SILESIAN UNIVERSITY OF TECHNOLOGY PUBLISHING HOUSE

SCIENTIFIC PAPERS OF SILESIAN UNIVERSITY OF TECHNOLOGY 2021
ORGANIZATION AND MANAGEMENT SERIES NO. 153

A STRATEGY FOR RESEARCHING THE PERFORMANCE OF WEB APPLICATIONS CLIENTS

Hubert ŁOPUSIŃSKI[1*], Mateusz ŁOPUSIŃSKI[2], Ireneusz J. JÓŹWIAK[3], Kacper STAROŚCIAK[4]

[1] Wroclaw University of Science and Technology, Faculty of Computer Science and Management; hubert.lopusinski@gmail.com
[2] Wroclaw University of Science and Technology, Faculty of Computer Science and Management; mateusz.lopusinski2@gmail.com
[3] Wroclaw University of Science and Technology, Faculty of Computer Science and Management; ireneusz.jozwiak@pwr.edu.pl, ORCID 0000-0002-2160-7077
[4] Wroclaw University of Science and Technology, Faculty of Electronics; kacper.starosciak@gmail.com
* Correspondence author

Purpose: The article presents a comparative analysis of the web application clients performance.
Design/methodology/approach: A strategy of comparing the performance of web application clients using the JavaScript programming language was proposed.
Findings: The metrics used to measure the performance of web application clients were presented.
Research limitations/implications: Techniques affecting the optimal work of web application clients, which can be used regardless of the technology applied were described.
Originality/value: Comparison of the performance of frameworks using the JavaScript programming language. The necessary steps during the process of testing the performance of web applications were proposed and measures relevant to the test were listed. The article is dedicated to the wide spectrum of computer system users.

Keywords: web application, performance, research strategy.

Category of the paper: Research paper.

1. Introduction

Poor software performance is a problem that developers have been struggling with from the very beginning of personal computers existence. Initially, the difficulty was in the implementation of programs that had to be run on equipment with much weaker parameters than those used today. The first IBM PCs had 4.77 MHz processors, while today it is about 3-5 GHz, which is a thousand times more. The most visible progress can be noticed in

RAM memory, because in the past PCs had an amount of RAM equal to 16 KB, and today the standard is from 8 to 16 GB, which means it is a million times more memory. The problem of software performance was then focused on the minimum consumption of hardware resources, which were so small, therefore so valuable. It is also an open secret, that people has not been requiring quick software response time because they were not used to new technologies and their capabilities yet.

It should be remembered that technology and its development are supposed to be human-friendly, facilitating and supporting their work, as well as providing entertainment, e.g. through various types of computer games. Modern society is accustomed to well-developed technology, which is why it has high requirements for software functionality and requires fast response time of mobile applications, web applications, desktop applications and websites. This is obvious, because every time we use a computer or smartphone, we always want them to perform our tasks as quickly and as efficiently as possible. In the 21st century, time is a highly valued resource. People try to save their time in every possible aspect of life, hence they would rather use from the quick meals, quick trainings, quick conversations etc. Many people get nervous very quickly when an application or computer system freezes, does not work properly, is not intuitive or it takes a long time to wait for a response from the system.

For the reasons mentioned above, the question of user feelings has become the subject of interest in the area of software development. A new profession is emerging, a designer of user experience employed in many companies. There are many similar applications on the market. Therefore, companies try to attract users and encourage them to use their product. A satisfied user is an additional profit for the application producer, e.g. by advertising or buying a license for the given software. On the other hand, a dissatisfied user will most likely not want to use the application and will be looking for applications with similar functionalities at the competing companies.

The user experience consists of many aspects, including the response time of the application, and thus the software performance. Manufacturers ensure that their software is the fastest and the best on the market, but these are usually mere lies. Software performance is verified by the empirical usage of the program by its clients, but there are many tools that can be used to audit system performance. Having measurements obtained using professional tools and using additional statistical methods and tests, comparison of the performance of various applications can be conducted.

In the article the performance analysis of web applications were presented. Necessary steps during the process of testing the performance of web applications were proposed. Important metrics to conduct performance test were listed. The performance of frameworks using the JavaScript programming language was compared.

2. Technologies used to create software

An important aspect for the manufacturer himself is the choice of technology used to create the software. First of all, it must be a new technology, because technological debt is very difficult and expensive debt for the company itself. Moreover, employees do not want to work using old technologies, they want to develop and use modern solutions. There are a lot of frameworks, libraries and tools on the market that support programmers during software development. Choosing the right tools for software development also affects the quality of the final product. Not every tool used in the software development process is perfect. There are better and worse tools that can be used to achieve the same set of system functionalities, but other features may vary, e.g. performance. The obvious fact is that the quality of the final product also includes experience and solutions used by programmers that are not dependent on technology.

3. Application performance

There are many scientific positions on the subject of researching application performance. Modern websites have much more complicated logic than before, and they must also be able to run using different internet networks at different speeds and on various types of mobile or computer devices. Because of that, testing the performance of web applications has gained in importance and is becoming a common topic of many researches. Jeremy Wagner claims that when considering application performance, we should take into the account: the number of users using the application, the profits of the company using the application and the users of the application itself (Wagner, 2020). As confirmation of his statement several examples were mentioned. The developer of the Pinterest app has been examining a time that it takes to display first page when turning on the app. By improving the user's waiting time for a response by 40%, an increase by 15% in application usage and registration of new users was achieved (Wagner, 2020). COOK increased profits by 7% and the number of pages visited during one visit by 10%, as a result of reducing the average loading time of the first page by 850 milliseconds (Wagner, 2020). Other studies say that 53% of website visits are dropped while waiting for a system response when the page load time is greater than 3 seconds (Wagner, 2020). The above research confirms the fact that the number of users using the application affects the performance of the application.

Regarding the company's profits, examples of two surveyed companies are given. Mobify company achieved an increase in annual revenues of over 380 thousand dollars, by reducing the time it took to load the first page (Wagner, 2020). AutoAnything achieved a 12-13%

increase in sales after halving the first page load time (Wagner, 2020). Based on these examples, it can be concluded that the time to load the first page of the application is greatly reflecting on the number of users using the application, and thus in the income of the application owner. In addition, Jeremy Wagner claims that when examining application performance, we should take into account the diversity of users using the application, i.e. different speeds of the Internet network that users use and the variety of devices owned by users (Wagner, 2020).

The consequences of low performance are also mentioned by Subraya (2006). He claims, that low software performance may result in temporary cessation in usage of the application, or the complete abandonment in use of the system. Moreover, it can also discourage others from accessing this manufacturer's products. As evidence, the results of a study done on 117 web pages, showing the consequences of a long page loading time were presented. This is shown in Table 1 (Subraya, 2006).

Table 1.
Percentage of pending users depending on how long the web page loads (Subraya, 2006)

Page loading time (s)	Percent of users waiting for the result (%)
10	84
15	51
20	26
30	5

4. Application performance tests and their measure

The authors of the article (Ninka, and Proko, 2013) on the subject of testing web application performance as basic tests state: scalability tests, load tests and stress tests. In turn, Sharmila S. and Ramadevi E. (2014) listed the following steps as necessary during the process of testing the performance of web applications:

1. Identifying the test environment.
2. Identifying the performance test.
3. Planning and designing the test.
4. Configuring the test environment.
6. Test implementation.
6. Test execution.
7. Analyze test results and retest if necessary.

In addition, the authors (Sharmila, and Ramadevi, 2014) of the above steps list measures important during the test, such as response time, bandwidth, CPU resource utilization, RAM, network input and output packets, and critical application errors. Another approach proposed by the authors (Sharmila, and Ramadevi, 2014) is to perform performance analysis based on historical data of the running application obtained from events in the application (logs).

In this approach, each event in the application should be saved to a log file on the hard disk or in the database. Particularly important are events appearing on the client-side applications and each user data, e.g. session data. As it comes to events, the authors (Sharmila, and Ramadevi, 2014) mean places where the user clicks on the page, which pages and for how long the user views them and how much time he stays on the page. Such data is sequentially aggregated using tools for analyzing logs, and based on that places where the application is inefficient may be discovered.

Another concept to carry out the performance tests is the method proposed by Kajol Mittal and Rizwan Khan (2018). These authors claim that testing application performance should be a permanent part of the software development process and such tests should be conducted on an ongoing basis during every build and every application deployment (Khan, and Mittal, 2018). In addition, performance testing according to what authors mentioned, can not be based on general measures, but on conditions strictly defined for a given application and specific functionality. For example, the loading time of the login page should not take longer than 5 seconds with 50 users using the application at the same time. They proposed a solution using CI/CD (Continuous Integration/Continuos Deployment), in which such rules would be written in the code. Such a system is designed to automate performance testing and performance analysis in the software lifecycle by integrating with the Jmeter performance testing tool and the Dynatrace performance analysis tool with the Continuos Integration Jenkins platform (Khan, and Mittal, 2018; Cassone et al., 2001).

5. Comparison of frameworks performance

In the research conducted by Mariano C.L. (2017), the author compares the performance of frameworks using the JavaScript programming language. Four frameworks with libraries were compared: BackboneJS, AngularJS, React-noJSX and React-JSX. To perform the tests, the application that runs tests using Node.js was implemented. The test for each framework was launched on three browsers: Chrome, Edge and Mozilla (Mariano, 2017). The application simulates user behavior by performing a sequence of actions (e.g. adding an item to the list, refreshing the list), while measuring the total duration of the test. The results of the tests conducted by the author are shown in Figures 1, 2 and 3, taking into account the type of web browser. The smaller the time, the better the test result. The time shown in Figures 1, 2 and 3 is the average of 25 tests runs (Mariano, 2017).

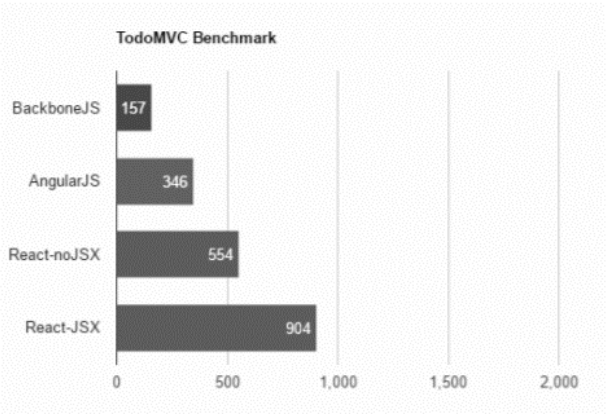

Figure 1. Average test duration for various JavaScript frameworks obtained by Mariano C.L. on a Chrome web browser (Mariano, 2017).

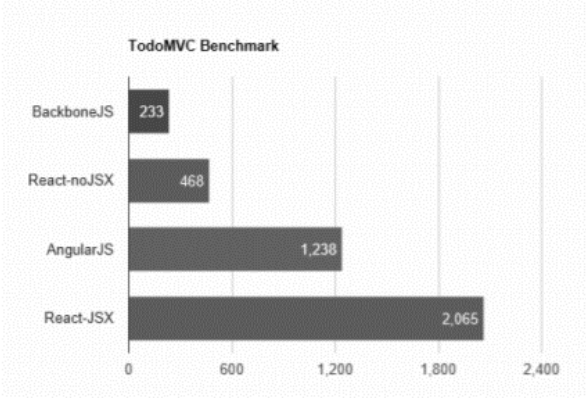

Figure 2. Average test duration for various JavaScript frameworks obtained by Mariano C.L. on a Microsoft Edge web browser (Mariano, 2017).

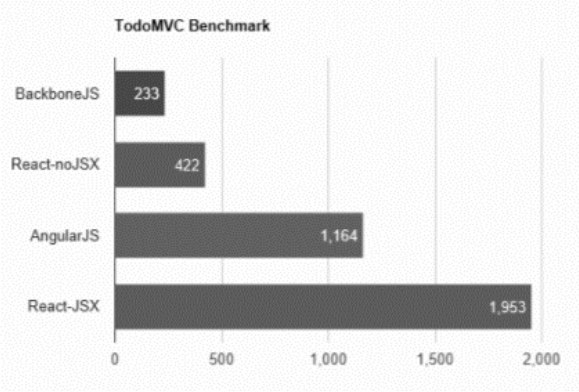

Figure 3. Average test duration for various JavaScript frameworks obtained by Mariano C.L. on a Mozilla Firefox web browser (Mariano, 2017).

Mariano C.L. in the conclusions of the tests indicated that the results slightly differ in different browsers (Mariano, 2017). However, in all tests BackboneJS turned out to be the most efficient JavaScript framework.

Similar performance testing of JavaScript frameworks was carried out by Meyghani (2020). He used the Stefan Krause tool (Krause, 2020) to test the performance of JavaScript frameworks and libraries. The tests were performed on a MacBook Pro (Retina, 15-inch, Mid 2015) with a 2.2 GHz Intel Core i7 processor, 16 GB RAM 1600 DDR3 memory and Intel Iris Pro

1536 MB graphics using Google Chrome version 69.0.3497.100. The author divided the frameworks he researched into pairs, where the framework with better results in pair was compared with a framework from another pair. Aspects such as: DOM element manipulation, start time, memory allocated by the application and actions simulation (understood as updating 1000 rows at the same time) were taken into account. The results of the author's research (Meyghani, 2020) are shown in Figure 4.

Based on Figure 4, it was noticed that in the Meyghani (2020) tests the Choo framework was the best. As a conclusion the author of above tests stated, that these tests are only examples and the research itself should be focused on application-specific requirements and take into account the real problem of users and business problems, instead of artificially generated behavior.

Name	elm-v0.19.0-bugfix2-keyed	choo-v6.13.0-keyed	mithril-v1.1.1-keyed	marionette-v4.0.0-beta.1-keyed	angular-optimized-v6.1.0-keyed	angularjs-v1.7.4-keyed	aurelia-v1.3.0-keyed
ready memory Memory usage after page load.	2.4 ± 0.3 (1.1) 2.636%	2.4 ± 0.3 (1.1) 3.436%	2.2 ± 0.2 (1.0) 0.014%	2.8 ± 0.3 (1.3) 100.000%	3.3 ± 0.2 (1.5) 0.037%	3.3 ± 0.3 (1.5) 0.128%	4.3 ± 0.3 (2.0) 0.000%
run memory Memory usage after adding 1000 rows.	6.0 ± 0.0 (1.8) 0.000%	3.4 ± 0.0 (1.0) 0.000%	6.1 ± 0.0 (1.8) 0.000%	4.8 ± 0.0 (1.4) 100.000%	6.0 ± 0.0 (1.7) 0.000%	10.9 ± 0.0 (3.2) 0.000%	8.5 ± 0.0 (2.5) 0.000%
update eatch 10th row for 1k rows (5 cycles) Memory usage after clicking update every 10th row 5 times	6.1 ± 0.0 (1.5) 0.000%	4.2 ± 0.0 (1.0) 0.000%	8.2 ± 0.0 (1.9) 0.000%	4.8 ± 0.0 (1.1) 100.000%	6.1 ± 0.0 (1.4) 0.000%	11.0 ± 0.0 (2.6) 0.000%	8.5 ± 0.0 (2.0) 0.000%
replace 1k rows (5 cycles) Memory usage after clicking create 1000 rows 5 times	6.1 ± 0.0 (1.7) 0.000%	3.6 ± 0.2 (1.0) 0.000%	6.2 ± 0.0 (1.7) 0.000%	5.3 ± 0.0 (1.5) 100.000%	6.4 ± 0.0 (1.8) 0.000%	11.5 ± 0.0 (3.2) 0.000%	8.8 ± 0.0 (2.4) 0.000%
creating/clearing 1k rows (5 cycles) Memory usage after creating and clearing 1000 rows 5 times	2.8 ± 0.0 (1.0) 0.000%	2.7 ± 0.0 (1.0) 0.000%	2.8 ± 0.0 (1.0) 0.000%	3.4 ± 0.0 (1.2) 100.000%	3.9 ± 0.0 (1.4) 0.000%	4.4 ± 0.4 (1.6) 0.003%	5.7 ± 0.0 (2.1) 0.000%

Figure 4. Test results obtained in tests by Meyghani (2020).

6. Summary

Based on the above documented research, pros and cons of different approaches can be noted. Both Mariano C.L. (2017) and Meyghani (2020) examined the metrics that were not relevant from the real point of application. Mentioned authors simulate user actions in their test

runs, where one of them has stated in the summary that such studies do not make sense, because they should focus on real problems specific to the application. In addition, Mariano C.L. did not provide specifications for the research environment that may be relevant when choosing a technology.

The authors of the article showed the problem of web application performance from the end user's viewpoint of the system. Comparison of the performance of web application clients using the JavaScript programming language was performed. Currently on the market there are a lot of JavaScript programming language frameworks supporting the development of client web applications. The authors of these frameworks claim that their product is the best on the market. Each of the JavaScript libraries also has its own group of supporters among programmers who often argue about which of them is the best, the most efficient. Such a wide range of existing solutions for creating web client applications means that new developers do not know which technology is worth learning, so that efficient web application clients can be implemented. The authors of this article has suggested users to familiarize themselves with the techniques used to conduct software performance tests. An important tip for the user is to become familiar with the tools available on the market that can be used during application performance tests, and the techniques for creating efficient web applications independent of the technologies used. Another important tip is to pay attention to the impact of web application performance on their real-life running in the market. Based on the impact of inefficient applications on the application manufacturer profits and application end-users (Wagner, 2020; Subraya, 2006) the authors of this article state that when conducting an software performance analysis, measures that are relevant to the end user of the application should be considered. Additionally, different types of devices with different screen sizes and networks with different data transmission parameters should be examined. The conducted performance research and analysis of the obtained data using statistical tests will be a reliable source of choice of technology for implementation by interested programmers.

References

1. Cassone, G., Elia, G., Gotta, D., Mola, F., Pinnola, A. (2001). Web Performance Testing and Measurement: a complete approach. *Telecom Italia Lab*. Retrieved from https://www.agileconnection.com/sites/default/files/article/file/2012/XDD3579filelistfilename1_0.pdf, 31.03.2020.
2. Khan, R., Mittal, K. (2018). Performance Testing and Profiling of Web based Application in Real Time. *International Journal of Computer Applications*, *Vol. 180, No. 46,* doi: 10.5120/ijca2018917220.

3. Krause, S. (2020). G*ithub repository of the JavaScript frameworks performance analysis tool*. Retrieved from https://github.com/krausest/js-framework-benchmark, 31.03.2020.
4. Mariano, C. L. (2017). *Benchmarking JavaScript Frameworks*. Master's thesis, 2017. doi:10.21427/ D72890.
5. Meyghani, A.J. (2020). *JavaScript Frameworks, Performance Comparison*. Retrieved from https://medium.com/@ajmeyghani/javascript-frameworks-performance-comparison-c566d19ab65b, 31.03.2020.
6. Ninka, I., Proko, E. (2013). Analyzing and Testing Web Application Performance. *International Journal Of Engineering And Science*, *Vol. 3, Iss. 10*, pp. 47-50.
7. Sharmila, S., Ramadevi, E. (2014). Analysis of Performance Testing on Web Applications. *International Journal of Advanced Research in Computer and Communication Engineering*, *Vol. 3, Iss. 3*.
8. Subraya, B.M. (2006). *Integrate Approach to Web Performance Testing: A Practitioner's Guide.* London: IRM Press.
9. Wagner, J. (2020). *Why performance matters*. Retrieved from https://developers.google.com/web/fundamentals/performance/why-performance-matters, 30.03.2020.