
# Finding a suitable performance testing tool

University of Oulu
Faculty of Information Technology and
Electrical Engineering / UNIT
Master's Thesis
Janne Annunen
29.03.2021

# Abstract

The pursuit of finding the most suitable testing software for each project is a difficult task as there are a lot of software effective finding certain kind of problems but completely missing others in the field of stress and load testing. A silver bullet solving all problems in a cost effective and reliable way has not yet been found. This project was done as a systematic literature review to find whether there are solutions documented capable of testing everything in a cost-effective way.

The document starts with an introduction of the task, originating from a real software testing company's suggestion of finding suitable test software that can, cost effectively and reliably, fulfil the needs of the company. A history section is describing the reason of testing importance, basics of testing and what others have found in their studies of the area. The research method is described in detail followed by results describing tools found during the research divided in sections by license type. The sectioning by license type was selected for the benefit of testing companies that are interested in further developing tools found to their own interest. Findings and answered research questions were presented and discussed followed by possible implications and further research suggestions to future scholars interested in the matter.

The systematic literature review found a total of 40 different tools identified during the data extraction process. One complete software system was available commercially including heavy support and help functions for the customer. A different approach linking open source and relatively inexpensive pieces of software together to achieve a composite solution was also identified. The solution included the most common and most popular individual piece of software identified by the study. All found pieces of software were listed and commented briefly mainly with information originating from the authors' home pages.

*Keywords*

Software testing, Stress testing, Load testing

*Supervisor*

D. Sc (Tech), Professor Mika Mäntylä

# Contents

# 1. Introduction

The purpose of this Master Thesis was to evaluate, whether a more suitable software stress load testing tool could be found among the most used free or commercial ones in the market, to replace the combination of BlazeMeter and JMeter software's for testing customer websites. The main reason is the need of finding possible solutions for future use as the cost factor, when upscaling Testing as a Service (TaaS) business is heavy as the form of reimbursement for each business magnitude level is fixed and annually charged in advance in the BlazeMeter solution format (Yan, Sun, Wang & Liu, 2012, BlazeMeter.com, 2019). Alas, the growth of business cannot be predicted and may cause unforeseen consequences to the company in form of over- or underestimating the market, which both are equally bad as overestimating forces the company to pay high annual fees in vain, while underestimating the market results in a situation where the company is not able to provide potential customers services in time and might result in loosing potential business opportunities to competitors providing same type of expertise (Prove, 2019). Business problems and opportunities often relate to increasing revenue or decreasing cost through the design of effective business processes according to Hevner, March, Salvatore, Park & Ram (2004).

The target of this thesis was to find out which is the most common Stress and Load testing software used according to literature findings and whether there is a tool or tool set suitable for complete website testing purposes. Prior research has been able to identify a lot of software capable of partially solving the complexity of testing as a whole and compared software against each other to justify the use of some software as a solution, alas, the solutions has not been complete or targeting a really marginal portion of the test area. The study was conducted as a systematic literature review and was able to identify some solutions capable of making testing more manageable, smoother and perhaps more productive linking different stages together with smart application programming interfaces (API).

These following chapters will cast a light on what has been studied on the subject by other authors, the methodology used, the study itself and how it was conducted, findings, discussion and implications the study might have on business development. The study will shed a light on the necessity of software change due to technical issues using the original setup, but will not bury deep into the technicalities of the proposed substitutes, rather encourage the possibility to test the other software solutions to find the most suitable package for the business volume and prospect chosen by the company officials as the lead strategy of the future as most of the software in commercials use foster supported test sessions to see, whether their product fits the customer needs and what is the most beneficial setup of the business kit provided.

# 2. History

The importance of internet as a medium between software providers providing web applications and users has grown a lot as web applications has become a very popular trend based on its flexibility and ease of access from everywhere. The nature of web software applications is versatile and can be used in various fields from education to entertainment, manufacturing to scientific simulation providing services directly from software companies to users according to Gan, Wei & Varadharajan (2005). According to Hossein (2013), "in the case of Web applications, performance of the system is a significant issue because Web users do not desire to stay too long for a reply to their requirements".

Hossain (2012) stated, that in cause of a website's poor quality, consumers stopped using the website or even abandoned using product the website was promoting entirely. Quality of Service (QoS) was originally released in this context by Cardoso, Sheth, Miller & Kochnut (2004) when they studied quality of service of workflows and web service processes. QoS itself is the key accessing how well Web-based applications meet customer expectations on two primary measure scales, availability and response time (Menache, 2012). Dhiauddin, Suffiani & Fahrurazi (2012) argued, there is no ready testing tool to verify, whether user experience and result reported by any testing tool are comparable in application performance and user experience in terms of response time experience.

According to Bezemer et al. (2016) performance evaluation activities require a considerable amount of time to ger statistically significant results in terms of common performance metrics such as response time, throughput, and resource utilization. Their study on how performance issues was addressed in DevOps, which is a modern software engineering paradigm that aims on high speed software change frequency and fast feedback cycles, found that 67% of the participants did not perform performance evaluation on regular basis, and those who did admitted, half of them used less than 5% of their time on them (Bezemer et al., 2019).

## 2.1 Testing

Testing in general can be divided into two categories, functional and non-functional requirement testing, according to Hossain (2013).

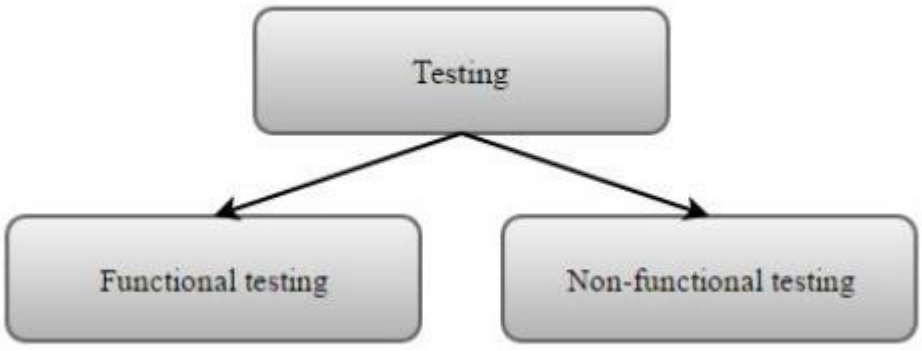



**Figure 1.** Testing divided into two main branches, Hossain (2013).

These two categories of testing are visualized in figure 1. Functional testing mainly focuses on validating functions and interactions that has been defined in users' requirements during software development stages according to the previously mentioned author (Hossain, 2013). Testing these functional requirements is not enough, even though the software seem to act according to specifications, the customer feeling might not be satisfying, as studies show almost 30% of users leaving the application, if the response time is more than eight seconds (Nguyen, 2012; Li, Shi & Li, 2013).

Pradeep and Charma (2019), Khan and Amjad (2016) and Paz and Bernandino (2016) defined testing terms described in Table 1 in their studies all regarding different tools and tool sets used for testing software as units, functionalities and as a ready package to be accepted by the customer

**Table 1**. Pradeep and Charma (2019), Khan and Amjad (2016) and Paz and Bernandino (2017) described different testing terms

| Term | Purpose |
|---|---|
| Smoke testing | Smoke testing ensures the working of important key features and the stability of software |
| Load testing / Volume testing | Evaluation of the software with intended number of users |
| Endurance Testing / Soak testing | The system is stressed for a longer period to check the performance |
| Scalability testing | The system is tested to be stable with the certified load, then users are subsequently increased to see the scalability performance |
| Stress testing / spike testing | The test analyses the robustness of the software. It identifies specific points where software modules get issues under extreme conditions of system failure |
| Fail over test | After relevant soak, load and stress tests are performed, fail over tests are performed to see, whether the software recovers from a critical situation or crashes completely |
| Security testing | Security testing is done to discover vulnerabilities and security loopholes in the software. It also includes penetration testing that tries to identify hacking or cracking probabilities of the software |
| Unit testing | Each level of software and module is tested to ensure correct behavior of the individual unit |
| Gorilla testing | The modules are tested repeatedly under assorted scenarios and inputs in order to verify the consistency of the software. The term also refers to frustrating testing that involves iterative testing of the same component and identification of bugs |
| Graphical User Interface (GUI) Testing | The proper functioning of the graphical interface of the software is tested to ensure functions work as required |
| Performance testing | The test set is a multidimensional evaluation of the software including speed, load, traffic, susceptibilities etc. |
| Acceptance testing | The evaluation of software based on the prescribed Software Requirements Specification (SRS) is done to make it deliverable. The levels and scores for acceptability of software product are investigated |

Ahmad, Brereton and Andras (2017) made a systematic mapping study on Empirical studies on software cloud testing methods to see, what empirical studies were made in the area software cloud-based testing to find out testing methods, application of these methods and purpose of testing using these methods. Software categories studied are presented in Table 2 as numbers.

**Table 2**. What category of software's were tested in cloud based testing studies (Ahmad, Brereton and Andras, 2017)

| Category | Studies (count) |
|---|---|
| Web services / Application testing | 10 |
| Mobile testing | 7 |
| Vulnerability and security configuration testing | 11 |
| Benchmarking | 8 |
| Testing SaaS | 10 |
| Testing cloud services | 10 |
| Large-scale testing | 7 |
| Other ways of testing | 6 |

Khan & Khan (2012) in their comparative study paper of different testing techniques describes and compares three main testing techniques: White box, Black box and Grey box testing. These testing techniques differ from each other mainly by tester's level of knowledge of the software running inside the software under test. Black box testing can be used for both functional and non-functional testing and is mainly used when systems under test are big and complex and there is no or little knowledge of the internal relationships of different parts of the program and the test is interested in whether the software does what it is supposed to. In white box testing the internal structure of the software is known to the tester (Khan & Khan, 2012). Therefore, the test is mainly applied to unit testing. Grey box testing is a combination of these two which means the tester has some insight of the operating software and relationships between different processes (Khan & Khan, 2012; Software testing fundamentals, 2017).

## 2.2 Software performance

Connie Smith, who coined out the term Software Performance Engineering in 1981, brought to the attention the "fix-it-later" attitude when it came to performance in software engineering (Smith, 1981). Menasce (2002) points out, that this lack of performance evaluation from beginning of design stage, could never be allowed to any other form of engineering using a quite illuminating example of mechanical engineering with an engine that should reach 4000 RPM to find out it does not go over 1500 RMP when built and tested. Clearly this kind of mismatch between requirements and performance is not possible as, in normal engineering, performance is an integral part of design process according to the author (Menasce, 2002).

Software performance is a pervasive quality aspect difficult to understand, because it is affected by every aspect of the design, code, and execution environment according to Woodside, Franks & Petriu ,2014. Same authors defined Software Performance Engineering in the study as follows: "Software Performance Engineering (SPE) represents the entire collection of software engineering activities and related analyses used throughout the software development cycle, which are directed to meeting performance requirements". Originally in the 70's, the need for efficient software was a necessity due to machine size, both in terms of memory and processor abilities (Smith, 1990). Unfortunately, the growth of hardware did not fix this issue, rather giving way to

more complex software that became systems of programs. Only high-end software's, such as flight control systems or other mission-critical embedded systems got the proper attention in performance perspective, as they had strict performance requirements in the specifications from beginning (Smith, 2015). As everything is related to cost effectiveness, "fix-it-later" is still a trend today. If the software performance engineering methods is not required by the contractor specifically, it is likely the performance issue is left out (Smith, 2015).

In the past software performance issues were discovered very late in the development of a product as performance validation, if even made, was one of the latest activities done to the software before publication (Barber, 2006). According to the previous author, with agile processes, the problem is unchanged or even worse. The tendency of having testing, diagnosis and tuning activities quite late in a software development cycle is confirmed by several studies, as these phases needs the system under development to be ready to act and execute in an environment, where it can be run and measured as it would in the final environment (Arlitt M., Krishnamurthy D., Rolia J, 2001; Avritzer a., Kondek j., Liu d. & Weyuker E. J., 2002; Barber S., 2004a; Barber S., 2004b; Field, Chatley & Wei, 2018).

In 2011 Vodde, one of the founders of LeSS framework (Large Scale Scrum), was interviewed by Kircher on an audio podcast regarding large Agile software development. He stated, that Continuous Integration (CI) is the most important practice in adopting agile at scale (Kircher & Vodde, 2011). Field et al. (2018) also adds, that performance testing is not only taking place late and is usually performed manually without any generic performance testing framework or tooling. According to Stefan, Horký, Bulej & Tuma (2017), only 0.4% of over 90000 open-source GitHub projects used any framework or was aligned with continuous delivery.

As software development has moved from artistic phase based, highly skilled software craftsmen towards a real industry, where quality is controlled by introducing a structured workflow comparable with any other manufacturing process rather than by skills of a few individuals (Ricca & Tonella, 2001). A systematic mapping study of testability and software performance was made by Hassan et al. (2016) that implied most, if not all, the studies focused on functional correctness of the software and very little is known regarding what software testability issues impacts non-functional properties other than the ones dealing with the time-factor (timeliness and response time). One solution to solve this might, according to Field et al. (2018), is to use virtual software performance testing, which allows mockups to facilitate testing before all parts of the software is implemented.

Ferme and Pautasso (2017) declared, that researchers and practitioners do identify the importance of performance testing in agile development processes, but states also, the existing techniques are fragmented, and the reaction speed is not synchronized to the intrinsic velocity of the software development. Their high-level solution is mentioned in the testing tools chapter that follows.

## 2.3 Testing tools

Ferme & Pautasso (2018) made a paper on performance test execution in continuous software development environments, where they noticed current performance testing approaches are mostly based on scripting languages and framework where users implement, in a procedural way, the performance tests issued to the system under test. This leave, still, the most important things undefined, the test goals and intents, according to the Ferme and Pautasso (2018). As a tool they suggest a declarative model-

driven approach using a domain specific language (DSL) solution that build on existing tools like BlazeMeter and other tools, but as there is a plethora of tools, the solution is more discussed in a higher level allowing the user to specify the performance intent, solution and performance test execution. A follow up on this quite recent work will be provided after applied to real-world usage scenarios and feedback collected, according to Ferme & Pautasso (2018). Shariff et al. (2019) stated, that JMeter is the de facto standard for testing request-based frameworks. Selenium based testing is very suitable for browser load testing but is unfortunately extensively resource heavy as each test user starts a new browser. The result, however, gives a more realistic view on the end-to-end behavior of an application under load (Shariff et al., 2019).

Cordell Vail (2005) made a large research on load, volume, performance, benchmark and base line testing tool evaluation, where he compared installation, usability, pricing of the usage and total benefit of the tools presented. Even though the paper went through an impressive number of tools, no recommendation could be given by the author of which tool set is best in terms of cost, usability or total revenue. Raj_esh_0201 (2008) uploaded a performance test tools comparison describing basic functionalities of some of the, at that moment, state of the art testing tools including LoadRunner, Silk Performer, JMeter and some other software tool setups, but also indicated no tool was superior to others as they all are, as also concluded by Kaur & Gupta (2013) in their research, best chosen by the user based on budget and nature of the software that has to be tested.

A study by Raulamo-Jurvanen, Mäntylä and Garousi (2017) addressed the problem of finding the right test automation tool in a large literature study, which addressed both formal studies and experience reports gathered from projects and contexts, shared online by practitioners. This, more informal data, is referred to as grey literature and is, according to the authors, an asset addressing the question of choosing right test tools most suitable for the system under test.

Different load testing tools are compared by different web pages in example by G2.com (2020) and Softwaretestinghelp.com (2020). Open Source and licensed programs are rated in several using terms like “highest rated” and “easiest to use” (G2.com, 2020, Softwaretestinghelp.com,2020).

**Table 3**. Features to the table were collected from homepages of HP LoadRunner, JMeter, Grinder, WebLOAD and Selenium.

| Feature | HP LoadRunner | JMeter | Grinder | WebLOAD | Selenium |
|---|---|---|---|---|---|
| Licensing | Expensive, six figures (2016) | Apache 2.0 | BSD-style open-source | Not available | Apache 2.0 |
| Virtual users | Restricted to license | Restricted by hardware | Yes | 50 free, over 50 by license | Yes |
| Cross platform | Windows and Linux | Windows, Linux, unix, mac | Windows, Mac, Linux | Windows and Linux | Windows, Mac, Unix, linux |
| Scriptable | C, VBA, VBScript, Jscript, VB, VB.NETC, C#, Java | Limited (XML) | Jython, Closure | Javascript | C#, Groovy, Java, Perl, PHP, Python, Ruby and Scala |

Table 3 includes some features obtained from the software authors' homepages for comparison. The selected features are gathered from the internet as this is, according to Raulamo-Jurvanen, Mäntylä and Garousi (2017), the primary source of information, alas, test tools and automation related services are ranked among the most required services from external consultants, which is acknowledged by practitioners. Raulamo-Jurvanen, Mäntylä and Garousi (2017) also claim, tool evaluation is only recommended if the people testing it can devote enough time and appropriate expertise to complete a thorough trial use as a study by Poston and Sexton (1992) already claims that trial use would often lead to wrong decisions, mainly due to lack of time for testing and evaluation of the tool and also indicates user expertise level issues to be an element causing result misinterpretation of the usability and functionality of the software.

# 3. Research method

The research method was a systematic literature review of tools used in Stress and Load testing conducted following both the guidelines provided by Kitchenham and Charters (2007) and guidelines by Petersen, Feldt, Mujtaba and Mattsson (2008).

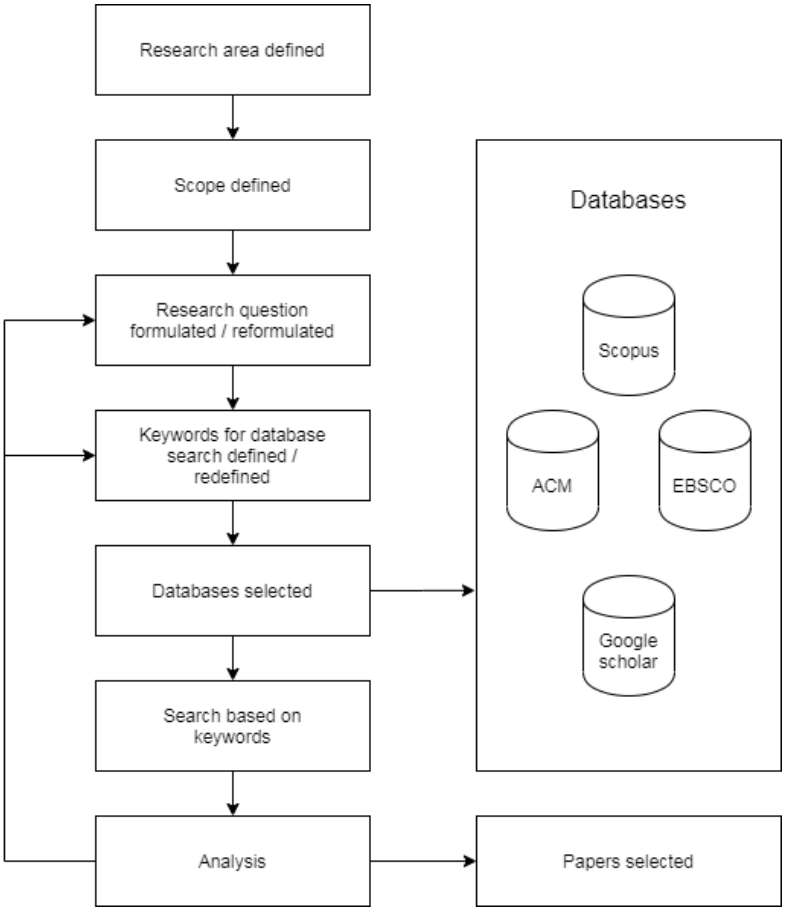



**Figure 2.** Search progress

Steps taken to achieve the literature review are presented in Figure 2 and are step by step explained in the following subchapters.

## 3.1 Research questions

To find out whether there is a superior tool on the market in aspect of usability and cost, following research questions were formulated.

> RQ1. What is the most common Stress and Load testing software tool used according to literature findings?
>
> RQ2. Is there a recommended tool or tool set for website testing purposes?

To answer both RQ1 and RQ2, current research literature had to be explored in order to find evidence of existence of such findings.

## 3.2 Search strategy

The scope of the study was defined to find suitable tools for stress and load testing in software development literature. The research questions were set and modified to their final form to fit the scope defined. Keywords for database searches were defined, searches made, words were redefined to final form to ensure enough relevant paper was included in search results. The study utilized references used by similar studies as these were similarly relevant for this paper.

## 3.3 Sources of data

Articles and journals were mostly accessible through Oulu University student login even though some sources required their own login-procedures according to their own security policies, especially when utilizing automated search engine and result modifying tools such as RStudio (https://rstudio.com/). Sources for data retrieved are listed below with a short summary of their key functions as described by Oulu university webpage "Communication and information engineering, electronics and information Processing Science subject guide: Articles and Databases" (http://libguides.oulu.fi). By accessing the page and logging in with university access codes, most of the material needed became available.

Scopus (http://www.scopus.com)

Scopus is a key reference database holding multidisciplinary abstract and citation database of journals, conference papers, trade publications, book series and patents

ACM Digital Library (https://dl.acm.org/)

ACM digital library is a full text database with articles and bibliographic citations mainly in computing sciences and a reference database

EBSCO Databases (https://www.ebsco.com)

EBSCO database library is a key reference database with many different subject areas with full text and reference databases

Google scholar (https://scholar.google.com)

Google scholar searches articles based on title using Google as information source. As Google scholar does not distinguish between academically approved and documents being in reviewing process, prudence is advised using documents not presented by other, academically stricter, sources.

## 3.4 Data collection

The search string used was of generic type:

(X1 OR X2 OR .. Xn) AND (Y1 OR Y2 OR .. Yn)

Where X covered words used in Stress and Load testing and Y covered the area of software engineering. As there were a relatively small number of suitable documents available, the search string had to be simplified to generic level to ensure enough potential documents would be presented in the search.

X: {Stress testing, Load testing, Tool}
Y: {Comp, Engi}

The search string itself was several times reformulated and searches were re-conducted based on the results, reflected against the research questions and object of the study. The literature search, which produced basic reference lists, was done by searching Scopus by Elsevier. The Scopus search found also documents preselected from other sources, which gave confidence in presenting the research sources as many instead of just one.

Documents found from Scopus resulted in 361 hits with Stress Testing as key indicator in the area of computer engineering (Appendix A). Common words were filtered out such as paper and software to better describe the important words in these papers. Scopus was chosen to demonstrate word cloud visualization due to best compliance of R-tool used for extracting information. The search scope limitation to less than 9 years of age dropped the document count to 112 documents.


**Figure 3.** Word cloud with stress testing as key word.

Stress testing as key indicator produced a word cloud shown in Figure 3. A similar search with the key indicator Load Testing resulted in 987 hits (Appendix B) and is visualised in Figure 4 below. The search scope limitation to less than 9 years of age dropped the document count to 36 documents.

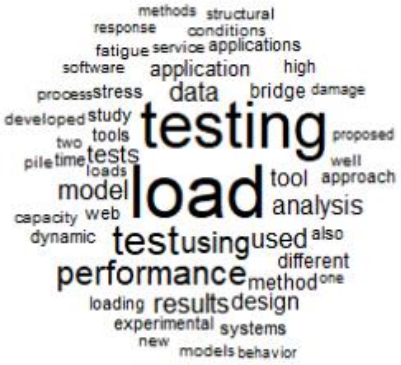



**Figure 4.** Word cloud with load testing as key word.

Both word clouds had testing as one of the key elements and indicates performance, analysis and model frequently appearing in the papers found by the search made from the Scopus library. Furthermore, the search made with the keyword “stress” shows “load” appearing in frequent words and vice versa, the search made with the keyword “load” indicates “stress” being one key word in these found documents.

No limitations were set on publication year in order to get as much relevant papers included in the preliminary search. “Load Testing” brought 987 hits in the forts phase and the key word “Stress Testing” gave 361 hits. Different forms of spelling of the key words did not affect the outcome of the search results. Only peer-reviewed documents were taken into consideration. Gray literature was included as some sources had made their studies on that area, was accepted and notified by fellow scholars. The study discarded documents older than from 2011 based on lack of technical value to the research and the fact, there was not much to find of value.

## 3.5 Inclusion process

The inclusion/exclusion decision of the documents retrieved from Scopus searches was made on the base of reading the title and analyzing the abstract due to the fact that full-text was not available through the sources used and paying for document possibly excluded later would be a too heavy load for a single person to handle. The classification of documents based on title, abstract or keywords was categorized as “irrelevant”, “maybe relevant” and “relevant”. The inclusion / exclusion process is described in Table 4

**Table 4**. The amount of included and excluded documents. Duplicates, irrelevant and documents not answering to the research questions were discarded

| Search hits according to criteria | Original count | Accepted for evaluation | Plausible | Excluded | Included |
|---|---|---|---|---|---|
| Load testing | 987 | 112 | 88 | 48 | 64 |
| Stress testing | 361 | 36 | 25 | 28 | 8 |
| Total | 1348 | 148 | 113 | 76 | 72 |

Duplicates or papers with the same content as other studies were excluded at the analysis and synthesis step.

## 3.6 Data extraction and synthesis

According to guidelines by Petersen et al. (2008), the text was suggested to be studied adaptively in order to use time efficiently. Some texts valued more relevant to the study were read in full text as all necessary information, especially regarding numbers and statistics, were not fully covered neither in the abstracts nor summary contents. The results were extracted, decoded and stored in excel-sheets.

# 4. Results

A total of 44 different tools were identified during the data extraction process. Both open source software with different licence types and commercial versions with scalable solution packets were recognized to the study. Model based and model-based machine learning solutions was also taken into consideration as the complexity of modern web software and the growing capability of Artificial Intelligence (AI) in model-based machine learning is probably going to play a role in future solutions and testing strategies. Table 5 displays, that the most common tool referred to or used as testing tool or evaluation tool for other solutions has by far been Apache JMeter with 33 hits during the study period. It has been referred to or used steadily throughout the research period as well as HP LoadRunner, which has been referred to or used in 10 different publications. Model-based testing solutions has been referred to or used 7 times and selenium 5 times as testing tool or evaluation method for other tools. Model-based machine learning as a performance testing solution has been presented 2019 for the first time in this documentation but is still worth mentioning as a future solution possibility. To better illustrate the growing interest in testing, tools table 5 shows the number of hits recorded between the years 2011 and 2016 is 15 as between the years 2017 and 2019 the number of hits is 18, even though the time span is only half of the previous.

**Table 5**. Hits recorded in documents reviewed

| Most hits / Year span | Years | JMeter | HP LoadRunner | Model based testing | Selenium | Model based Machine learning |
|---|---|---|---|---|---|---|
| **2011 – 2019** | **9** | **33** | **10** | **7** | **5** | **1** |
| **Progress opened to illustrate growth of interest** | | | | | | |
| 2017 – 2019 | 3 | 18 | 5 | 3 | 2 | 1 |
| 2011 – 2016 | 6 | 15 | 5 | 4 | 3 | 0 |

One of the main reasons why documents prior to 2011 were discarded was the lack of research of testing tools. As table 6 shows, since 2011 there has been a steadily increasing need for research in the matter, with 2016 as the year, when the research and comparison of performance testing tools became interesting and relevant for scholars.

**Table 6**. Hits recorded in documents reviewed in annual level 2011 - 2019

| Year | Hits |
|---|---|
| 2011 | 1 |
| 2012 | 2 |
| 2013 | 5 |
| 2014 | 7 |
| 2015 | 6 |
| 2016 | 13 |
| 2017 | 13 |
| 2018 | 12 |
| 2019 (until August 2019) | 7 |

Web services and applications were mostly measured and tested during the research period. Table 7 presents what was tested in the documentation reviewed from 2011-2019. Web services and applications were mostly tested during the period. Internet of Things (IoT) is a growing test area, that showed up in the documentation 2018. Business Process Execution Language (BPEL) and Web Services Business Process Execution Language (WS-BPEL) has been tested throughout the research span.

The variety of tests has grown as the complexity of systems grow interconnecting with each other. Most documentation has by far been done regarding test tools of web services and applications. The tool that has throughout the years been the most popular for testing web software is Apache JMeter. BPEL and WS-BPEL documentation had not specified any specific testing tool used, merely new approaches and solved issues for the functionality of the BPEL and WS-BPEL software tool itself. IoT had used two different software very much based on the needs of the tested environment. The third document was a modelling of what should be measured in the future when testing IoT in general. In the developer tool segment, a combination of Wessbas, Apache JMeter and InspectIT was used for reducing the maintenance effort for parameterization of representative load tests using annotations improving throughput time by automating what should be tested. Table 7 has the chapter described as numbers.

**Table 7**. Hits recorded in documents reviewed

| Most hits / Year span | Web services / applications | BPEL / WS-BPEL | IoT | Big Data | Developer tools |
|---|---|---|---|---|---|
| **2011 – 2019** | **49** | **4** | **3** | **2** | **1** |
| **Most popular** | **Apache JMeter (23)** | **Tools not specified** | **MQTT broker (1), Soap UI (1)** | **Netdata (1), Modast (1)** | **Wessbas, Apache JMeter and Inspect IT used together (1)** |
| 2017 – 2019 | 22 | 2 | 3 | 1 | 1 |
| Most popular | Apache JMeter (13) | Tools not specified | MQTT broker (1), Soap UI (1) | Netdata (1) | Wessbas, Apache JMeter and Inspect IT used together (1) |
| 2011 – 2016 | 27 | 2 | 0 | 1 | 0 |
| Most popular | Apache JMeter (10) | Tools not specified | NA | Modast (1) | NA |

## 4.1 Testing Software found in the literature review

A large variety of software was mentioned and extracted in the literature review process. 40 different tools were documented to be used or evaluated by different authors. Even though JMeter and HP LoadRunner were the most referred ones, all the mentioned pieces of software was collected and provided with a short comment, mainly from the authors' homepage, organised according to license. The license is briefly commented at the beginning of each section.

### 4.1.1 Apache License 2.0

The Apache license 2.0 is a highly permissive open software license that allows the users to distribute, modify and use the software for any purpose, as long as the user complies with the license terms, that state existing copyright, patent, trademarks and attribution notices are not removed (apache.org, 2020) . As a limitation, you must add notifications of modifications made to the original software (apache.org, 2020). Table 8 lists software mentioned, that uses Apache 2.0 license including name of the tool, key function or operation, the official URL if found and a short description, mainly from the software's official loading URL.

**Table 8**. Programs under Apache License 2.0

| | Tool | Function or key operation | Official URL |
|---|---|---|---|
| 1 | **Apache Bench** | Apache HTTP server benchmarking tool | httpd.apache.org/docs/2.4/programs/ab.html |
| | Apache Bench is a tool for benchmarking your Apache Hypertext Transfer Protocol (HTTP) server. | | |
| 2 | **Apache Flood** | Load Testing, Performance Testing | httpd.apache.org/test/flood/ |
| | Flood is a profile-driven HTTP load tester. In layman's terms, it means that flood can generate large amounts of web traffic. Flood's flexibility and power arises in its configuration syntax. It can work well with dynamic content. | | |
| 3 | **Apache JMeter** | Load Testing, Performance Testing | JMeter.apache.org/ |
| | Apache JMeter may be used to test performance both on static and dynamic resources, Web dynamic applications. It can be used to simulate a heavy load on a server, group of servers, network or object to test its strength or to analyze overall performance under different load types | | |
| 4 | **Appium** | Testing of Hybrid, Native and Mobile Web Apps | appium.io/ |
| | Appium is a mobile test automation framework (with a tool) that works for all: native, hybrid and mobile web apps for iOS and Android. Appium is a great choice for test automation framework as it can be used for all these different app/web types. | | |
| 5 | **Gatling** | Performance Testing, Load Testing | gatling.io/ |
| | Gatling is a highly capable load testing tool. It is designed for ease of use, maintainability and high performance. | | |
| 6 | **Grinder** | Load Testing | grinder.sourceforge.net/ |
| | The Grinder is a load testing framework that makes it easy to run a distributed test using many load injector machines. Test scripts are written in Jython, and can call out to arbitrary Java code, providing support for testing a large range of network protocols. The Grinder comes with a mature plug-in for testing HTTP services, HTTP scripts can be recorded easily from a browser session. | | |
| 7 | **Selendroid** | Automation Testing for Mobile Apps | selendroid.io/ |
| | Selendroid is a test automation framework which drives off the UI of Android native and hybrid applications (apps) and the mobile web. | | |
| 8 | **Selenium** | Automation of Web Browsers Regression Automation, Exploratory Testing | seleniumhq.org/ |
| | Selenium is many things but at its core, it is a toolset for web browser automation that uses the best techniques available to remotely control browser instances and emulate a user's interaction with the browser. Although used primarily for front- | | |

| | | | |
|---|---|---|---|
| | end testing of websites, Selenium is at its core a browser user agent library. The interfaces are ubiquitous to their application, which encourages composition with other libraries to suit your purpose. | | |
| 9 | **TestNG** | Server Testing Performance Testing Data Driven Testing | testng.org |
| | TestNG is a testing framework designed to simplify a broad range of testing needs, from unit testing (testing a class in isolation of the others) to integration testing (testing entire systems made of several classes, several packages and even several external frameworks, such as application servers). | | |

### 4.1.2 Commercial software

According to Technopedia the definition of commercial software is that any software or program that is designed and developed for licensing or sale to end users or that serves a commercial purpose is commercial software (Technopedia, 2020). Both proprietary and open-source software can be classified as commercial depending on licensing as is or as a part of a service. Products are normally licensed, not sold, to the end user (Technopedia, 2020). Table 9 lists software mentioned, that uses commercial software licensing including name of the tool, key function or operation, the official URL if found and a short description, mainly from the software's official loading URL.

**Table 9**. Programs under commercial licenses

| | **Tool** | **Function or key operation** | **Official URL** |
|---|---|---|---|
| 10 | **Amazon kinesis** | Testing real time video and data stream applications | aws.amazon.com/kinesis/ |
| | Amazon Kinesis is a managed, scalable, cloud-based service that allows real-time processing of streaming large amount of data per second. It is designed for real-time applications and allows developers to take in any amount of data from several sources, scaling up and down that can be run on EC2 instances. | | |
| 11 | **HP ALM** | To schedule and run tests | microfocus.com/en-us/products/quality-center-quality-management/download |
| | HP ALM/Quality Center is an application lifecycle management tool for software quality assurance and test management to deliver apps quickly with confidence. | | |
| 12 | **HP LoadRunner** | Stress testing, Performance testing | microfocus.com/en-us/products/loadrunner-professional/ |
| | LoadRunner is a software testing tool from Micro Focus. It is used to test applications, measuring system behavior and performance under load. LoadRunner can simulate thousands of users concurrently using application software, recording and later analyzing the performance of key components of the application. | | |
| 13 | **Silk test** | Functional testing, Regression testing | microfocus.com/en-us/products/silk-test/ |
| | Silk Test is a test automation solution for web, mobile & enterprise apps, enabling software testers & developers to conduct functional & regression tests. | | |
| 14 | **SoapUI** | SOAP Testing, REST Testing | soapui.org/ |
| | SoapUI is the world's leading Functional Testing tool for SOAP and REST testing. With its easy-to-use graphical interface, and enterprise-class features, SoapUI allows you to easily and rapidly create and execute automated functional, regression, and load tests. | | |
| 15 | **WAPT** | Recorder and Load testing | loadtestingtool.com |

| | | | |
|---|---|---|---|
| | Record, use several systems for load generation, remotely control test execution, monitor server performance and handle complex parameterization. | | |
| 16 | **WebLoad** | Load Testing, Response Validation Testing | radview.com/webload-download/ |
| | WebLOAD is a load testing tool from Radview software that tests for performance and scalability but also for verifiability (validating the correctness of return results). ... This past April Radview released an open source community edition of WebLOAD under GPL, available at webload.org | | |

### 4.1.3 GPL licenses

The GNU GPL (General Public License) or simply GPL is a permissive license that gives the end user the right to use, share and modify the software if the copyleft rule is respected and preserved under same equivalent license terms (GNU.org, 2020). GPL version 1 from 1989 made the distributors publish their code in human readable source code form and made sure the licensed software GPLv1 could be combined with software under more permissive codes preserving same terms (GNU.org, 2020). GPLv2 in 1991 stated the GPL license may be distributed only if all license obligations can be fulfilled.

The GNU Library General Public License version 2 was released to ensure C-libraries and other software libraries in the same year. GPLv3 increased compatibility with other software licenses such as Apache license 2.0 and GNU Affero General Public license, which should be used for software interacting over a network (GNU.org, 2020). Lesser General Public License (LGPL) allows the work to be linked with and used in a different form of software licensed program which does not apply (L)GPL licensing (GNU.org, 2020). Software applying GPL and derived licenses are listed in table 10.

**Table 10**. Programs under GPL and derived licenses

| | **Tool (license)** | **Function or key operation** | **Official URL** |
|---|---|---|---|
| 17 | **Siege (GPL)** | Web server testing tool | joedog.org/siege-home/ |
| | Siege is an open source regression test and benchmark utility. It can stress test a single URL with a user defined number of simulated users, or it can read many URLs into memory and stress them simultaneously. The program reports the total number of hits recorded, bytes transferred, response time, concurrency, and return status. Siege supports HTTP/1.0 and 1.1 protocols, the GET and POST directives, cookies, transaction logging, and basic authentication. Its features are configurable on a per user basis. | | |
| 18 | **OpenSTA (GPL)** | Stress Testing, Web Load Testing | opensta.org/ |
| | The current toolset has the capability of performing scripted HTTP and HTTPS heavy load tests with performance measurements from Win32 platforms. | | |
| 19 | **Pylot (GPL)** | Load Testing, Benchmarking, Capacity Planning, System Tuning | testmatick.com/testing-tools/pylot/ |
| | Pylot is a tool for testing performance and scalability of web applications. It simulates HTTP requests and checks how the server responds. After the tests the instrument creates the test report that includes important metrics. | | |
| 20 | **Ansible (GPL)** | Distributed systems testing | www.ansible.com/products/ |
| | Under RedHat for testing Ansible contributions | | |
| 21 | **Httperf (GPLv2)** | Web server performance tool | github.com/httperf/httperf |

| | | | |
|---|---|---|---|
| | httperf is a tool for measuring web server performance. It provides a flexible facility for generating various HTTP workloads and for measuring server performance.<br>The focus of httperf is not on implementing one particular benchmark but on providing a robust, high-performance tool that facilitates the construction of both micro- and macro-level benchmarks. The three distinguishing characteristics of httperf are its robustness, which includes the ability to generate and sustain server overload, support for the HTTP/1.1 and SSL protocols, and its extensibility to new workload generators and performance measurements. | | |
| 22 | **Tsung (GPLv2)** | Stress Testing, Distributed Load Testing | tsung.erlangprojects.org/ |
| | Tsung is an open-source multi-protocol distributed load testing tool. It can be used to stress HTTP, WebDAV, SOAP, PostgreSQL, MySQL, LDAP, MQTT and Jabber/XMPP servers. | | |
| 23 | **Flowping (GPLv3)** | Stress testing | github.com/k13132/flowping |
| | The FlowPing is an application which allow user to perform variety of network throughput and stress tests. The application utilize UDP(User Datagram Protocol). | | |
| 24 | **Jattack (GPLv3)** | WebRTC stressing tool | prezi.com/krg1esxoa6ug/jattack/ |
| | Jattack is an automated stressing tool for the analysis of the performance of WebRTC-enabled server-side components | | |
| 25 | **TailBench (LGPL)** | Performance testing tool | tailbench.csail.mit.edu/ |
| | A benchmark suite and evaluation method for testing Latency-critical applications | | |
| 26 | **Bench4Q (LGPLv2.1)** | Load simulation tool | projects.ow2.org/view/bench4q/ |
| | Bench4Q is a QoS oriented B2C benchmark for Internet Middleware. It makes many extensions of TPC-W, especially for load simulation and metrics analysis of a benchmark. | | |
| 27 | **CLIF (LGPLv3)** | Performance testing | clif.ow2.io/ |
| | Automated performance testing, performance testing in continuous integration, providing a simple web user interface for CLIF, monitoring QoS or applications QoE and possibly send alerts in case of bad responsiveness. | | |
| 28 | **MultiMechanize (LGPLv3)** | Load Testing, Performance Testing, Scalability Testing | multimechanize.readthedocs.io/en/latest/ |
| | Multi-Mechanize is an open source framework for performance and load testing. It runs concurrent Python scripts to generate load (synthetic transactions) against a remote site or service. | | |

### 4.1.4 MIT licenses

The MIT license is a highly permissive open software license that gives permission to reuse and modify code for any purpose if the original copy of the MIT license is included in their distribution (opensource.org/licenses/MIT, 2020). Table 11 presents a list of software using MIT licenses.

**Table 11**. Programs under MIT License

| | Tool | Function or key operation | Official URL |
|---|---|---|---|
| 29 | **AutoPerf** | Testing tool for web applications | github.com/mejbah/AutoPerf |

| | | | |
|---|---|---|---|
| | Autoperf is a tool for automated diagnosis of performance anomalies in multithreaded programs. It operates in two phases:<br>Profiling: Collects hardware performance counters from annotated sections of a program by running it with performance representative inputs.<br>Anomaly Detection: Creates a model of application performance behavior by training an Autoencoder network. It finds out the best performing network by training for input dataset (collected in profiling phase). AutoPerf uses the trained model for anomaly detection in future executions of the program. | | |
| 30 | **Capybara** | Simulation of User Behavior | github.com/teamcapybara/capybara |
| | Capybara helps you test web applications by simulating how a real user would interact with your app. It is agnostic about the driver running your tests and comes with Rack::Test and Selenium support built in. WebKit is supported through an external gem | | |
| 31 | **Cucumber** | Acceptance Testing | cucumber.io/z |
| | A cucumber is a tool based on Behavior Driven Development (BDD) framework which is used to write acceptance tests for the web application. It allows automation of functional validation in easily readable and understandable format (like plain English) to Business Analysts, Developers, Testers, etc | | |
| 32 | **Excactpro** | Trading system testing | exactpro.com/ |
| | A tool for testing high load trading systems with the required performance characteristics | | |
| 33 | **HULK - HTTP Unbearable Load King** | Ddos attack tester | github.com/siarheidudko/hulk |
| | This tool is a dos tool that is meant to put heavy load on HTTP servers in order to bring them to their knees by exhausting the resource pool, its is meant for research purposes only and any malicious usage of this tool is prohibited. | | |
| 34 | **Locust** | Performance Testing, Load Testing, Benchmarking | locust.io |
| | Locust is an easy to use, scriptable and scalable performance testing tool. You define the behavior of your users in regular Python code, instead of using a clunky UI or domain specific language. This makes Locust infinitely expandable and very developer friendly. | | |
| 35 | **Watir** | Automation Testing | watir.com/ |
| | Watir stands for Web Application Testing In Ruby. It facilitates the writing of automated tests by mimicking the behavior of a user interacting with a website. | | |
| 36 | **Webrat** | Acceptance Testing, Browser Simulation | github.com/brynary/webrat |
| | Webrat lets you quickly write expressive and robust acceptance tests for a Ruby web application. | | |
| 37 | **FltNesse (MIT, Common Public License 1.0)** | Acceptance Testing | fitnesse.org/ |
| | FitNesse automated acceptance tests are power tools for fixing a broken requirements process. | | |

### 4.1.5 Other or not specified

Three software was referred to in findings of the literature study, but license type was not specified, or claimed license agreement being “other”. These are presented in table 12.

**Table 12**. Programs, that has not specified license type

| | Tool (License) | Function or key operation | Official URL |
|---|---|---|---|
| 38 | **WebRTCBench (not specified license)** | WebRTC stressing tool | github.com/ucisysarch/WebRTCBench |
| | WebRTCBench, an open source tool for performance assessment of WebRTC implementations which allows testing applications making use of video and audio through WebRTC standards and collects performance indicators. | | |
| 39 | **Canoo Web Test (other)** | Automation Testing | webtest.canoo.com/ |
| | CanooWebTest is an OpenSource tool that uses Ant and HttpUnit to implement functional testing of web applications. | | |

## 4.2 Recording software mentioned in the literature review

Software used for recording and play back user actions on web browsers are listed in table 13. These tools are used to mimic user behaviour to be repeated in test sessions, often altered to suite the test scripts purposes, in example a multitude of user logins, purchases, downloads and so on to test a web service or application.

**Table 13**. Tools for recording browser activity mentioned in the literature review

| | Tool | License type | Official URL |
|---|---|---|---|
| 40 | **BadBoy** | Commercial | badboy1.software.informer.com/2.1/ |
| | Badboy embeds Internet Explorer and monitors and controls its actions. Badboy makes web testing and development easier with dozens of features including a simple yet comprehensive capture/replay interface, powerful load testing support, detailed reports, graphs | | |
| 41 | **Blazemeter** | Commercial (Platform as a Service) | https://www.blazemeter.com/ |
| | A self-service load testing Platform as a Service (PaaS), which is compatible with open-source Apache JMeter | | |
| 42 | **Selenium IDE** | Apache 2.0 license | selenium.dev/selenium-ide/ |
| | Selenium IDE is an easy-to-use and integrated development environment used by web app developers to record, edit, and debug tests. | | |
| 43 | **Wessbas** | Apache 2.0 license | wessbas.github.io/ |
| | Wessbas is more than a recording tool. First, a system- and tool-agnostic domain-specific language (DSL) allows the layered modeling of workload specifications of session-based systems. Second, instances of this DSL are automatically extracted from recorded session logs of production systems. Third, these instances are transformed into executable workload specifications of load generation tools and model-based performance evaluation tools (Vögele, Hoorn, Schulz, Hasselbring & Krcmar, 2016). | | |

The plethora of software used can be explained partially by the need for solutions better suiting the particular software tested as there is no silver bullet to be found as Kaur & Gupta (2013) argued the testing software was best chosen based on budget and nature of the software that has to be tested. This seem still to be the issue as the testing is becoming fragmented, and the reaction speed of testing is not synchronized to the great velocity of the software development (Ferme and Pautasso, 2017).

# 5. Findings

The literature research found two strong and widely used software, JMeter and HP LoadRunner, which has both evolved to fulfil the needs of users throughout the span of the literature review. Evidence of this was presented in table 5, which presented hits of reference to the software from 2011 – 2019, and proved the hits had a relatively steady count throughout years 2011 – 2016 and 2017 – 2019 in relation to the whole count. A more detailed analysis of what tools actually were used for, shows an interest in testing the tool itself (table 14) for new approaches and solving new issues as API (Application Programming Interface) issues or optimizing the usage for better or completely new approaches for load and stress testing of software.

The results file includes the whole collected data from which the findings and analysis is derived from. It can be found in Appendix A (Appendix A Results.pdf). The document numbers referred to in the results file are documented in appendixes B and C (Appendix B LoadTesting.pdf, Appendix C StressTesting.pdf).

**Table 14**. Distribution of performance testing targeting the testing tool itself

| **Performance testing of tool tested (total)** | **35** |
|---|---|
| Complete solution set presented | 6 |
| Solving API / new issues | 16 |
| Optimizing use / new approach of use | 13 |

A total of 6 complete solutions were presented as capable of fulfilling the whole test scenario, but only two of the presented solutions found by the study are potential contenders of doing so. These contenders are presented later in this chapter. 16 instances had presented and tested new issues or solved API obstacles and 16 was focused on optimizing the tool usage or took a completely new approach to a problem found by scholars and practitioners earlier.

Comparative studies of testing software were documented a total of 13 instances. JMeter and HP LoadRunner were mentioned in the same comparison or description document only two times in the same document. Software performance was tested in 24 documents using a variety of tools. A table of these findings is presented as table 15.

**Table 15**. Distribution of performance testing targeting the testing tool itself

| **Usage** | **Count** |
|---|---|
| Tool comparison and Software under test combined | **37** |
| Comparison / description of tools (including new tools) | 13 |
| Only tool descriptions and comparison, no usage of tool | **5** |
| Used as performance evaluation tool for Software under test | 24 |
| Tool used as a Verification tool (any tool) | **16** |
| New approach of use | **16** |

As five of the documents were descriptive and only described the function of the test software, numbers in table 15 would not add up without taking them into account in the table. As evaluation tools for software change verification and performance change

evaluators, which was documented 24 times, of which JMeter was mentioned 13 times and HP LoadRunner three times.

However, deciding whether the document was dedicated to solely test the performance of the software or evaluate the tested system performance capacity was not clear at all at some instances, as some tests required a totally new approach or solution of how to be able to measure software under test performance. To target this documenting related problem, table 16 illuminates to which extent the most mentioned software were represented, when systems under test and performance tools evaluation numbers were combined.

**Table 16**. Systems under test and performance tools evaluation numbers combined

| **Usage** | **Count** |
|---|---|
| **Total** | **67** |
| Software performance testing (testing tool used as a verification tool) | 16 |
| Software performance testing (new approach of use) | 16 |
| Tool performance (complete solution set presented) | 6 |
| Tool performance (solving API / new issues) | 16 |
| Tool performance (optimizing use / new approach of use) | 13 |
| | |
| **JMeter** being a part of the test or solution | **33** |
| **HP LoadRunner** being a part of the test or solution | **10** |

Reason for the high count of hits is on the account of JMeter due to the Apache open source origin and licensing, which makes it feasible for cost effective and innovative development. HP LoadRunner persist, most likely due to effective response to market changes, a complete package portfolio including all necessary for customer needs and well-organized customer support which compensates the for the pricing.

## 5.1 Research questions and answers

To answer the research questions a total of 148 documents was reviewed and a total of 72 documents matched the research criteria. RQ1 was easily answered by counting hits of usage, as the main tool or usage as comparison tool for other projects was Apache JMeter.

### 5.1.1 RQ1. What is the most common Stress and Load testing software tool used according to literature findings?

The most common tool used found by the literature research was Apache JMeter, which is an open source software under Apache 2.0 license. Key features of Apache JMeter are, according to Sharma, Shetty, Subramanian and Iyer (2016) and Abbas, Sultan, and Bhatti (2017) are that JMeter can run on any operating system as it is built on a Java platform. It can run in distributed mode thus making it scalable. Jmeter is ready to support a large number of different protocols making it nimble such as HTTP, SMTP, POP3, LDAP, JDBC, SOAP and TCP. It has also a lot of pre- and post-processors which are implemented around sampler providing advanced setup, teardown parametrization, and correlation capabilities. Multiple built-in and external listeners help to visualize and analyze performance test results and integration with major build and continuous integration systems are possible. And JMeter is free of cost, which is one of the major advantages.

Problems related to Apache JMeter according to Sharma, Shetty, Subramanian and Iyer, (2016) and Abbas, Sultan, and Bhatti (2017) are that JMeter takes more time on one-time installation and has been recorded to be unstable under huge load. It has no built-in monitoring and script writing might be challenging and time consuming. The benefits of Apache JMeter exceed the problems related to the use as JMeter is widely used, well documented and being free of charge, keeps it attractive to end users and developers.

### 5.1.2 RQ2. Is there a recommended tool or tool set for website testing purposes?

The most used tool for testing websites that included all phases (Virtual User Generator, Controller, Load Generator and Analysis) was HP LoadRunner thus also being the most expensive. (Sharma, Shetty, Subramanian & Iyer, 2016, Abbas, Sultan & Bhatti, 2017.) There is at least one alternative solution, which combines several free and low-cost programs as a composite solution to accomplish web testing service as a whole (Lee, Lin, Lin & You, 2018). Both solutions are presented in the next subchapters as an answer to RQ2.

#### *HP LoadRunner*

Key features of HP LoadRunner are according to LoadRunner (2020) homepage that HP LoadRunner runs on Linux and Windows systems. It has a built in interactive recording and scripting system giving browser-based and native mobile applications the possibility of being tested using the most advanced network behavior and service virtualization in the industry. Simple, elastic, and realistic tests can be ran from multiple geographies and tests can be performed by scaling load testing in the cloud up and down to simulate the demands of business applications. Performance testing can be integrated into your development environment including IDE, continuous integration, and build systems. Application performance bottlenecks can be identified using non-intrusive, real-time performance monitors that leverage application-layer and code-level data for root cause and analytics.

Problems related to HP LoadRunner were identified by Sharma, Shetty, Subramanian and Iyer (2016) and Abbas, Sultan, and Bhatti (2017) to be the price of the software. It has a tendency of occasionally crashing under heavy load and the installation takes a lot of time. As it is a complete system, the controller user interface is complex, and it has some configuration issues across firewalls. HP LoadRunner has rather poor measuring at non-Windows server statistics, which can be counted as a deficiency. Nevertheless, HP LoadRunner was the most referred testing platform, that included all phases of Testing as a Service required for a complete business scenario.

#### *Composite solution*

Lee, Lin, Lin and You (2018) documented the first phase of their composite solution in 2016 and presented a second, more sophisticated, version of their solution in 2018 (Lee, Lin, Lin & You, 2016, Lee et al., 2018). The key features of the proposed composite solution (Lee et al., 2018) are that adapters have been devised to bridge the gap between the inputs and outputs of six web testing software selected for the solution which are Badboy, JMeter, Cacti, Xdebug, Selenium IDE, and Selenium WebDriver. The solution has been developed for the automated composition of the web testing software to work as a complete composite system based on a continuous integration framework presented by Jenkins using Hudson APIs, that can be globally shared among plugins (Jenkins,

2020). The composite web testing service can be delivered via email using two primary components for easy access. The composite test frame presented has promising prospects as most of the tools are free of cost as presented in chapter 4.1.

# 6. Discussion, limitations and implications

The literature review reviled a multitude of tools used for web testing purposes, unfortunately leaving some promising candidates unmentioned due to missing notations in selected documents. The main commercial product presented in Chapter 5 has kept the same hit rate throughout the review session as a testing tool for websites and as comparison for other web testing tools. As a simple solution, the commercial market leader in complete solutions is always an option, if time is of essence and finance is not a problem. However, the need for cost effective web software testing tools for specialised web software testing companies and other software developing companies, is imminent. The composite solution presented in Chapter 5 could be a promising frame, as the main problem with isolated tools is how they communicate with each other when creating composite systems to speed up and make web testing services faster and more cost effective.

As a limitation to this study, the exclusion of grey literature material should be mentioned as a restricting factor as well as the excluding documents based on paying for use. The grey literature material option usage option came in a late phase of the study and was not applied due to excessive workload as the whole inclusion / exclusion process as well as the downloading and review would have to be started from scratch. The study, however, recognises the value of such study and strongly recommends future studies to apply such an approach to ensure more and possibly different aspects of the testing tool environment. The exclusion of documents needing financial involvement is due to the nature of the work being done by single person and not someone contracted by a company to ensure access to all available material.

As implication to future work, the presented Jenkins continuous integration framework with Hudson APIs (Application Programming Interface), is most certainly worth testing with other tools probably already used in companies doing testing services. Familiar tools make the use of improved test solutions less unattractive, saves time, effort and keeps the results comparable to previous test sessions making tool based and result interpretive bias smaller and overall effort more manageable. As there were a lot of software described for different test functions, bridging the gaps between different testing stages with application programming interfaces to avoid laborious manual handling and making the process faster and more efficient could be a way of making Software Testing as a Service a more gainful business giving the company an edge to even improve their productivity and enlarge test setup scope.

As time is one of the most precious and costly valuable to companies making business, experimenting with new ideas is not always feasible, it opens an opportunity for future scholars to investigate new possibilities using API’s with close relationship to companies doing business in the software field of stress and load testing. The need for such skills will probably grow in the future as software to be tested is expanding in an explosive rate and speed is the key issue of modern software development, regardless of whether the professionals testing the software are inside the software company or doing the testing as a business.

# References


Abbas R., Sultan Z, & Bhatti S. N., (2017). *Comparative analysis of automated load testing tools: Apache jmeter microsoft visual studio (tfs) loadrunner siege*. Communication Technologies (ComTech) 2017 International Conference on, pp. 39-44.

Ahmad A, Brereton P & Andras P (2017). *A systematic mapping study of empirical studies on software cloud testing methods.* In Proc. IEEE International Conference on Software Quality, Reliability and Security Companion, July 2017, pp.555-562.

Apache.org (2020). *https://www.apache.org/licenses/LICENSE-2.0.* Retrieved from *www.apache.org/licenses* 15.12.2020

Arlitt M., Krishnamurthy D. & Rolia J. (2001). *Characterizing the Scalability of a Large Web-based Shopping System*. ACM Trans. on Internet Technology, v 1, pp. 44-69.

Avritzer A., Kondek J., Liu D. & Weyuker E. J. (2002). *Software performance testing based on workload characterization.* Published in Proc. WOSP'2002, Rome, pp. 17-24.

Barber S. (2004a). *Beyond performance testing.* Parts 1-14, IBM DeveloperWorks, Rational Technical Library, 2004, www-128.ibm.com/developerworks/rational/library/4169.html

Barber S. (2004b). *Creating Effective Load Models for Performance Testing with Incomplete Empirical Data.* Published in Proc. 6th IEEE Int. Workshop on Web Site Evolution, pp. 51-59.

Barber S. (2006). *Tester PI: Performance Investigator. Better Software*. March 2006, pp 20 – 25.

Bezemer C-P, Eismann S, Ferme V, Grohmann J, Heinrich R, Jamshidi P, Shang W, van Hoorn A, Villavicencio M, Walter J & Willnecker F. (2018). *How is Performance Addressed in DevOps?* A Survey on Industrial Practices. In Proc. ICPE 2019

BlazeMeter.com (2019) https://blazemeter.com/pricing/

Cardoso J., Sheth A., Miller J., Arnold J. & Kochut K. (2004). *Quality of service for workflows and web service processes.* Web Semant.: Sci. Serv. Agents World Wide Web, 1 (3), pp. 281-308

Dhiauddin M., Suffiani M. & Fahrurazi F.R. (2012). *Performance Testing: Analyzing Differences of Response Time between Performance Testing Tools.* Published in proceeding of International Conference on Computer & Information Science (ICCIS) 2012.

Ferme V. & Pautasso C. (2017). *Towards Holistic Continuous Software Performance Assessment.* In Proc. QUDOS@ICPE 2017. 159–164

Ferme V. & Pautasso C. (2018). *A Declarative Approach for Performance Tests Execution in Continuous Software Development Environments*. Published in ICPE '18, pp. 261-272.

Field T., Chatley R. & Wei R. (2018). *Software Performance in Virtual Time.* WOSP-C Workshop ICPE'18 Companion, April 9–13, 2018, Berlin, Germany

G2.com (2020). https://www.g2.com/categories/load-testing

Gan, Z. -., Wei, D. -., & Varadharajan, V. (2005). *Evaluating the performance and scalability of web application systems.* Paper presented at the Proceedings - 3rd

International Conference on Information Technology and Applications, ICITA 2005, I 111-114. Retrieved from www.scopus.com 19.6.2019

GNU.org (2020). https://www.gnu.org/licenses/licenses.html. Retrieved 15.12.2020

Hevner, A., March, R., Salvatore, T., Park, J. & Ram, S. (2004). *Design Science in Information Systems Research.* MIS Quarterly, 28(1), pp. 75–105.

Hassan M. M., Afzal W., Lindstrom B., Shah S. M. A., Andler S. F. & Blom M. (2016). *Testability and software performance: A systematic mapping study.* Presented in Proceedings of the 31st Annual ACM Symposium on Applied Computing. ACM, pp. 1566–1569

Hossain, M. S. (2012). *Performance evaluation web testing for ecommerce web sites.* Paper presented at the 2012 International Conference on Informatics, Electronics and Vision, ICIEV 2012, 842-846. doi:10.1109/ICIEV.2012.6317531 Retrieved from www.scopus.com 19.6.2019

Hossain. N. I. (2013). *A comparative evaluation of approaches for Web Application Testing.* International Journal of Soft Computing and Software Engineering [JSCSE], 3(3):333–341

HP Loadrunner (2020). microfocus.com/en-us/products/loadrunner-professional/ retrieved 15.12.2020

Jenking (2020). *https://www.jenkins.io/doc/developer/architecture/web/*. Retreived 15.12.2020

Kaur Dr. H., Gupta G. (2013). *Comparative Study of automation testing tools:selenium, quick test professional and testcomplete.* International Journal of Engineering Research and Application, vol. 3, no. 5, pp. 1739-1743.

Khan F. & Khan M.E. (2012). *A comparative study of white box, black box, grey box testing techniques.* (IJACSA), Vol. 3, No.6.

Khan R. & Amjad M. (2016). *Smoke testing of web application based on ALM tool*. 2016 International Conference on Computing, Communication and Automation (ICCCA), Noida, pp. 862-866.

Kircher, M (producer) & Vodde, B (guest) (2011) January 5. *Episode 170: Large Agile Software Development with Bas Vodde* [Audio podcast]. Retrieved 12.7.2018 from http://www.seradio.net/2011/01/episode-170-large-agile-software-development-with-bas-vodde/

Kitchenham B., Charters S., (2007). *Guidelines for Performing Systematic Literature Reviews in Software Engineering.* Technical Report EBSE2007-01, School of Computer Science and Mathematics, Keele University

Lee S.-J., Lin Y.-C., Lin K.-H., & You J.-L., (2016). *Composing and Delivering Heterogeneous Web Testing as a Composite Web Testing Service.* International Computer Symposium (ICS), pp. 605-610, December 15-17

Lee S.-J., Lin Y.-C., Lin K.-H., & You J.-L., (2018). *A system for composing and delivering heterogeneous web testing software as a composite web testing service*. J. Inf. Sci. Eng. 34 (3), 631–648

Li P., Shi D. & Li J. (2013). *Performance test and bottle analysis based on scientific research management platform.* 10th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), pp. 218–221.

Menascé D. A. (2002). *Load Testing of Web Sites.* Article in IEEE Internet Computing, August 2002

Menascé D.A. (2002). *Software, performance, or engineering?* Proceedings of the 3rd International Workshop on Software and Performance (WOSP'02), ACM Press, Rome, Italy, pp. 239-242

Nguyen T. (2012). *Using control charts for detecting and understanding performance regressions in large software.* Proceedings - IEEE 5th International Conference on Software Testing, Verification and Validation, ICST 2012, pages 491–494.

Opensource.org (2020). https://opensource.org/licenses/MIT. Retrieved 15.12.2020

Paz, S., Bernardino, J. (2017). *Comparative analysis of web platform assessment tools.* In: Proceedings of the 13th International Conference on Web Information Systems and Technologies - Volume 1: WEBIST, pp. 116–125 (2017). ISBN 978-989-758-246-2. https://doi.org/10.5220/0006308101160125

Petersen K., Feldt R., Mujtaba S., Mattsson M. (2008). *Systematic Mapping Studies in Software Engineering.* 12th International Conference on Evaluation and Assessment in Software Engineering, June 2008, pp. 71-80.

Poston R.M & Sexton M.P., (1992). *Evaluating and selecting testing tools.* Software, IEEE. IEEE 9, 3, 33-42.

Pradeep S., Sharma Y. K. (2019). *A Pragmatic Evaluation of Stress and Performance Testing Technologies for Web Based Applications*. 2019 Amity International Conference on Artificial Intelligence (AICAI), pp. 399-403.

Prove Oy (2019). *DSR Tool Requirements.* Interview of the responsible persons for Stress and Load testing in the company in a DSR-course

Raj_esh_0201, (2008). *Performance Test Tool Comparison*. https://www.scribd.com/document/27765939/Performance-Test-Tools-Comparison. Accessed on 6.6.2019

Ricca F. & Tonella P. (2001). *Analysis and Testing of Web Applications.* Proc. of the IEEE International Conference on Software Engineering, May 2001.

Shariff SM, Li H, Bezemer C, Hassan AE, Nguyen THD, Flora P (2019). *Improving the testing efficiency of selenium-based load tests.* In: 2019 IEEE/ACM 14th International Workshop on Automation of Software Test (AST), pp 14–20

Sharma M., Shetty A, Subramanian S. & Iyer V. s., (2016). *A Comparative Study on Load Testing Tools*. *Int. Journal of Innovative Research in Computer and Communication Engineering*, vol. 4, no. 2, February 2016

Smith C. U. (1981). *Increasing Information Systems Productivity by Software Performance Engineering.* Proc. CMG XII International Conference, December 1981.

Smith C. (1990). *Performance Engineering of Software Systems*. Addison-Wesley

Smith C.U. (2015). *Software Performance Engineering Then and Now: A Position Paper.* Proceedings of the 2015 Workshop on Challenges in Performance Methods for Software Development

Softwaretestinghelp.com (2020). https://www.softwaretestinghelp.com/performance-testing-tools-load-testing-tools/

*Software Testing Fundamentals Black-box testing, White-box testing, grey-box testing.* Retrieved February, 3, 2017, from http://softwaretestingfundamentals.com/black-box-testing/

Stefan P., Horky V., Bulej L. & Tuma P. (2017). *Unit Testing Performance in Java Projects: Are We There Yet?.* In Proceedings of the 8th ACM/SPEC on International

Conference on Performance Engineering (ICPE '17). ACM, New York, NY, USA, 401–412. https://doi.org/10.1145/3030207.3030226

Technopedia (2020). https://www.techopedia.com/definition/4245/commercial-software. Retrieved 15.12.2020

Vail C. (2005) *Stress, load, volume, performance, benchmark and base line testing tool evaluation and comparison.*
Source: http://www.vcaa.com/tools/loadtesttoolevaluationchart-023.pdf. Accessed on 6.6.2019

Vögele, C., van Hoorn, A., Schulz, E., Hasselbring, W., Krcmar, H.: *WESSBAS: extraction of probabilistic workload specifications for load testing and performance prediction–a model-driven approach for session-based application systems*. J. Softw. Syst. Model. (2016).

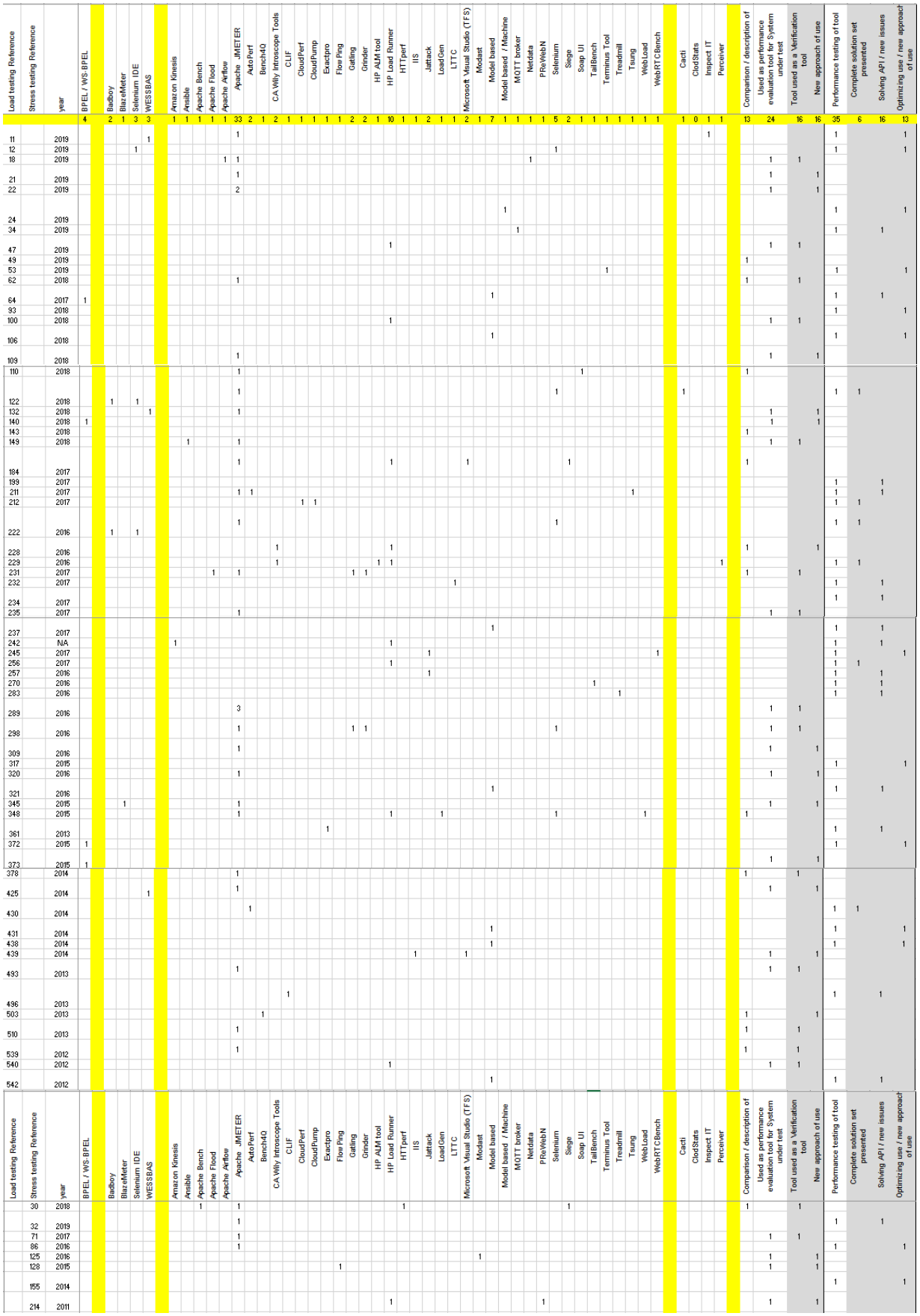
Woodside M., Franks G. & Petriu D. (2014). *The Future of Software Performance Engineering.* Carleton University, Ottawa, Canada.
{cmw | greg | petriu}@sce.carleton.ca

Yan M., Sun H., Wang X. & Liu X. (2012). *WS-TaaS: A Testing as a Service Platform for Web Service Load Testing.* IEEE 18th International Conference on Parallel and Distributed Systems, pp. 456-463

# 7. Appendixes

Appendix A. Results.pdf
Appendix B. LoadTesting.pdf
Appendix C. StressTesting.pdf

# Appendix A. Results



# Appendix B. Load Testing

11, "reducing the maintenance effort for parameterization of representative load tests using annotations","Schulz H.","Software Testing Verification and Reliability",2020-01-01,"Copyright © 2019 John Wiley & Sons, Ltd
12, "investigation on reliability estimation of loosely coupled software as a service execution using clustered and non-clustered web server","Bora A.","International Journal of Engineering, Transactions A: Basics",2020-01-01,"© 2020 Materials and Energy Research Center. All rights reserved.
18, "industrial track: architecting railway kpis data processing with big data technologies","Suleykin A.","Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019",2019-12-01,"© 2019 IEEE.
21, "an efficient performance testing of web services","Hasnain M.","Proceedings - 22nd International Multitopic Conference, INMIC 2019",2019-11-01,"© 2019 IEEE.
22, "an experience report of generating load tests using log-recovered workloads at varying granularities of user behaviour","Chen J.","Proceedings - 2019 34th IEEE/ACM International Conference on Automated Software Engineering, ASE 2019",2019-11-01,"© 2019 IEEE.
24, "on the utility of machine learning for service capacity management of enterprise applications","Muller H.","Proceedings - 15th International Conference on Signal Image Technology and Internet Based Systems, SISITS 2019",2019-11-01,"© 2019 IEEE.
34, "model-based load testing in the iot system","Matic M.","IEEE International Conference on Consumer Electronics - Berlin, ICCE-Berlin",2019-09-01,"© 2019 IEEE.
47, "improving the testing efficiency of selenium-based load tests","Shariff S.M.","Proceedings - 2019 IEEE/ACM 14th International Workshop on Automation of Software Test, AST 2019",2019-05-01,"© 2019 IEEE
49, "a pragmatic evaluation of stress and performance testing technologies for web based applications","Pradeep S.","Proceedings - 2019 Amity International Conference on Artificial Intelligence, AICAI 2019",2019-04-26,"© 2019 IEEE.
53, "performance modeling for cloud microservice applications","Jindal A.","ICPE 2019 - Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering",2019-04-04,"© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.
62, "denial of service attack generator in apache jmeter","Grabovsky S.","International Congress on Ultra Modern Telecommunications and Control Systems and Workshops",2019-01-31,"© 2018 IEEE.
64, "a resource-aware model-based framework for load testing of ws-bpel compositions","Krichen M.","Lecture Notes in Business Information Processing",2019-01-01,"© 2019, Springer Nature Switzerland AG.
93, "ubiquitous application testing on cloud","Khan A.","2018 International Conference on Smart Computing and Electronic Enterprise, ICSCEE 2018",2018-11-15,"© 2018 IEEE.
100, "research on automatic test of web system based on loadrunner","Linling Q.","13th International Conference on Computer Science and Education, ICCSE 2018",2018-09-19,"© 2018 IEEE.
106, "model-driven method for performance testing","Javed Z.","2018 7th International Conference on Reliability, Infocom Technologies and

Optimization: Trends and Future Directions, ICRITO 2018",2018-08-01,"© 2018 IEEE.
109, "continuous integration and continuous delivery pipeline automation for agile software project management","Arachchi S.","MERCon 2018 - 4th International Multidisciplinary Moratuwa Engineering Research Conference",2018-07-27,"© 2018 IEEE.
110, "performance and load testing: tools and challenges","Lenka R.K.","2018 International Conference on Recent Innovations in Electrical, Electronics and Communication Engineering, ICRIEECE 2018",2018-07-01,"© 2018 IEEE.
122, "a system for composing and delivering heterogeneous web testing software as a composite web testing service<sup>*</sup>","Lee S.","Journal of Information Science and Engineering",2018-05-01,"© 2018 Institute of Information Science. All rights reserved.
132, "exploiting load testing and profiling for performance antipattern detection","Trubiani C.","Information and Software Technology",2018-03-01,"© 2017 Elsevier B.V.
140, "distributed and resource-aware load testing of ws-bpel compositions","Maâlej A.","ICEIS 2018 - Proceedings of the 20th International Conference on Enterprise Information Systems",2018-01-01,"© 2018 by SCITEPRESS - Science and Technology Publications, Lda.
143, "cloud-based test tools: a brief comparative view","Kilinç N.","Cybernetics and Information Technologies",2018-01-01,"© 2018 Bulgarian Academy of Sciences.
149, "svload: an automated test-driven architecture for load testing in cloud systems","Noor J.","2018 IEEE Global Communications Conference, GLOBECOM 2018 - Proceedings",2018-01-01,"© 2018 IEEE.
184, "comparative analysis of automated load testing tools: apache jmeter, microsoft visual studio (tfs), loadrunner, siege","Abbas R.","International Conference on Communication Technologies, ComTech 2017",2017-10-11,"© 2017 IEEE.
199, "analytics-driven load testing: an industrial experience report on load testing of large-scale systems","Chen T.","Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track, ICSE-SEIP 2017",2017-06-30,"© 2017 IEEE.
211, "autoperf: automated load testing and resource usage profiling of multi-tier internet applications","Apte V.","ICPE 2017 - Proceedings of the 2017 ACM/SPEC International Conference on Performance Engineering",2017-04-17,"© 2017 ACM.
212, "cloudperf: a performance test framework for distributed and dynamic multi-tenant environments","Michael N.","ICPE 2017 - Proceedings of the 2017 ACM/SPEC International Conference on Performance Engineering",2017-04-17,"© 2017 Copyright held by the owner/author(s).
222, "composing and delivering heterogeneous web testing software as a composite web testing service","Lee S.","Proceedings - 2016 International Computer Symposium, ICS 2016",2017-02-16,"© 2016 IEEE.
228, "web application's performance testing using hp loadrunner and ca wily introscope tools","Khan R.","Proceeding - IEEE International Conference on Computing, Communication and Automation, ICCCA 2016",2017-01-10,"© 2016 IEEE.
229, "smoke testing of web application based on alm tool","Khan R.","Proceeding - IEEE International Conference on Computing, Communication and Automation, ICCCA 2016",2017-01-10,"© 2016 IEEE.

231, "comparative analysis of web platform assessment tools","Paz S.","WEBIST 2017 - Proceedings of the 13th International Conference on Web Information Systems and Technologies",2017-01-01,"Copyright © 2017 by SCITEPRESS - Science and Technology Publications, Lda. All rights reserved.
232, "lttc: a load testing tool for cloud","Geetha Devasena M.","Advances in Intelligent Systems and Computing",2017-01-01,"© Springer Nature Singapore Pte Ltd. 2017.
234, "simulating user interactions: a model and tool for semi-realistic load testing of social app backend web services","Brune P.","WEBIST 2017 - Proceedings of the 13th International Conference on Web Information Systems and Technologies",2017-01-01,"Copyright © 2017 by SCITEPRESS - Science and Technology Publications, Lda. All rights reserved.
235, "improving performance of web application approaches using connection pooling","Gupta K.","Proceedings of the International Conference on Electronics, Communication and Aerospace Technology, ICECA 2017",2017-01-01,"© 2017 IEEE.
237, "checking response-time properties of web-service applications under stochastic user profiles","Schumi R.","Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)",2017-01-01,"© 2017, IFIP International Federation for Information Processing.
242, "performance testing approach to aws kinesis stream and loadrunner","Dasgupta D.","imPACt 2017 - Internet, Mobile, Performance and Capacity, Cloud and Technology",2017-01-01,"© imPACt 2017 - Internet, Mobile, Performance and Capacity, Cloud and Technology. All rights reserved.
245, "webrtc testing: challenges and practical solutions","Garcia B.","IEEE Communications Standards Magazine",2017-01-01,"© 2017 IEEE.
256, "some aspects of qos for high performance of service- oriented computing in load balancing cluster-based web server","Bora A.","Handbook of Research on Recent Developments in Intelligent Communication Application",2016-12-12,"© 2017, IGI Global.
257, "jattack: a webrtc load testing tool","Amirante A.","2016 Principles, Systems and Applications of IP Telecommunications, IPTComm 2016",2016-12-09,"© 2016 IEEE.
270, "tailbench: a benchmark suite and evaluation methodology for latency-critical applications","Kasture H.","Proceedings of the 2016 IEEE International Symposium on Workload Characterization, IISWC 2016",2016-10-03,"© 2016 IEEE.
283, "treadmill: attributing the source of tail latency through precise load testing and statistical inference","Zhang Y.","Proceedings - 2016 43rd International Symposium on Computer Architecture, ISCA 2016",2016-08-24,"© 2016 IEEE.
289, "a framework to evaluate the effectiveness of different load testing analysis techniques","Gao R.","Proceedings - 2016 IEEE International Conference on Software Testing, Verification and Validation, ICST 2016",2016-07-18,"© 2016 IEEE.
298, "systematic scalability assessment for feature oriented multi-tenant services","Preuveneers D.","Journal of Systems and Software",2016-06-01,"© 2015 Elsevier Inc. All rights reserved.
309, "cloud-based load testing method for web services with vms management","Shojaee A.","Conference Proceedings of 2015 2nd International Conference on Knowledge-Based Engineering and Innovation, KBEI 2015",2016-03-17,"© 2015 IEEE.

317, "w taas: an architecture of website analysis in a cloud environment","Mungekar S.","Proceedings on 2015 1st International Conference on Next Generation Computing Technologies, NGCT 2015",2016-01-07,"© 2015 IEEE.
319, "a framework for composing heterogeneous service tools involved in load testing lifecycle","Lee S.","Applied System Innovation - Proceedings of the International Conference on Applied System Innovation, ICASI 2015",2016-01-01,"© 2016 Taylor & Francis Group.
320, "how to emulate web traffic using standard load testing tools","Brady J.","imPACt 2016 - Internet, Mobile, Performance and Capacity, Cloud and Technology",2016-01-01
321, "a model based approach to combine load and functional tests for service oriented architectures","Maalej A.","CEUR Workshop Proceedings",2016-01-01
345, "automatic performance analysis of cloud based load testing of web-application &amp its comparison with traditional load testing","Arslan M.","Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS",2015-11-25,"© 2015 IEEE.
348, "a survey on load testing of large-scale software systems","Jiang Z.","IEEE Transactions on Software Engineering",2015-11-01,"© 2015 IEEE.
361, "high performance load generator for automated trading systems testing","Guriev D.K.","Proceedings - 2013 Tools and Methods of Program Analysis, TMPA 2013",2015-07-21,"© 2013 Exactpro Systems, LLC.
372, "a comparative evaluation of state-of-the-art load and stress testing approaches","Maâlej A.","International Journal of Computer Applications in Technology",2015-01-01,"© 2015 Inderscience Enterprises Ltd.
373, "study on the limitations of ws-bpel compositions under load conditions","Maâlej A.","Computer Journal",2015-01-01,"© 2014 The British Computer Society. All rights reserved.
378, "delivering web service load testing as a service with a global cloud","Yan M.","Concurrency Computation",2015-01-01,"© 2014 John Wiley & Sons, Ltd.
425, "automatic extraction of probabilistic workload specifications for load testing session-based application systems","Van Hoorn A.","Proceedings of the 8th International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS 2014",2014-01-01,"© Copyright 2015 ICST.
430, "perfcenterlite: extrapolating load test results for performance prediction of multi-tier applications","Apte V.","Proceedings of the 8th International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS 2014",2014-01-01,"© Copyright 2015 ICST.
431, "ltf: a model-based load testing framework for web applications","Zhou J.","Proceedings - International Conference on Quality Software",2014-01-01,"© 2014 IEEE.
438, "continuous validation of load test suites","Syer M.","ICPE 2014 - Proceedings of the 5th ACM/SPEC International Conference on Performance Engineering",2014-01-01
439, "performance and load testing of cloud vs classic server platforms: (case study: social network application)","Cico O.","Proceedings - 2014 3rd Mediterranean Conference on Embedded Computing, MECO 2014 - Including ECyPS 2014",2014-01-01
493, "model-based performance testing of web services using probabilistic timed automata","Abbors F.","WEBIST 2013 - Proceedings of the 9th International Conference on Web Information Systems and Technologies",2013-11-11

496, "a scalable benchmark as a service platform","Tchana A.","Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)",2013-10-09
503, "migrating load testing to the cloud: a case study","Gao Q.","Proceedings - 2013 IEEE 7th International Symposium on Service-Oriented System Engineering, SOSE 2013",2013-08-05
510, "model-based performance testing in the cloud using the mbpet tool","Abbors F.","ICPE 2013 - Proceedings of the 2013 ACM/SPEC International Conference on Performance Engineering",2013-05-30
539, "ws-taas: a testing as a service platform for web service load testing","Yan M.","Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS",2012-12-01
540, "research of load testing and result application based on loadrunner","Zhang H.","Proceedings of the 2012 National Conference on Information Technology and Computer Science, CITCS 2012",2012-12-01
542, "perfext: performance extrapolation tool","Duttagupta S.","Proceedings of International Conference on Computational Intelligence, Modelling and Simulation",2012-12-01

# Appendix C. Stress Testing

30, "denial of service attack generator in apache jmeter","Grabovsky S.","International Congress on Ultra Modern Telecommunications and Control Systems and Workshops",2019-01-31,"© 2018 IEEE.
32, "dynamojm: a jmeter tool for performance testing using dynamic workload adaptation","Huerta-Guevara O.","Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)",2019-01-01,"© 2019, IFIP International Federation for Information Processing.
71, "a multi-objective metaheuristic approach to search-based stress testing","Gois N.","IEEE CIT 2017 - 17th IEEE International Conference on Computer and Information Technology",2017-09-11,"© 2017 IEEE.
86, "improving stress search based testing using a hybrid metaheuristic approach","Bernardo Gois F.","Proceedings of the 2016 42nd Latin American Computing Conference, CLEI 2016",2017-01-25,"© 2016 IEEE.
125, "“overloaded! ” — a model-based approach to database stress testing","Meira J.","Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)",2016-01-01,"© Springer International Publishing Switzerland 2016.
128, "flowping - the new tool for throughput and stress testing","Vondrous O.","Advances in Electrical and Electronic Engineering",2015-12-01,"© 2015 ADVANCES IN ELECTRICAL AND ELECTRONIC ENGINEERING.
155, "implementing taas-based stress testing by mapreduce computing model","Hwang G.","Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS",2014-01-01,"© 2014 IEEE.
214, "investigation on performance testing and evaluation of prewebn: a java technique for implementing web application","Kalita M.","IET Software",2011-10-01