Physical Education, Health and Social Sciences
https://journal-of-social-education.org
E-ISSN: 2958-5996
P-ISSN: 2958-5988

Automated Load Testing Tools for Performance Monitoring

Komal Farooq[1] , *Muhammad Hammad U Salam[2], Fouzia Sher Akbar[1], Zohaib Shafique[3]

[1] Department of Computer Science and Information Technology, Women University of AJK, Bagh
[2] Department of Computer Science and Information Technology, University of Kotli AJK,Kotli
[3] School of Information Technology & Engineering, Melbourne Institute of Technology
Corresponding Author: *Muhammad Hammad Salam(hammad.salam@uokajk.edu.pk)

DOI: https://doi.org/10.63163/jpehss.v4i1.1059

Abstract

Automated load testing is very important in modern age; because the manual testing need extensive labor and time; while the automated testing is done through some automated tools. There are verities of automated load testing tools are available in market and everyone claim that the performance is better than the other automated load testing tools. The common users and software industries have no idea which tool performance is better; in market so many open source tools are available. The main problem how to guide the common user and the software industries, which tool performance is better, here we have taken few automated load testing tools and compared the performance of these tools. In this article we have compared just four automated load testing tools, the comparison was based on three main things (throughput and response time and total time) and these all tools are web based tools. The results suggest that the JMeter performance is better than other three automated load testing tool.

Keywords: Automated load testing, Load testing tools, Performance monitoring, Response time, Throughput.

Introduction

In the field of automated load testing tools, we have compared the performance of automated load testing tools; we have given the same load to the all four tools at same time and check the performance. The mainly objective of automated load testing, is measurement the performance, which satisfied the load requirements of a system (Hung Q et al., 2001). The basic methodology of automated load testing tools is judgment of several steps which performing load on web and it is performed under number of users (J.D. Meier et al., 2007). Quality is conformance to requirements and prevention of defects. The quality of software is measure in the form of verification and validation. Software testing insure the quality of software, manual testing is labor extensive, in modern age trend of automated testing is increasing day by day. Many researchers try to explain and compared the automated load testing tools, but they did not compare the performance of tools with experiments, we have compared the automated tools through experimental and result suggest that JMeter perform well as compare to other. The comparison of automated load testing tools mainly impacts on the software industry and future researchers. Specially, it has a great impact on the software industries because the software industry makes automated load testing tools for testing, if they experimentally compare the tools. They can easily select tool, which is the best for their industry and how can they make a

new tool for future generation and which tool performance is better. For the development of new tools, which parameter is needed? Sadiq et al authors compared common refer automated performance tools (Sadiq et al., 2016). Križanić J et al in this article authors compared and analyzed the web load testing tools and measured the performance of different tools (Križanić J. et al., 2010). Using the testing procedures the quality of a software product can be checked that product which undergone for checking. The main thing in the testing it is an ongoing procedure, which is related with some process for producing a quality product. So according IEEE computer society the testing is a method of assessing system component and a system by manual or through automated tools and it mean to verify the user requirements or meet the user requirements (Tsung, 2016). In this we can checked the software products and their working position. Through software testing we can identified the defects which may a software product have. The software testing is main part of software development life cycle, testing have main role to judge the product. Software testing can be done by two ways one is automated and second, in modern days the automated testing is prefer because it not labor extensive and time consuming, through automated tools we can repeat the scripts many time (Tsung. 2016). The following advantages of automated testing high execution speed, many time repetitions and less labor need. The manual testing need more time and it have more chance of errors. The automated testing tools compared by different authors in these articles (Iqbal et al., 2016; Y. Cai et al., 2004). The rest of this paper is organized as follows; section second gives a brief review of performance testing and load testing and load testing tools and section three gives results and section four give features comparison of tools, while conclusion is drawn in section five.

Performance Monitoring

For collecting and analyzing the data, performance monitoring is an ongoing process to compare how well a program; project or policy is being implemented against the expectations. This is the one of way for performance testing, which provides an opportunity to test application differently by it functionalities. In Figure 1 four different types of process are define, when test cases want to generated, it choose any one and executing tests again those. This type of testing is not only to solve the problem and application design/construction in the early stages, but also makes strong applications simple. Same with performance testing plan a new application and start the development-related activities, as well as performance test planning and infrastructure also started and right after the first version (which may not be released), the application to test its performance. Most of these tests are tests designed to proactively search performance limitations to find bug and bottlenecks. It was here when the test-driven development, follow a risk-based approach and application of critical test are more likely and highly impacting areas. Here in Figure 1, some main performance tests that are expected to be conducted in this way.

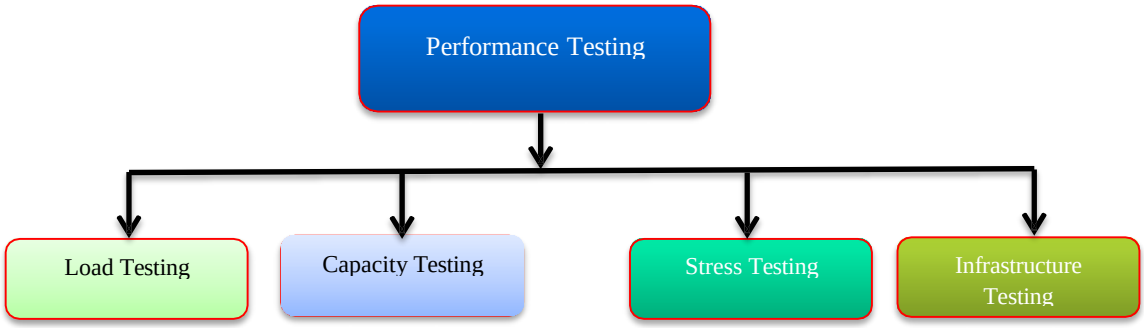

Figure 1: Performance Testing (Test case)

The performances of all monitoring tools which are based on the concepts have similar to the load

testing tools that exists with two similar components: the controller, which is known as the monitoring server, and it is due to the work and monitoring the proxy server model, as it is shown in Figure 2, (Vibhu Saujanya et al., 2006; H. Sarojadevi 2016; Stankovic et al., 2006; D. Evangelin Geetha et al., 2001).

$$N=AR*RT \quad (1)$$

Number of Items in the System = Arrival Rate X Response Time (2)

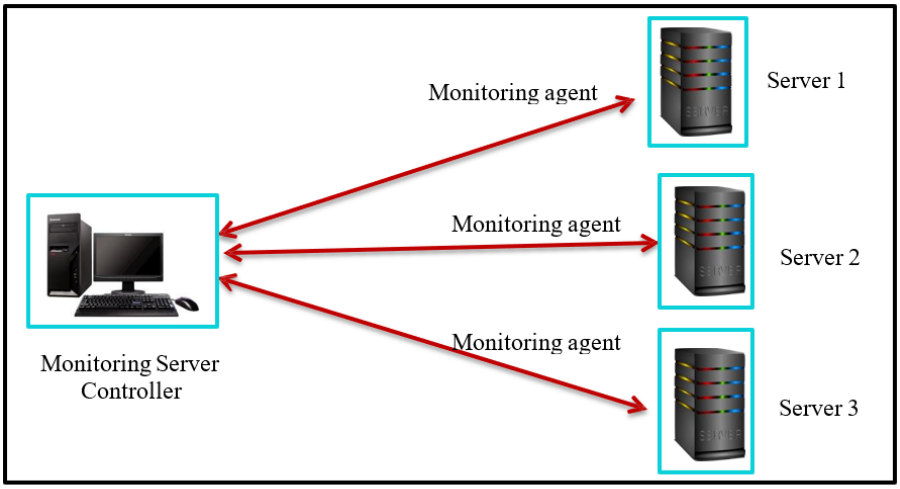

Figure 2: Performance

Monitoring Tools Concept

Load Testing

Figure 3 shows that the measure QoS of sites performance, which is based on actual customer behavior. When customers access your site, a script recorder uses their requests to create interaction scripts. The load generator then replicates the script that may be modified by the test parameters to the site (Menascé D A. et al., 2002).

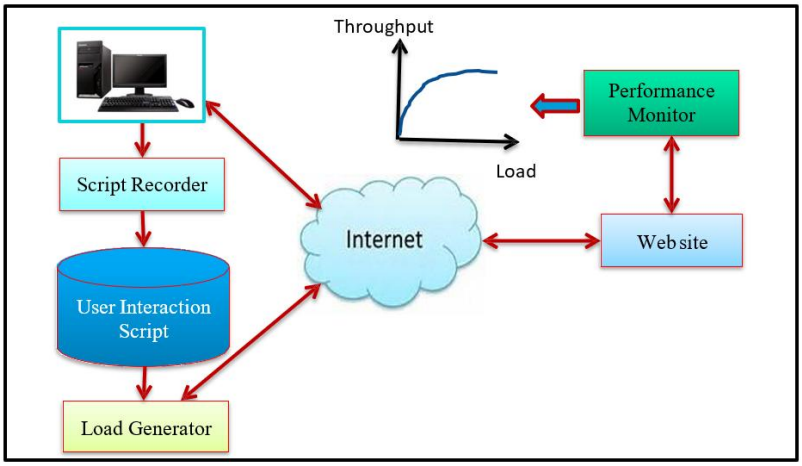

Figure 3: Load Testing

Figure 3, the load test process the script recorder creates a user interaction script, which is based on actual request. In this process the load generator, generate the load then sends the actual load to the site based on the script and test parameters and the monitor its performance. The relationship between performance and load testing is, so we can use automated load testing tools to calculate web sites load in terms of performance. We can increase the total number of

virtual users till to reach desired load. In automated load testing for virtual user generated different test case (running different load test) for different values load testing and it is time consuming. Hence we can also find accurate and faster automated load testing through some performance model (like analytic or simulation performance model (SPM)). If we want to find accurate and faster result, for this we must combine the analytic model or SPM. Daniel A et al in load testing used some basic relationships for performance, which increase (scalability analysis) performance of the load testing. Daniel et al is considering a scenario and in this scenario many virtual users and each user try to submit the request to the websites. According to the Daniel law of response time (LRT) become (Menascé D A. et al., 2002).

$$\mathbf{R = (N\ (VU)\ /\ X) - Z} \qquad \mathbf{(3)}$$

$$\mathbf{R = (VU/AT) - ATT} \qquad \mathbf{(4)}$$

N (VU) = Number of users (Virtual users (VU)), X = throughput per second (Average throughput (X) = (AT)) is the average throughput in request per second and Z = think time in second (average think time (ATT)) is average think time in seconds.

Load Testing Tools

The automated load testing tools are subjecting process in computer, such as a network the work load and approaching to its limits of specifications. The first main objective of automated load testing tools maximum work handles by a system without substantial performance degradation. The load testing conducted in two different ways and first one is called longevity testing, which is also called endurance testing, the ability to handle constant work load for longer time. One of the important features of automated load testing is distributed load on multiple machines. Most of automated load testing tools work on the concept which is shown in Figure 4. This Figure has two main components: controller and load generator, the controller user interface console which provides the control over load generators. User interface console (UI), UI console is used for running, setting and controlling the execution of the tests, setting the test involves defining hosts, which will run agents/load generators, specifying number of virtual users which are simulating the real load, scheduling parameters, etc. Agents are usually small applications which communicate with the controller and generate load on the target server according to parameters received from the controller. Agents can be run on multiple hosts in order to generate higher load. Another important aspect of a load testing tool is a mechanism used to generate a test case scenario (D.A. Menascé et al., 2000; P.J. Denning et al., 1978; D.A. Menascé et al., 2002; D.A. Menascé et al., 1999; S. Vinoski et al, 2002). The Figure 4 shows the working of automated load testing tools.

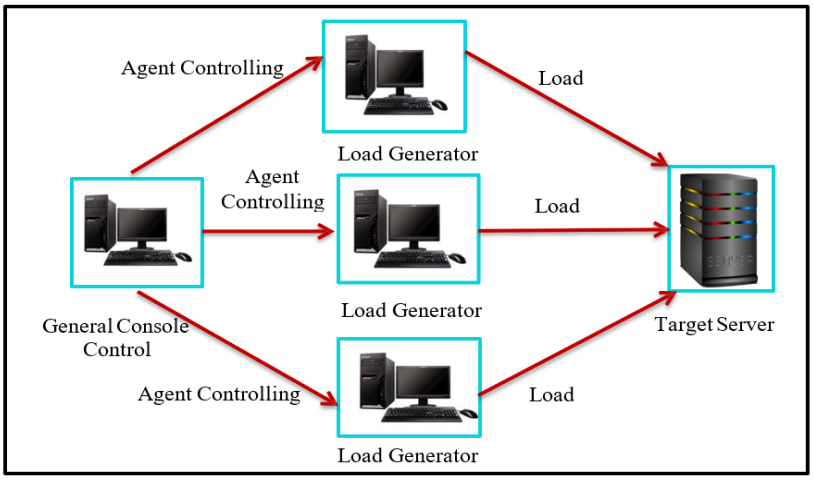

Figure 4: Working of Automated Load Testing Tools

$$VU = AVG / (60/ADV) \quad (5)$$

Where; VU = Virtual users load test number (which we want to figure out) ; AVG = in one hour average number of visitors ;ADV = average duration visitor and 60= sixty minutes in hour . So the mathematically form become

LVU = ((AV) per hour) / (UTR) per hour))(6)

(LVU) = Load Test Virtual User = average visitors in one hour which is divided by user turnover rate per hour.

Grinder

The grinder is open source tool and it is based on the java programming as the medium of instruction. It is automated load testing tool; automatic load test and provide test BSD. This tool was developed by 'Paco Gomez' and maintained by 'Philip Aston'. During the past few years the community had also done much to improve and translation feature and repair. The grinder tool is consist of grinder console, which monitors the results in real time and various grinder agents and the console can be used as the developing test suites or basic interactive development environment for editing. The grinder agents in each headless generator can have multiple staffs to create load. These are the main features of the grinder (such as TCP and proxy server TCP proxy server , log in to the network to the action of grinder test command code, asynchronous testing in with the number of agent increase, staying in this functions that any java API to create or modify a test command code, and flexibility of parametric, including the ability to immediately establish test information and it will be able to use an external source such as (database and file , full access to test the results of Post-processing and assertion support for correlation, content verification and Multi protocols (Križanić J et al., 2010).

Gatling

Gatling is a free and open source test tool for automated performance testing. It is developed and maintain by 'stéphane landelle'. The Gatling has a graphics interface base for testing. However test development can be found in the Domain Specific Language (DSL), easily read or write. The Gatling has the following characteristics as "HTTP recorder, a field of self-Specific Language, DSL developed test according to the scala, use non-blocking asynchronous produces high load, full support for HTTP Protocol , JDBC, and load the JMS test with the test data-driven, powerful, flexible, validation and affirmation, integrated information system and the

report of the "Load" uses asynchronous non-blocking methods to generate higher loads, full support for HTTP (S) protocols, and JDBC and JMS workloads testing, data-driven test input, powerful and flexible authentication and assertion systems, and integrated information load reporting.

Tsung

The first name of Tsung was ‘ IDX tsunami’ , it is not a java base tool , it is open source performance testing tool and Tsung base on erlang and it must need to be install on (Ubuntu Debian, it is just as easy to obtain apt Install erlang). Tsung was developed by ‘Nicolas Niclausse’ in 2001 and initially implemented for distributed load-test solution, Jabber (XMPP) supports more protocols in a few months and in 2003, now a days Tsung is able to perform Http protocol load testing, it is full featured performance testing tool that supports modern protocols web sockets, authentication systems and the database. The following Tsung main features "inherently distributed design, multi-threaded Erlang architecture on the mid-range developer machine simulation of thousands of virtual users, support for multiple protocols, support for HTTP and Postgres test logger, load generator and the metrics of the operating system of the application under test can be gathered through a variety of protocols, vigorous scenarios, and blending behavior. The Tsung is able to Integrate, flexible and load condition, which allows you to define and incorporate any number of load mode test. Tsung is data-driven test from external data sources, easy-to-read embedded load forms and collected load and display the load. Tsung did not provide a graphical user interface for testing development or implementation, so you must use shell script that is: Tsung recorder is a bash script records in a utility, to capture and postgres HTTP requests and establish Tsung profile from their main Tsung, bash control commands to starting and stopping and debugging and viewing status test (Tsung, 2016).

Apache JMeter

In this analysis Apache JMeter is used as desktop application and it is user friendly GUI , which provided friendly environment for testing, in JMeter the development and debugging is so easier and the first version of Apache JMeter was available from March 9, 2001 and since from 2001 to 2017 JMeter is commonly adopted. Now JMeter is a most popular and commonly used open source tool, and alternatives such like tools Load Runner and Silk Performer and other special solutions. The JMeter has a modular structure and it has core through the plug in expansion. This means that all implemented function and protocols are plugging, which was developed by Apache foundation. Apache JMeter can be implemented in a distributed manner and can run on any java based operating system (O/S), so when we need a higher load, through single machine we can created load. This means that the primary JMeter machine controls multiple remote hosts, and it supports multiprotocol and the out of box protocols are (POP3 FTP, SMTP, HTTP, LDAP, HTTP, JMS, SOAP, TCP, JBDC).

Post-processing provides advanced settings, disassembly parameterization and correlation functions, various assertions for defining standards, numerous in-built and exterior hearers for visualization and analysis of performance examination outcomes, as well as main build and nonstop incorporation system and JMeter performance testing is a chunk of software development life cycle (Križanić J et al., 2010).

Results and Discussion

In result section here we consider a case study of automated load testing tools. We have taken four automated load testing tools for performance mentoring. In this case we are checking the performance in term of testing time, response time and throughput, for comparison,

we have use a simple HTTP protocol and GET request from 20 threads with 100,000 iterations and each tool will send requests as fast as it can. Here we are compared the load results of these tools with the following metrics.

- The (Average Response (AR) Time (T)) in terms of mints per second = ((ART)/ (m/s)).
- The (Average Throughput(ATP)) total requests in one second = ((ATP) / s)
- The total(Test Execution Time(TET)) per mints = ((TTET) / m)

In the first, let's look at the average response and total test execution time and then average throughput on requests and finally total test execution time per minutes

Average Response Time

In the Figure 5, JMeter has the lowest average response time and Grinder is the highest average response time. It means JMeter is more efficient as compared with other testing tools. Tsung is also better than Gatling and Grinder. In first case JMeter response with in ten second and Tsung take little more time to response, its response was ten and half second and Gatling response little more late thirteen seconds and Grinder response is so late, thirty one second. It means the response time of JMeter is better than other tools. In this case we have consider the average response time per second, how many requested are response in one second and on they-axis it shows the number of response request per second and on x-axis it shows the name of tools.

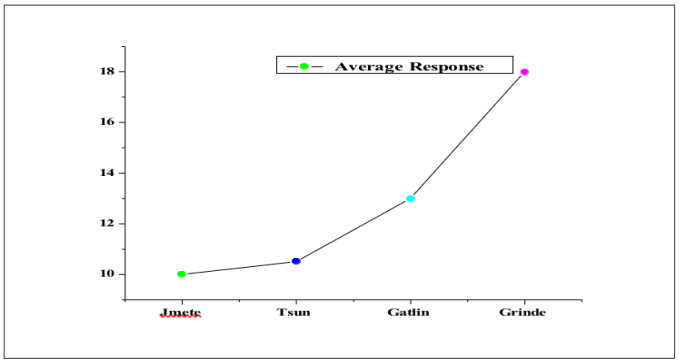

Figure 5: Average Response Time

Average Throughput

In this Figure 6 the fastest response time with highest average throughput is JMeter. The Tsung throughput is also near to JMeter but lower then JMeter and Grinder has the slowest times with lowest average throughput. The average throughput in one second is of JMeter is nineteen hundred , Tsung also same throughput in one second, Gatling is thirteen hundred in one second and Grinder throughput in one second is one thousand. So the performance in term of throughput is better of Jmeter then other tools. In this case we have considered the throughput, the total number of request per second and on the y-axis it shows the number of response per second and on x-axis it shows the name of tools.

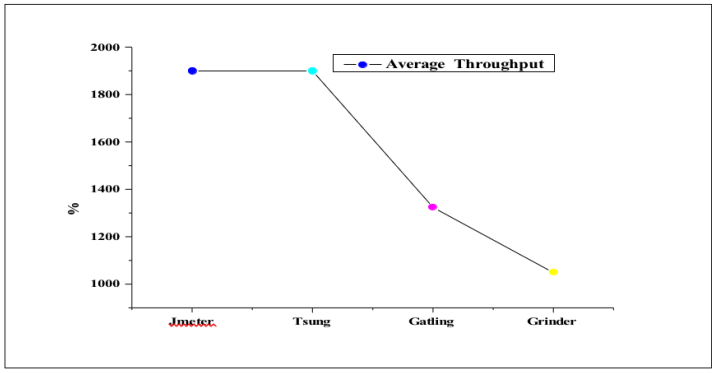

Figure 6: Average Throughput

Total Testing Time

As shown in the Figure 7, JMeter has the lowest total testing time and Grinder is the highest total testing time. It means JMeter is more efficient as compared with other testing tools. Tsung is also better then Gatling and Grinder. The JMeter in executed test in less time, on same load JMeter execute it in seventeen mints, Tsung execute in nineteen mints , Gatling execute in twenty five mints and Grinder execute in thirty one mints. So the JMeter performance better and maximum test executer in less time as compared other tools . In this case we have consider the total test case execute in one mints and on the y-axis it shows the number of test cases executed per mints and on x-axis it shows the name of tools.

Discussions

The Apache JMeter is a powerful tool and its perform load testing very well and compelling way, so the Apache JMeter can also recommend for load testing and it is best tool for load testing, which allows so many users in a one time and provided friendly and self-service environment. We can test the performance of any mobile application, API or site just within ten minutes. The combination of JMeter and BlazeMeter is more attractive for the developers.
JMeter can easily create large scale tests for simple scalability. We have compared in lab and given maximum load, JMeter performance is better than other tools. JMeter has the lowest total testing time and other tools testing time is more than JMeter. JMeter has the lowest average response time. The fastest response time with highest average throughput of JMeter. Furthermore, results can easily be share across the distributed teams, in web-based interactive reporting and overawed to the restrictions of the JMeter standalone user interface.

Features Comparison of Tools

In Table 1, shows the comparison of the each automated load testing tools, offered features and their strength and weakness. JMeter full GUI based and other tools are not GUI based. Tsung is only support LINUX and UNIX operating system; other tools support any type of operating system. Tsung and JMeter support XML language for test language. Each tool support different protocol, JMeter generated load report in Plugins, Graph, Embedded tables, XML, CVS and Tsung and Gatling in HTM. Overall the performance of JMeter is better than other three tools.

Features	Grinder	Tsung	Gatling	JMeter
GUI	Any Console	Unix/Linux No	Only Recorder	Full
OS	Any	Linux/Unix	Any	Any
Test recorder	TCP (including HTTP)	Postgres & HTTP	HTTP	HTTP
Extension language	Clojure, Python	Erlang Erlang	Scala	JavaScript, Bean shell, Java, jell
Test language	Clojure, Python	XML	Scala	XML
No of protocol	POP3, SMPT, JMS, LDAP, HTTP, SOAP, JBDC	LDAP, AMQP, We b Socket, XMPP, MYSQL, P ostger, WebDAV, H TTP,	JDBC, JMS, HTTP	IMAP, SMT, POP3, JM S, TCP, LDAP, SOAP, HTTP, FTP.
Load reports	Console	HTML	HTML	Plugins, Graph, Embedded tables, XML, CVS.
Host monitoring	NO	YES	NO	Perf Mon plugin
Limitations of tools	Python knowledge for test development and reports of editing is plain and brief	It supported and tested only Linux Systems.	It does not scale and supports Limited protocols scala based DSL languages knowledge Required Scale	It does not bundled reporting easy to interpret

Table 1: Features Comparison

Conclusion

In this paper, the taxonomy of different testing tool has been presented such as functional, loading, and management test automation. Now testing through automation tool has become the necessity of today era. Because it is not only save time but also save the manual effort required to test the software for error. The selection of tool requires the requirement gathering and to categorize them in a well manner. This will help a lot to evaluate the tool in a best manner. As many automated software testing tools are available in market and are used to test software but we conducted a comparative study of few best tools and suggest tool that is mostly used in market for automation testing. In this article we give the load to different automated load testing tools and check the response time and total time and throughput, finally concluded that the JMeter is better response time and throughput. We have given same load to all four tools and JMeter performance is better than other tools. JMeter support any type of operating system and Tsung support Unix/Linux. JMeter supports full graphical user interface and other tools do not supports the full graphical user interface. In future research can be compared all other tools by experimental and give best tool for the software industries for automated load testing.

References

Nguyen, H. Q. (2001). *Testing applications on the Web: Test planning for Internet-based systems*. John Wiley & Sons.

Meier, J., Farre, C., Bansode, P., Barber, S., & Rea, D. (2007). Performance testing guidance for web applications: patterns & practices. Microsoft press.

Iqbal, M. S., Sidaq, M., Shabbir, M., Rehman, A., Sajad, M., & Khan, T. (2016). Assessment of Inspection Tools with Standard, Management Review, Technical Review, Inspection and Walkthroughs. International Journal of Computer Science and Information Security, 14(5), 622. Gatling - http://gatling.io/.

Sajad, M., Sadiq, M., Naveed, K., & Iqbal, M. S. (2016). Software Project Management: Tools assessment, Comparison and suggestions for future development.

International Journal of Computer Science and Network Security (IJCSNS), 16(1), 31.

Sadiq, M., Iqbal, M. S., Malip, A., & Othman, W. M. (2015). A Survey of Most Common Referred Automated Performance Testing Tools. ARPN Journal of Science and Technology, 5(11), 525-536.

Križanić, J., Grgurić, A., Mošmondor, M., & Lazarevski, P. (2010, May). Load testing and performance monitoring tools in use with AJAX based web applications. In MIPRO, 2010 Proceedings of the 33rd International Convention (pp. 428-434). IEEE.

Iqbal, M. S., Sidaq, M., Shabbir, M., Rehman, A., Sajad, M., & Khan, T. (2016). Assessment of Inspection Tools with Standard, Management Review, Technical Review, Inspection and Walkthroughs. International Journal of Computer Science and Information Security, 14(5), 622. Tsung - http://tsung.erlang-projects.org/.

Benedikt, M., Freire, J., & Godefroid, P. (2002). VeriWeb: Automatically testing dynamic web sites. In In Proceedings of 11th International World Wide Web Conference (WWW'2002.

Cai, Y., Grundy, J., & Hosking, J. (2004, September). Experiences integrating and scaling a performance test bed generator with an open source case tool. In Proceedings of the 19th IEEE international conference on Automated software engineering (pp. 36-45). IEEE Computer Society.

Curran, K., & Duffy, C. (2005). Understanding and reducing web delays. International Journal of Network Management, 15(2), 89-102.

Deng, Y., Frankl, P., & Wang, J. (2004). Testing web database applications. ACM SIGSOFT Software Engineering Notes, 29(5), 1-10.

Draheim, D., Lutteroth, C., & Weber, G. (2005, March). A source code independent reverse engineering tool for dynamic web sites. In Software Maintenance and Reengineering,

2005. CSMR 2005. Ninth European Conference on (pp. 168-177). IEEE. International Federation of Classification Societies. Conference,

Bock, H. H., Jajuga, K., & Sokolowski, A. (2002). Classification, Clustering and Data Analysis: Recent Advances and Applications. Springer.

Elbaum, S., Rothermel, G., Karre, S., & Fisher II, M. (2005). Leveraging user-session data to support web application testing. IEEE Transactions on Software Engineering, 31(3), 187-202.

D. A. Menasc, Mercury Interactive Corporation. Load Testing to Predict Web Performance. Technical Report WP-1079-0604, Mer-cury Interactive Corporation, 2004.

Ricca, F., & Tonella, P. (2001, July). Analysis and testing of web applications. In Proceedings of the 23rd international conference on Software engineering (pp. 25-34). IEEE Computer Society.

Shaw, J. C., Baisden, C. G., & Pryke, W. M. (2002). Performance testing—a case study of a combined Web/telephony system. BT technology journal, 20(3), 76-86.

Sharma, V. S., & Trivedi, K. S. (2007). Quantifying software performance, reliability and security: An architecture-based approach. Journal of Systems and Software, 80(4), 493- 509.

Sarojadevi, H. (2012). Performance testing: methodologies and tools. Journal of Information Engineering, 2(3).

Stankovic, N. (2006, May). Patterns and tools for performance testing. In Electro/information

Technology, 2006 IEEE International Conference on (pp. 152-157). IEEE.

Geetha, D. E., Kumar, T. S., & Kanth, K. R. (2011). Predicting the software performance

during feasibility study. IET software, 5(2), 201-215.

Menasce, D. A., & Almeida, V. A. (2000). Scaling for E-Business: Technologies. Models, Performance, and Capacity Planning. [25]. P.J. Denning and J.P. Buzen, "The Operational Analysis of Queuing Network Models," ACM Computing Surveys, vol. 10, no. 3, Sept. 1978, pp. 225-261.

Daniel A. Menasce, & Almeida, V. A. (2002). Capacity Planning for Web Services: metrics, models, and methods. Prentice Hall PTR.

Menascé, D. A., Almeida, V. A., Fonseca, R., & Mendes, M. A. (1999, November). A methodology for workload characterization of e-commerce sites. In Proceedings of the 1st ACM conference on Electronic commerce (pp. 119-128). ACM.

Vinoski, S. (2002). Web services interaction models. Current practice. IEEE Internet Computing, 6(3), 89-91.

Menascé, D. A. (2002). Load testing of web sites. IEEE Internet Computing, 6(4), 70-74.