin Park, "Software Performance Monitoring Using Aggregated Performance Metrics by Z-Value", G. Lee et al. (Eds.): ICHIT 2012, LNCS 7425, pp. 708–715, 2012.

[10] Lloyd G. Williams and Connie U. Smith. 1995. Information Requirements for Software Performance Engineering. In Proceedings of the 8th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation: Quantitative Evaluation of Computing and Communication Systems (MMB '95), Heinz Beilner and Falko Bause (Eds.). Springer-Verlag, London, UK, 86-101.

[11] Jovica Đurković, Jelica Trninić, Vuk Vuković, "Test Software Functionality, but Test its Performance as Well", Management Information Systems, Vol. 6 (2011), No. 2, pp. 003-007.

[12] Ulrich Herzog, Jerry Rolia, "Performance validation tools for software/hardware systems", Performance Evaluation 45 (2001) 125–146.

[13] Jing Xu, "Rule-based automatic software performance diagnosis and improvement", Performance Evaluation 67 (2010) 585_611.

[14] Hangjung Zo, Derek L. Nazareth, and Hemant K. Jain. 2010. Security and performance in service-oriented applications: Trading off competing objectives. Decis. Support Syst. 50, 1 (December 2010), 336-346.

[15] H. Sarojadevi, "Performance Testing: Methodologies and Tools", Journal of Information Engineering and Applications www.iiste.org ISSN 2224-5758 (print) ISSN 2224-896X (online) Vol 1, No.5, 2011.

[16] C. U. Smith and L. G. Williams. 1993. Software Performance Engineering: A Case Study Including Performance Comparison with Design Alternatives. IEEE Trans. Softw. Eng. 19, 7 (July 1993), 720-741.

[17] Murray Woodside, Greg Franks, and Dorina C. Petriu. 2007. The Future of Software Performance Engineering. In 2007 Future of Software Engineering (FOSE '07). IEEE Computer Society, Washington, DC, USA, 171-187.

[18] Li Yuanyuan, Xiao Peng, DengWu, "The Method To Test Linux Software Performance", 978- 1-4244-6947-5/ 10 20 10 IEEE.

[19] Ibidokun Emmanuel Tope, Pavol Zavarsky, Ron Ruhl, and Dale Lindskog. 2011. Performance Evaluation of Oracle VM Server Virtualization Software 64 Bit Linux Environment. In Proceedings of the 2011 Third International Workshop on Security Measurements and Metrics (METRISEC '11). IEEE Computer Society, Washington, DC, USA, 51-57

[20] Elaine J. Weyuker and Filippos I. Vokolos. 2000. Experience with Performance Testing of Software Systems: Issues, an Approach, and Case Study. IEEE Trans. Softw. Eng. 26, 12 (December 2000

[21] Yan Jin, Antony Tang, Jun Han, and Yan Liu. 2007. Performance Evaluation and Prediction for Legacy Information Systems. In Proceedings of the 29th international conference on Software Engineering (ICSE '07). IEEE Computer Society, Washington, DC, USA, 540-549.

[22] Filippos I. Vokolos and Elaine J. Weyuker. 1998. Performance testing of software systems. In Proceedings of the 1st international workshop on Software and performance (WOSP '98). ACM, New York, NY, USA, 80-87.

[23] Giovanni Denaro, Andrea Polini, and Wolfgang Emmerich. 2004. Early performance testing of distributed software applications. SIGSOFT Softw. Eng. Notes 29, 1 (January 2004), 94-103.

[24] Vittorio Cortellessa, Pierluigi Pierini, and Daniele Rossi. 2007. Integrating Software Models and Platform Models for Performance Analysis. IEEE Trans. Softw. Eng. 33, 6 (June 2007), 385-401.

[25] Nidhi Tiwari and T. S. Harish Kashyap. 2011. Performance analysis of a website in production using modeling and simulation: case study. In Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques (SIMUTools '11). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 499-505.

[26] Harry Sneed, "Testing Software for Internet applications", Volume 1, Issue 1.

[27] Emanuel R. Baker. 2001. Which Way, SQA?. IEEE Softw. 18, 1 (January 2001), 16-18.

[28] Norman F. Schneidewind. 2002. Body of Knowledge for Software Quality Measurement. Computer 35, 2 (February 2002), 77-83

[29] Jeffrey Voas and William W. Agresti. 2004. Software Quality from a Behavioral Perspective. IT Professional 6, 4 (July 2004), 46-50.

[30] John L. Anderson, Jr., "How To Produce Better Quality Test Software", IEEE Instrumentation & Measurement Magazine August 2005.

[31] Wei-Tek Tsai, Yinong Chen, and Ray Paul. 2005. Specification-Based Verification and Validation of Web Services and Service-Oriented Operating Systems. In Proceedings of the 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS '05). IEEE Computer Society, Washington, DC, USA, 139-147.

[32] Antonia Bertolino. 2007. Software Testing Research: Achievements, Challenges, Dreams. In 2007 Future of Software Engineering (FOSE '07). IEEE Computer Society, Washington, DC, USA, 85-103.

[33] Matt Chu, Christian Murphy, and Gail Kaiser. 2008. Distributed In Vivo Testing of Software Applications. In Proceedings of the 2008 International Conference on Software Testing, Verification, and Validation (ICST '08). IEEE Computer Society, Washington, DC, USA, 509-512.

[34] Christof Lutteroth, Gerald Weber, "Modeling a Realistic Workload for Performance Testing", 12th International IEEE Enterprise Distributed Object Computing Conference.

[35] Alberto Avritzer, Joe Kondek, Danielle Liu, and Elaine J. Weyuker. 2002. Software performance testing based on workload characterization. In Proceedings of the 3rd international workshop on Software and performance (WOSP '02). ACM, New York, NY, USA, 17-24.

[36] Vahid Garousi, Lionel C. Briand, and Yvan Labiche. 2006. Traffic-aware stress testing of distributed systems based on UML models. In Proceedings of the 28th international conference on Software engineering (ICSE '06). ACM, New York, NY, USA, 391-400.

[37] Reis, S.; Metzger, A.; Pohl, K.: A Reuse technique for Performance Testing of Software Product Lines. In: Proc. of the Intl. Workshop on Software

Product Line Testing, Mannheim University of Applied Sciences, Report No. 003.06, (2006) 5-10

[38] J. Križani_, A. Grguri_, M. Mošmondor, P. Lazarevski, "Load testing and performance monitoring tools in use with AJAX based web Applications", MIPRO 2010, May 24-28, 2010, Opatija, Croatia.

[39] Yanyan Lu, Haiyan Wu, Yingxue Wang, "Web Application Performance Analysis Based on Comprehensive Load Testing", ICWMMN2006 Proceedings.

[40] S. Abu-Nimeh, S. Nair, and M. Marchetti. 2006. Avoiding Denial of Service via Stress Testing. In Proceedings of the IEEE International Conference on Computer Systems and Applications (AICCSA '06). IEEE Computer Society, Washington, DC, USA, 300-307.

[41] Osama Hamed1 and Nedal Kafri, "Performance Testing for Web Based Application Architectures (.NET vs. Java EE)" ©2009 IEEE.

[42] Chien-Hung Liu, David C. Kung, Pei Hsia, and Chih-Tung Hsu. 2000. Structural Testing of Web Applications. In Proceedings of the 11th International Symposium on Software Reliability Engineering (ISSRE '00). IEEE Computer Society, Washington, DC, USA, 84-.

[43] Wing Lam. 2001. Testing E-Commerce Systems: A Practical Guide. IT Professional 3, 2 (March 2001), 19-27.

[44] Harrine Freeman, "Software Testing", IEEE Instrumentation & Measurement Magazine September 2002.

[45] James B. Michael, Bernard J. Bossuyt, and Byron B. Snyder. 2002. Metrics for Measuring the Effectiveness of Software-Testing Tools. In Proceedings of the 13th International Symposium on Software Reliability Engineering (ISSRE '02). IEEE Computer Society, Washington, DC, USA, 117-.

[46] Juichi Takahashi and Yoshiaki Kakuda. 2002. Effective Automated Testing: A Solution of Graphical Object Verification. In Proceedings of the 11th Asian Test Symposium (ATS '02). IEEE Computer Society, Washington, DC, USA, 284-

[47] Sebastian Elbaum, Srikanth Karre, and Gregg Rothermel. 2003. Improving web application testing with user session data. In Proceedings of the 25th International Conference on Software Engineering (ICSE '03). IEEE Computer Society, Washington, DC, USA, 49-59.

[48] Fevzi Belli and Christof J. Budnik. 2005. Towards self-testing of component-based software. In Proceedings of the 29th annual international conference on Computer software and applications conference (COMPSAC-W'05). IEEE Computer Society, Washington, DC, USA, 205-210.

[49] Dirk Draheim, John Grundy, John Hosking, "Realistic Load Testing of Web Applications", Proceedings of the Conference on Software Maintenance and Reengineering (CSMR'06).

[50] Gerardo Canfora and Massimiliano Di Penta. 2006. Testing Services and Service-Centric Systems: Challenges and Opportunities. IT Professional 8, 2 (March 2006), 10-17.

[51] Hyunsook Do and Gregg Rothermel. 2006. On the Use of Mutation Faults in Empirical Assessments of Test Case Prioritization Techniques. IEEE Trans. Softw. Eng. 32, 9 (September 2006), 733-752.

[52] Diwakar Krishnamurthy, Jerome A. Rolia, and Shikharesh Majumdar. 2006. A Synthetic Workload Generation Technique for Stress Testing Session-Based Systems. IEEE Trans. Softw. Eng. 32, 11 (November 2006), 868-882.

[53] Harry M. Sneed and Shihong Huang. 2006. WSDLTest - A Tool for Testing Web Services. In Proceedings of the Eighth IEEE International Symposium on Web Site Evolution (WSE '06). IEEE Computer Society, Washington, DC, USA, 14-21

[54] Wen-Li Dong and Hang YU. 2006. Web Service Testing Method Based on Fault-coverage. In Proceedings of the 10th IEEE on International Enterprise Distributed Object Computing Conference Workshops (EDOCW '06). IEEE Computer Society, Washington, DC, USA, 43-.

[55] Shiping Chen, David Moreland, Surya Nepal, and John Zic. 2008. Yet another Performance Testing Framework. In Proceedings of the 19th Australian Conference on Software Engineering (ASWEC '08). IEEE Computer Society, Washington, DC, USA, 170-179.

[56] Ana Cavalli, Stephane Maag, and Gerardo Morales. 2007. Regression and Performance Testing of an e-Learning Web Application: dotLRN. In Proceedings of the 2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System (SITIS '07). IEEE Computer Society, Washington, DC, USA, 369-376.

[57] Xingen Wang, Bo Zhou, and Wei Li. 2010. Model Based Load Testing of Web Applications. In Proceedings of the International Symposium on Parallel and Distributed Processing with Applications (ISPA '10). IEEE Computer Society, Washington, DC, USA, 483-490.

[58] Subhasri Duttagupta, Rajesh Mansharamani "Extrapolation Tool for Load Testing Results. ", 27-30 June 2011 ,69 - 76 ,978-1-61782-309-1 ,12181329 , Performance Evaluation of Computer & Telecommunication Systems (SPECTS), 2011

[59] Subhasri Duttagupta and Manoj Nambiar. 2011. Performance Extrapolation for Load Testing Results of Mixture of Applications. In Proceedings of the 2011 UKSim 5th European Symposium on Computer Modeling and Simulation (EMS '11). IEEE Computer Society, Washington, DC, USA, 424-429.

[60] Li Zhang, Yinghui Chen, Fan Tang, and Xiong Ao. 2011. Design and implementation of cloud-based performance testing system for web services. In Proceedings of the 2011 6th International ICST Conference on Communications and Networking in China (CHINACOM '11). IEEE Computer Society, Washington, DC, USA, 875-880.

[61] Hu Maogui, Wang Jinfeng "Application of Automated Testing Tool in GIS Modeling∗" World Congress on Software Engineering © 2009 IEEE

[62] Tara Astigarraga1, Eli M. Dow2, Christina Lara1, Richard Prewitt2, Maria R. Ward3. "The Emerging Role of Software Testing in Curricula" 978-1-4244-6042-7/10 ©2010 IEEE.

[63] Matthias Beyer, Win fried Dulz, Kai-Steffen Hielscher "Performance Issues in Statistical Testing". Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB), 2006 13th GI/ITG Conference ,27-29 ,March 2006 ,1 - 17,

[64] Sheng Huang, Zhong Jie Li, Ying Liu, Jun Zhu, Yang Hua Xiao, and Wei Wang. 2011. Regression Testing as a Service. In Proceedings of the 2011 IEEE International Conference on Web Services (ICWS '11). IEEE Computer Society, Washington, DC, USA, 484-491.

[65] XiangFeng Meng. "Analysis of Software Automation Test Protocol" 2011 International Conference on Electronic & Mechanical Engineering and Information Technology, (Volume: 8), 12-14 Aug. 2011 ,4138 – 4141,978-1-61284-087-1 ,Harbin, Heilongjiang .

[66] Xiaoying Bai_ y, Muyang Li_, Bin Chen_, Wei-Tek Tsai_ z, Jerry Gao_ x "Cloud Testing Tools "Proceedings of The 6th IEEE International Symposium on Service Oriented System Engineering (SOSE 2011), 12-14 Dec. 2011 ,1 - 12 ,78-1-4673-0410-8 ,Irvine, CA.

[67] Arabi Keshk, Amal Ibrahim "Ensuring the Quality Testing of Web Using a New Methodology "2007 IEEE International Symposium on Signal Processing and Information Technology, 15-18 Dec. 2007 ,1071 - 1076 ,978-1-4244-1835-0 ,Giza

[68] Torkey, F.A. ; Menoufia Univ., Shebeen El-Kome ; Keshk, A. ; Hamza, T. ; Ibrahim, A. "A New Methodology for Web Testing"IEEE CCC Code: 1-4244-1430-X/07/ ©2007 IEEE.

[69] Heejin Kim, Byoungju Choi, and W. Eric Wong. 2009. Performance Testing of Mobile Applications at the Unit Test Level. In Proceedings of the 2009 Third IEEE International Conference on Secure Software Integration and Reliability Improvement (SSIRI '09). IEEE Computer Society, Washington, DC, USA, 171-180.

[70] Qinglin Wu, Yan Wang; "Performance Testing and Optimization of J2EE-based Web Applications" 2010 Second International Workshop on Education Technology and Computer Science, 6-7 March 2010 ,681 - 683 ,978-1-4244-6389-3,Wuhan

[71] Guo Wenming, Fu Xiangling, Feng Jianmei"A Data-driven Software Testing Tools Integration System "978-1-4244-5392-4/10/ ©2010 IEEE, 10-12 Dec. 2010 ,1 - 4 ,978-1-4244-5392-4 ,Wuhan

[72] Philip Robinson and Carmelo Ragusa "Taxonomy and Requirements Rationalization for Infrastructure in Cloud-based Software Testing "2011 Third IEEE International Conference on Coud Computing echnology and Science.,2011 ,454 - 461 ,978-1-4673-0090-2,Athens

[73] Wenji Cao, Jianping Tang, Yanlin Shao, and Qingyun Wang. 2009. The Web Performance Testing Algorithm of the Socket-group-based Strategy. In Proceedings of the 2009 WRI World Congress on Software Engineering - Volume 04 (WCSE '09), Vol. 4. IEEE Computer Society, Washington, DC, USA, 246-250.

[74] Pu yunming, Xu mingna;" Load Testing for Web Applications "The 1st International Conference on Information Science and Engineering (ICISE2009), 26-28 Dec. 2009 ,2954 - 2957 ,978-1-4244-4909-5

[75] Marco A. S. Netto, Suzane Menon, Hugo V. Vieira, Leandro T. Costa, Flavio M. de Oliveira, Rodrigo Saad, and Avelino Zorzo. 2011. Evaluating Load Generation in Virtualized Environments for Software Performance Testing. In Proceedings of the 2011 IEEE International

Symposium on Parallel and Distributed Processing Workshops and PhD Forum (IPDPSW '11). IEEE Computer Society, Washington, DC, USA, 993-1000.

[76] Michael Gopshtein and Dror G. Feitelson. 2011. Trading off quality for throughput using content adaptation in web servers. In Proceedings of the 4th Annual International Conference on Systems and Storage (SYSTOR '11). ACM, New York, NY, USA, , Article 6 , 14 pages.

[77] Jayshankar Sankarasetty, Kevin Mobley, Libby Foster, Tad Hammer, and Terri Calderone. 2007. Software performance in the real world: personal lessons from the performance trauma team. In Proceedings of the 6th international workshop on Software and performance (WOSP '07). ACM, New York, NY, USA, 201-208.

[78] Daniel A. Menascé. 2002. Load Testing of Web Sites. IEEE Internet Computing 6, 4 (July 2002), 70-74.

[79] Andreas Leitner, Ilinca Ciupa, Bertrand Meyer, and Mark Howard. 2007. Reconciling Manual and Automated Testing: The Auto Test Experience. In Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS '07). IEEE Computer Society, Washington, DC, USA, 261a-.

[80] Shuang Wang and Jeff Offutt "Comparison of Unit-Level Automated Test Generation Tools "IEEE International Conference on Software Testing Verification and Validation Workshops.

[81] Nawwar Kabbani, Scott Tilley , Lewis Pearson "Towards an Evaluation Framework for SOA Security Testing Tools "SysCon 2010 – IEEE International Systems Conference San Diego, CA, April 5–10, 2010.

[82] Gordon Fraser, Andrea Arcuri;" Whole Test Suite Generation "IEEE TRANSACTIONS ON SOFTWARE ENGINEERING 2012.

[83] Khaled M. Mustafa, Rafa E. Al-Qutaish, Mohammad I. Muhairat "Classification of Software Testing Tools Based on the Software Testing Methods "2009 Second International Conference on Computer and Electrical Engineering 978-0-7695-3925-6/09 2009 IEEE.

[84] Chang-Sik Cho, Dong-Chun Lee, Kang-Min Sohn, Ji-Hoon Kang "Scenario-Based Approach for Black box Load Testing of Online Game Servers "2010 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery.

[85] Ravi MuWramala, Ratnakar Pedagani, and Himanshu Keskar "TEST DATA MANAGEMENT SYSTEM FOR AVIATION SOFTWARE "0-7803-8539-X/04/ 2004 IEEE.

[86] Gwang-hun Kim, Hui-choun Moon, Gi-Pyeung Song, Seok-Kyu Shin "Software performance testing scheme using Virtualization technology "978-1-4244-5130-2/09/ © 2009 IEEE.

[87] Zhang Xinfeng, Shen Yong, SongGe "Stress testing on car remote monitoring system "978-1-4244-8039-5/11/ ©2011 IEEE.

[88] Pingyu Zhang, Sebastian Elbaum, and Matthew B. Dwyer "Automatic Generation of Load Tests" 978-1-4577-1639-3/11/ 2011

[89] Chih-Wei Ho and Laurie Williams. 2007. Developing software performance with the performance refinement and evolution model. In Proceedings of the 6th international workshop on Software and performance (WOSP '07). ACM, New York, NY, USA, 133-136.

[90] Jimoh, R.g & Abikoye, O.C. ;"SOFTWARE MPERFORMANCE EVALUATION ALGORITHM EXPERIMENT FOR IN-HOUSE SOFTWARE USING INTER-FAILURE DATA" International journal of engineering and management sciences 2012

[91] http://www8.hp.com/us/en/software-solutions/software.html?compURI=1175451#.UMRVI-T7LOc last access =15-12-2012

[92] http://www.testpro-int.com/wp content/pdf/hploadrunner.pdf last access= 15-12-2012

[93] http://www.scribd.com/doc/26462234/LoadRunner-9-5-Tutorial last access = 15-12-2012

[94] http://grinder.sourceforge.net/g3/features.html#Standards last access = 16-12-2012

[95] http://grinder.sourceforge.net/g3/tutorial-perks.html last access = 16-12-2012

[96] http://loadstorm.com/files/grinder-manual.pdf last access = 16-12-2012

[97] http://www.jk-itberatung.de/grinder/ GrinderInTheCloud.pdf last access = 16-12-2012

[98] http://grinder.sourceforge.net/links.pdf last access = 16-12-2012

[99] http://www.scribd.com/doc/48775592/2/Apache-JMeter-Features-Overview last access =16-12-2012

[100] http://www.scribd.com/doc/7499267/Load-Testing-With-JMeter last access = 16-12-2012

[101] http://jmeter.apache.org/usermanual/component_reference.html#Monitor_Results last access = 15-12-2012

[102] http://en.wikipedia.org/wiki/Apache_JMeter last access = 15-12-2012

[103] http://www.packtpub.com/beginning-apache-jmeter/book last access = 15-12-2012

[104] http://apache-jmeter.blogspot.com/ last access = 15-12-2012

[105] http://documentation.microfocus.com/help/index.jsp?topic=%2Fcom.microfocus.silkperformer.doc%2FSILKPERF-5F28C3C0-BDWLT-BENEFITS-CON.html last access = 15-12-2012

[106] http://www.powertest.com/software-performance-testing-micro-focus-silkperformer.html last access = 15-12-2012

[107] http://techpubs.borland.com/silk_gauntlet/SilkPerformer/2010/EN/NetExplorerUserGuide.pdf last access = 15-12-2012

[108] http://supportline.microfocus.com/documentation/books/ASQ/SilkPerformer/2010R2/EN/SAPTutorial.pdf last access = 15-12-2012

[109] http://qageek.files.wordpress.com/2007/05/lr_monitor.pdf last access = 15-12-2012

[110] http://www.whiteboxqa.com/StudentMaterial/Guides/LoadRunner%20Controller%20Guide%208.1.pdf last access = 15-12-2012

[111] http://documentmythink.googlecode.com/files/loadrunner_tutorial.pdf last access = 15-12-2012

[112] http://publib.boulder.ibm.com/infocenter/rpthelp/v7r0m0/index.jsp?topic=/com.ibm.rational.test.lt.doc/topics/c_productdesc.html last access = 15-12-2012

[113] http://en.wikipedia.org/wiki/IBM_Rational_Performance_Tester last access = 15-12-2012

[114] http://rational.royalcyber.com/what-we-do/software-quality-assurance/rational-performance-tester/ last access = 15-12-2012

[115] https://www.ibm.com/developerworks/library/r-hellorpt/r-hellorpt-pdf.pdf last access = 15-12-2012

[116] https://www.ibm.com/developerworks/library/r-hellorpt/section7.html last access = 15-12-2012

[117] http://www.opensta.org/ last access = 15-12-2012

[118] http://www.iperformax.com/downloads/http___opensta.org_docs_.pdf last access = 15-12-2012.

[119] http://www.scribd.com/doc/59890116/OpenSTA-Tutorial last access = 15-12-2012

[120] http://mentora.us/documents/OpenSTAStarEast2007.pdf last access = 15-12-2012

[121] Mohamad S. Bayan and Joao W. Cangussu. 2006. Automatic Stress and Load Testing for Embedded Systems. In Proceedings of the 30th Annual International Computer Software and Applications Conference - Volume 02 (COMPSAC '06), Vol. 2. IEEE Computer Society, Washington, DC, USA, 229-233.

[122] Shiping Chen, David Moreland, Surya Nepal, and John Zic. 2008. Yet Another Performance Testing Framework. In Proceedings of the 19th Australian Conference on Software Engineering (ASWEC '08). IEEE Computer Society, Washgton, DC, USA, 170-179

[123] Tiantian Gao; Yujia Ge; Gongxin Wu; Jinlong Ni, "A Reactivity-based Framework of Automated Performance Testing for Web Applications," Distributed Computing and Applications to Business Engineering and Science (DCABES), 2010 Ninth International Symposium on , vol., no., pp.593,597, 10-12 Aug. 2010.

[124] R. Thirumalai Selvi, , Sudha, Dr. N. V. Balasubramanian "Performance Analysis of Proprietary andNon-Proprietary Software" Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol IIMECS 2008, 19-21 March, 2008, Hong Kong.

[125] Amanpreet Kaur "Outsourcing Software quality" Norwegian University of Science and Technology Department of Computer and Information Science, 2013