Czech University of Life Sciences Prague

Faculty of Economics and Management

Department of Information Technologies

Bachelor Thesis

Software quality and testing

Mukhiddinova Gulshan

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

BACHELOR THESIS ASSIGNMENT

Gulshan Mukhiddinova

Systems Engineering and Informatics
Informatics

Thesis title

Software quality and testing

Objectives of thesis

The main objective of the thesis is to evaluate current software testing tools.
The partial objective are such as following:
- To make a comprehensive literature review of the current trends and tools for software quality evaluation and testing.
- To compare and evaluate software testing tools based on given criteria.
- To select an optimal software testing tool and testing methodology in a case study and formulate conclusion.

Methodology

The methodology of this thesis will be based on desk research of available scholar and professional literature. Based on the findings in the literature review, research questions will be set out and addressed in the practical part. Then, a case study will be developed to find and select an optimal software testing tool and testing methodology for a given purpose. In the thesis, the scientific methods such as analysis, synthesis, comparison, induction and deduction will be used. Following on the theoretical findings and results of the case study, final conclusion will be made.

The proposed extent of the thesis

30 – 40 pages

Keywords

Software, testing, quality assurance, testing methodology, validation, ISO.

Recommended information sources

APRIL, Alain; LAPORTE, Claude Y. Software Quality Assurance. John Wiley & Sons, 2018.

CHEMUTURI, Murali. Mastering software quality assurance: best practices, tools and techniques for software developers. J. Ross Publishing, 2010.

ISO, I. S. O. Iec25010: 2011 systems and software engineering–systems and software quality requirements and evaluation (square)–system and software quality models. International Organization for Standardization, 2011, 34: 2910.

SINGH, Inderjeet; TARIKA, Bindia. Comparative analysis of open source automated software testing tools: Selenium, sikuli and watir. International Journal of Information & Computation Technology, 2014, 4.15: 1507-1518.

Expected date of thesis defence

2020/21 SS – FEM

The Bachelor Thesis Supervisor

Ing. Miloš Ulman, Ph.D.

Supervising department

Department of Information Technologies

Electronic approval: 11. 9. 2018

Ing. Jiří Vaněk, Ph.D.

Head of department

Electronic approval: 19. 10. 2018

Ing. Martin Pelikán, Ph.D.

Dean

Prague on 12. 03. 2021

Declaration

I declare that I have worked on my bachelor thesis titled "Software quality and testing" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the bachelor thesis, I declare that the thesis does not break copyrights of any other person.

In Prague on 08.01.2021 _____

Acknowledgement

I would like to thank my supervisor Ing. Ulman Milos for his valuable advice, suggestion, and support during my on this thesis. Also, I would like to thank my parents for supporting my university studies.

Software quality and testing

Summary

The bachelor thesis attempts to explore and provide an overview of automated software testing tools. The main objective of the thesis is to recommend best software automation testing tool for start-ups and developers. The literature review thesis deals with the methodology of testing, and software development life cycle. In the practical part the author compared tools such as Selenium, Watir, TestComplete, QTP. The comparison was performed by multi-criteria analysis based on selected criteria and set weights. The analyzed criteria were initial release, test development platform, application under test, scripting languages, browser support, ease of installation and use, framework, continuous integrations, product support, and cost to generate meaningful result. The tool which got more score used for testing the web site.

Keywords: Testing tools, testing, automated testing, testing methodologies, manual testing, QTP, Selenium, Watir, TestComplete, SDLC.

Software quality and testing

Souhrn

Bakalářská práce se pokouší prozkoumat a poskytnout přehled automatizovaných nástrojů pro testování softwaru. Hlavním cílem práce je doporučit nejlepší nástroj pro testování automatizace softwaru pro začínající a vývojáře. Literární přehledová práce se zabývá metodikou testování a životním cyklem vývoje softwaru. V praktické části autor porovnával nástroje jako Selenium, Watir, TestComplete, QTP. Porovnání bylo provedeno multikriteriální analýzou na základě zvolených kritérií a stanovených hmotností. Analyzovaná kritéria byla počáteční vydání, platforma pro vývoj testů, testovaná aplikace, skriptovací jazyky, podpora prohlížečů, snadná instalace a použití, framework, nepřetržitá integrace, podpora produktů a náklady za účelem vytvoření smysluplného výsledku. Nástroj, který získal více bodů, se použil pro testování webu.

Klíčová slova: Testovací nástroje, testování, automatizované testování, testovací metodiky, ruční testování, QTP, selen, Watir, TestComplete, SDLC.

Table of content

List of tables

List of pictures

1 Introduction

The main peak of interest in software testing came in the nineties in the United States. The rapid development of computer-aided software development and networking technologies has led to an increase in production in the software market. Increased competition between software manufacturers required increased attention to product quality. As the product range expanded and prices became more affordable, consumers began to pay more attention to the quality of software. Currently, almost all areas of life are subject to computerization. Not only are computers used in everyday life for ordinary purposes, but they are also needed when it comes to much more significant areas such as medicine, transportation, construction, security, and many others. The software of today's computers is represented by millions of programs: from primitive games to ultra-sophisticated scientific programs. Thus, the issue of software quality becomes especially important, since it is not only a matter of comfort, but also safety.

Since data processing affects our life more and more, computer errors can have consequences such as causing material damage, breach of secrecy and a lot of others, now. Many software developers pay attention to "how to look a product" and "what it does." The program does not do what its end-user reasonably expects it to do. Day after day developers is creating so much software that's why every software’s must be qualitative and tested.

Software is a collection of programs executed by a computer system. The software cannot exist separately from the computer system. Where to use a computer, determines the software created for it. The computer itself will not do anything.

Quality assurance is an integral part of the production. The quality of the software product is characterized by a set of properties that determine how the product is "good" in terms of stakeholders, such as the product customer, sponsor, end-user, product developers and testers, support engineers, marketing, training, and sales staff. Each of the participants may have different ideas about the product and how good or bad it is, that is, how high the quality of the product is. Thus, setting the task of ensuring the quality of the product results in the task of identifying stakeholders, their quality criteria and then finding the optimal solution that meets these criteria. Testing is one of the most established ways to ensure the

quality of software development and is included in the set of effective tools of a modern software quality assurance system.

Testing plays a vital role in developing quality software. Software testing - checking the correspondence between the actual and expected behavior of the program, performed on a finite set of tests selected in a certain way. Testing is one of the quality control techniques, which includes activities for the planning of work, design of tests, implementation of testing and analysis of the results.

There are a lot of different automation testing tools on the market. Success on the project depends on choosing right tool for the certain company. There are a lot of different automation testing tools on the market. Right tool can save company's, developer's budget, time, and sources.

2 Objectives and Methodology

2.1 Objectives

The main objective of the thesis is to evaluate current software testing. The thesis will investigate and consider problems that arise during software development. The partial objective of this is:

Firstly, to make a comprehensive literature review of current trends in software quality assurance, software verification and validation. Moreover, the software quality process and major software testing methodologies will be introduced and evaluated from the perspective of the research.

Secondly, the most used software testing tools will be evaluated and compared based on selected criteria considering web application development.

Thirdly, based on a case study an optimal software testing tool and testing methodology will be proposed for software developer and conclusion will be formulated.

2.2 Methodology

The methodology of this thesis will be based on desk research of available scholar and professional literature. Based on the findings in the literature review, research questions will be set out and addressed in the practical part. Then, a case study will be developed to find and select an optimal software testing tool and testing methodology for a given purpose. In the thesis, the scientific methods such as analysis, synthesis, comparison, induction, and deduction will be used. Following on the theoretical findings and results of the case study, conclusion will be made.

3 Literature Review

Author of the section will explain in simple word how software testing works from its inside and will go through advantages of automating software tools and its stages. The literature review shows software development, types, difference of testing, and tools which use for testing. Every company or individual can choose tools which available on the market which fit to them. There are a lot of tools with its features. Success in any project depends on to choose right tool for automation. But not all companies, startups and individuals can afford automation testing tool which cost more than their incomes. In this section we will see popular automating commercial and open-source software tools which available in the market. Every section has a small conclusion regarding the experience and obtained knowledge from the reviewed literature.

Software testing is a method to evaluate a software application's functionality to determine whether the software produced meets the stated specifications and to locate the defects to ensure that the product is free of defects to produce the product of quality. It is one of the quality control techniques (Quality Control), which includes planning, compiling tests, performing testing and analysis of the results. Software testing includes some different actions like analyzing, planning, test case development, writing test reports.

During SDLC (Software Development Life Cycle) testers try to find bugs, defects, errors. The period of testing, the software can occur an inconsistency in the actual result of the program with the expected result. The defects can occur not only in the software code, but also in any documentation, in architecture and design, in the settings of the tested application or test environment.

Depending on the degree of involvement of the tester in the process there are two main types of testing: manual and automated. For manual testing, the tester opens in the checked page in the browser and performs actions there as it would act site user. He constantly monitors correctness work of the site's functionality, for the convenience of working with the interface and for other factors. Each independent sequence of actions the tester and the checks performed by him in the process is called test case 1or a test script. For example, this could be scenario: "Go to the registration page, fill in the fields" login, “email" “Password”, “first name” and “last name”, click on the “register” button and make sure we get to the created profile page. "Automated testing consists in writing auto tests- programs tailored to the implementation of those originally laid down in them test cases and only them. Thus, by running the test, the tester can immediately

get information about whether they were detected during execution given error scenario. In case of a negative answer, the need for manual execution of the test case disappears. Being a good example technology push 2 strategies", This approach at its initial stage quite expensive, because the development of each auto test is usually it takes incomparably more time than its one-time execution manually by a tester. However, over time, using automated testing is beginning to bring tangible benefits, allowing you not to think about a scrupulous check covered by auto tests functionality and instantly receiving notifications in case of breakdowns virtually no cost. This saves a lot of testers time, and hence the company's money. Disadvantages of test automation is that it is effectively applicable only to relatively simple scripts and checks. Automation of complex multi-step scenarios can take an unacceptably long time due to unexpected nuances or technical problems. Also, a lot of time can go to support tests if necessary, often add to them changes following changes in the tested functionality. (1)

3.1 Software Development Life Cycle

SDLC stands for Software Development Life Cycle. Software industry uses the SDLC to design, develop high quality softwires. The purpose of SDLC to produce high-quality products to fulfill the expectation of customers. There is an international standard for software life-cycle process ISO/IEC 12207. It aims to be the standard that defines all the tasks required for developing and maintaining software.

Different life cycle models for software development are specified and developed which are followed during the software development process. Each process has own steps to ensure quality to the software development.

Following models more used for SDLC:

- V Model
- Spiral Model
- Waterfall Model
- Agile

3.1.1 V Model

The V Model is one of the SDLC (Software Development Life Cycle) models. The model was offered by Barry Boehm in 1979. (2). Under the V-Model, the corresponding testing phase of the development phase is planned in parallel. So, there are Verification phases on one side of the ‘V’ and Validation phases on the other side. The Coding Phase joins the two sides of the V-Model.

3.1.2 Waterfall Model

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. The waterfall model is good for medium tasks. Developers can use in the it in the middle of project. At every moment in time team executes one type of work.

3.1.3 Spiral Model

The spiral model also offered by Barry Boehm in 1986. The model is evaluation on waterfall model. (2). The spiral model has four phases. The first one is identification requirements, the second is design and prototyping, the third is construct or build, the fourth is evaluation and risk analysis. A software project repeatedly passes through these phases in iterations called Spirals.

3.1.4 Agile model

Agile is one of the mostly used and popular methods for projects. The good side is developers use less documentation. The agile model is documentation-friendly, its mix of incremental and spiral models. The model has three injections. It goes through planning, requirement analysis, designing, building, and testing. The main disadvantage of an agile model is the complexity of its application to large projects.

3.2 Software testing types

There are two types of testing. They are manual testing and automated testing. Both have weight for testing and used for various aims. Testers can use many different types of testing for the project to achieve to their goals. Not all testing is equal, they differ from each other their methods.

3.2.1 Manual testing

Manual testing is a manually conducted testing method by an individual to identify

software bugs without the use of automated testing tools. It means the tester have to test and analyze the behavior of website, mobile app or desktop app by hand. In manual testing tester does not need skills or knowledge in any testing tool. Before automated testing, a new application tester should test software manually.

Manual testing mainly works good for testing:

- Functionality
- User interface (UI)
- User experience (UX)
- Website and app behavior

Manual testing takes long time and more expensive to execute test. During the testing tester can get bored of it. Additionally, it is difficult to test multilingual sites manually. Accordingly, more test managers prefer automated testing than manual testing. However, in the project it is not possible to test all application by automation tool, this makes manual testing imperative.

3.3 Types of automation testing

Automation tools mean using automation tools to execute the test case. It is significantly faster than manual testing. The test can run over night without human intervention. One of the advantages of using automation testing tool is it is possible to record and re-play written test suite. It is comfortable to use it for multilingual sites. Compared to manual testing in automation testing tester should know at least basic programming coding which required for using specific testing tool. Programmers, developers, or corporations can choose right testing tool which best

fit for the project requirements, project budget. Additionally, firm should consider who on the team has skills to use the tool before purchasing it (3).

There are three type of tools which available on the market. They are open-source, commercial and custom tools.

3.3.1 Open-source tools

Open-source tools are openly published to use for programmers. They do not have to pay for it, its free software to use. They can download it to their personal computers and can change or modify the program.

Open-source software may have fewer features than commercial tools. (3)

3.3.2 Commercial tools

Commercial tools are produced for commercial purposes or for sale. Usually, the large companies can afford commercial software testing tools. (3)

Compared to open-source tools commercial tools have more features and support from a vendor.

3.3.3 Custom tools

In some situations, testing process has specific requirements in the project. In this position open-source or commercial tools cannot do special requirement. Thus, test manager can develop special custom tool. (3)

3.4 Testing methodologies

3.4.1 Black box

For the first time the term "black box" was mentioned by the psychiatrist W. R. Ashby in his book "Introduction to Cybernetics" in 1959. He wrote that the black box method allows one to study the behavior of a system, abstracting from its internal structure.

The Black Box is a testing strategy or method, based only on testing according to functional specifications and requirements, while not looking at the internal structure of the code and without access to the database.

In the field of testing, the black box method is a testing technique that is based on working with external software interfaces, without knowing the internal structure of the system.

This method is called “Black Box”, because in this method the software under test looks like a black box for the tester, inside which some processes take place, but the tester knows nothing about them in principle. This technique detects errors in the following categories:

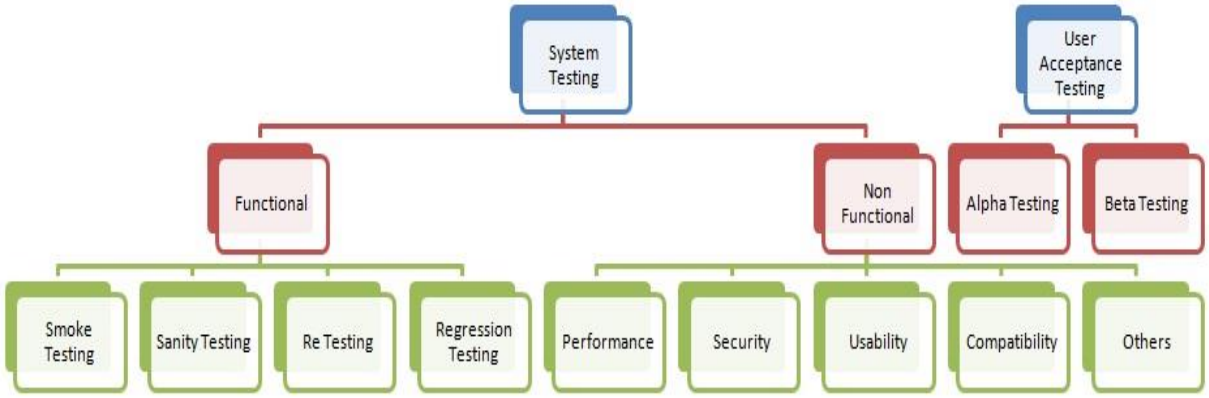
• Interface errors.

• Missing or incorrectly implemented features.

• Insufficient performance or system behavior errors.

• Incorrect data structures or poor organization of access to external databases.

Thus, since the tester has no idea about the internal structure and structure of the system, he needs to concentrate on what the program does, and not on how it does it.



Picture 1 Black box testing (4)

3.4.2 Functional testing

Smoke testing

Smoke testing is a short cycle of tests, the purpose of which is to confirm the fact that the installed application starts and performs functions after new or edited code has been built. Upon completion of testing the most important segments of the application, objective information is provided about the presence or absence of defects in the operation of the tested segments. Based on the results of smoke testing, a decision is made to send the application for revision or the need for its subsequent full testing.

Sanity testing

Sanity is a type of testing that serves as proof of the functionality of software functions or a module in accordance with the technical requirements declared by the customer. Sanitary testing is often used to test parts of a program or applications when certain environmental changes are made to it. This type of testing is usually performed manually.

Regression testing

The need to perform manual tests, the number of which is growing steadily with each build, but the whole point of which is to verify that the previously working functionality continues to work correctly. (5)

System testing

This testing of the program, such testing verifies the compliance of the program with the stated requirements.

3.4.3 Non-functional testing

Usability testing

This is a testing method that allows you to assess the degree of usability of the application, the rate at which users learn when working with the program, as well as how users of the developed product find it understandable and attractive in the context of the given conditions. Such testing is necessary to ensure the most positive user experience when working with the application.

User interface testing

This is testing the correctness of the display of user interface elements on various devices, their correct response to various actions by the user, and an assessment of how expected the program behaves. Such testing makes it possible to assess how effectively the user will be able to work with the application and how the application's appearance corresponds to the approved documents created by the designers. When testing the user interface, the main task of the tester is to identify visual and structural flaws in the graphical interface of the application, check the possibility and ease of navigation in the application and the correctness of the application processing data input from the keyboard, mouse, and other input devices. User interface testing is necessary to ensure that the interface meets approved requirements and standards, and to ensure that the user can interact with the application's graphical interface.

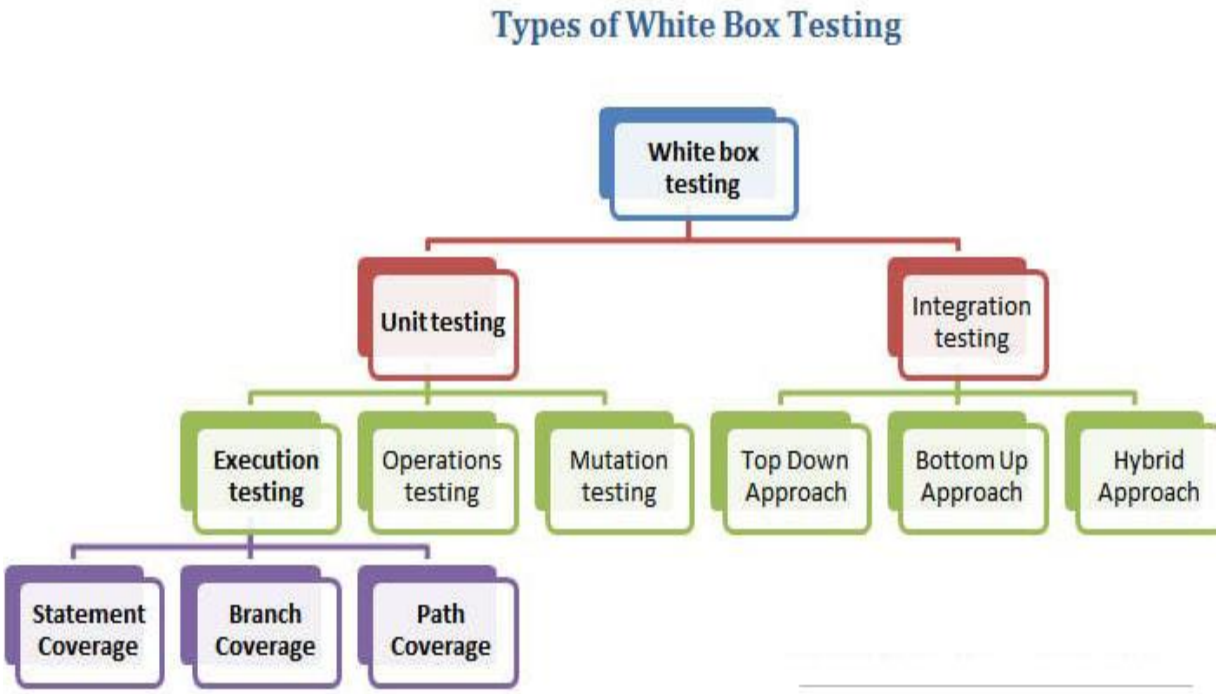
Stress testing

This type of testing is quite often carried out for software that works with valuable user data, the continuity of work and the speed of recovery from failures of which are critical for the user. Failure testing and recovery tests a program's ability to recover from hardware failure quickly and successfully, network outages, or critical errors in the software itself. This makes it possible

to assess the possible consequences of a failure and the time required for the subsequent recovery of the system. Based on the data obtained during testing, the reliability of the system can be assessed, and, if the performance is unsatisfactory, appropriate measures aimed at improving the recovery systems can be taken.

White box

White box also known as (glass box, clear box, and no box tests) internal structure design and coding.



Picture 2 White box testing (6)

Unit testing

Unit testing consists in checking each individual module (original element of the system) by running automated tests in an artificial environment. Implementations of such tests often use various stubs and drivers to simulate the operation of a real system. Automated Unit Testing is the very first opportunity to run and test source code. Creation of Unit tests for all modules of the system allows to identify errors very quickly in the code that may appear during development.

Integration testing

This testing of individual modules of the system for correct interaction. The main purpose of integration testing is to find defects and to reveal incorrect behaviour associated with errors in the interpretation or implementation of the interaction between modules.

Grey box

Grey box is mix of black and white boxes. Thus, the tester only partially knows the internal structure of the program. For example, it is assumed that you have access to the internal structure of the software to develop the most effective test cases, while the testing itself will be carried out by the black box method. Or testers can follow the black box method in everything, however, to make sure that certain algorithms are working correctly, they can look at the information in the logs or analyze the program records in the database. A slightly smaller number of testers test the grey box strategy, which implies partial access, for example, to the structure of databases or sets of parameters that services accept.

3.5 Types of frameworks for automatization

Within each software development and delivery team, the QA team is responsible for developing, implementing, and executing tests. For each type of testing, a test scenario, principles, rules, and tools for conducting should be defined. A testing framework is a collection of these guides, tools, and practices that helps test engineers execute test scripts efficiently. It gives advantages like increase re-usage of code, reduced script maintenance cost, improves test structure.

There are several types of automation use for different purposes. Some of most popular types of automated testing frameworks are in below section.

Data Driven automation framework test scripts are executed and verified based on data stored in a central data warehouse or database (SQL, ODBC resources, csv or xls files)

Keyword Driven automation framework programming skills are not required in this framework, since the keywords used to create tests are separate from the technical code. It is enough for a tester to have an idea of the whole set of actions implemented in the framework.

Modular automation framework: the application is divided into separate modules and each module is tested in an isolated state.

Hybrid automation framework is combination of different frameworks.

3.6 Characterization of Tools

3.6.1 Selenium

Selenium is developed in 2004. It is an open-source software and is free. The tool test only web applications which is friendly with various browsers, and platforms such as Linux, Mac, and Window. Additionally, it has record and playback features for writing tests without the need to learn test scripting languages. Big corporations such as Netflix, Google, HubSpot, Fitbit prefer work with this tool. The selenium supports several languages like Java, C#, Python, JavaScript and so on. There are different kind of Selenium for example, Selenium IDE, Selenium WebDriver, Selenium Grid (7). It requires good programming skills and experience to install and integrate Selenium with other tool and frameworks.

3.6.2 Watir

Watir (Web Application Testing in Ruby), pronounced as “Water” is open-source software family of Ruby libraries for automating web browsers. Ruby is object-oriented language and typically its simpler and faster than other languages. The tool helps in automating web application no matter which language the application is written. It supports Internet Explorer, Firefox, Chrome, Opera and Safari. It is more young than other tools and the initial release was 2012. Watir allows to use cross browser which means tester can run test parallel in with multiple browser combinations. Cross browser parallel test automation allows to save execution time. It has limited support on the open-source community because it was developed several years ago. Testers can use only one language which is Ruby.

3.6.3 TestComplete

TestComplete was developed in 1999 by AutomatedQA and commercial test tool. TestComplete is an automated testing tool that allows to create tests for Windows applications, desktop, and mobile platforms. This feature of tool allows testers to build strong testing framework that utilizes the broad spectrum of available software testing methodologies. It offers some key test automation features such as keyword-driven and data-driven testing, cross-browser testing TestComplete cannot use other operating systems like Linux, Unix, MacOS. TestComplete is very convenient to use for beginners, those who are not particularly good at programming. There are a lot of script languages to use for example JavaScript, Python, VBScript, Jscript, Delphi Script, C++, C#. One of the disadvantages of the tool is there are additional fees for extra modules and add-ons.

3.6.4 QTP

HP Quick Test Professional (QTP) the second name of the tool is Unified Functional Testing (UFT), one of the leading tools for automating functional and regression testing, is HP's flagship product in its lineup. It was developed in 2001 and commercial test tool. Unlike several other products for automating functional testing, QTP allows you to control the generated script text in the process of recording user actions, thereby reducing the time required to develop a test. The tool allows to create and execute the test only with basic programming skills to get started. Disadvantages of the tool are high cost for upgrades and additional modules, supported only by VBScript.

Selection of tool is one of the biggest challenges should be solved before going to automation. Firstly, the author identified requirements, studied various tools and its capabilities. Following criterion will help to select the best tool for small and start-up businesses.

3.6.5 Criteria for tools selection

The author of the thesis chose criteria based on the literature review. There are many criteria which need to work on it before deciding. Testing of automation largely depends on

understanding the project functionality in detail. The tester manager can choose criteria which is necessary to their project requirements. (8)

Test development platform - The tool should work in different operating systems such as Windows, Linux, Apple macOS (8)

Application under test- The organization should consider the project requirements whether it is for desktop-based, browser-based, or mobile-based application. (8)

Programming skills - Learning tools, and learning the language are two different aspects. It is important to select a tool that offers a language that is familiar with organizational resources and will help to minimize the learning curve. (8)

Scripting languages - The tool should have several widely used languages. Nowadays most used languages for testing are Java, JavaScript, C#, Python, Ruby, PHP (9)

Browser support - People have different browser into their mobile phones, laptops, and desktops. One test can work into other browsers differently. That is why it is important to have many browsers. (10)

Ease of learning – Some of tools needed time to learn and experience. Usually, commercial tools are easy to learn while open-source tool needs more time and sources. (10)

Ease of installation and use - the tools should be easy to use and installation. Test managers must take into account while choosing tool that do the tool require installing various tools or not. Open-source tools need additional software to install while commercial tool just require to setup and run. (11)

Record-Playback – it is not recommended to have in test automation, but it is good have a tool. It makes learning process easy and helps scenarios to be easily automated. (8)

Data-driven framework – if organization is using data-driven framework they need to have connect to any data sources such as CSV file, Excel file, XML file and Database. (8)

Product support - Product can be supported by tool official community, dedicated staff because they can guide through scripting or open-source community. Usually, commercial tools have professional community support that ensure training material, tutorial, and official forums while open-source tool have limited support on open-source community. (8)

Cost- Budget of organization is the most important thing to consider while choosing the tool. There is not only price of automation tool but also the price of add-ons, the support fee, the training fee, and upgrade fee. The developed companies use commercial paid tools. But for small and startups is good to choose low priced tools. (8)

4 Practical Part

Base of the lecture review we formulated the following research question.

RQ1: Are the free automated website testing tools capable to do the same as commercial paid tools?

RQ2: Which automated website testing tool would be optimal for a small startup company?
author of this bachelor thesis compared automated software testing tools criterions with Multiple Criteria Decision-Making method.

4.1 Multiple criteria decision analysis

Multiple criteria decision analysis (MCDA) is a method which uses ranking criteria to solve many complex decisions. MCDA accept solution in the base of many criteria. It deals with selection best tool among all available alternatives based on various criteria.

Key steps of MCDA

1. Define an objective.
2. Define criteria (measures for success)
3. List of options
4. Weight of criteria (importance)
5. Rate of options
6. Calculate and select.

The objective as mentioned before is which tool is the best for small businesses. The criteria were taken from the scientific literatures. They are test development platform, application under test, programming skills, scripting languages, browser support, ease of learning, ease of installation and use, record-playback, data-driven framework, product support and cost. (8)

4.1.1 List of options

The author of the thesis chose popular tools in the market. They are used by many companies as web application automation software testing tool. The basic purpose of these tools is comparable, but usability, features, and functionality are different. As mentioned above there are two open-source and two commercial tools which author tried to put all cons and pros of them. They are: Selenium, Watir, TestComplete, QTP.

Features	Selenium	Watir	TestComplete	QTP
Test development platform	Cross-platform	Cross-platform	Windows	Windows
Application under test	Web apps	Web apps	Windows desktop, Web, Mobile apps, API/Web services	Windows desktop, web, mobile apps
Programming skills	Needs to have prgramming skills	Partial	Needs to have prgramming skills	Partial. Quite easy to edit, navigate, parametrize
Scripting languages	Java, C#, Perl, Python, JavaScript, Ruby, PHP	Ruby	JavaScript, Python, VBScript, JScript, Delphi, C++ and C#	VBScript
Browser support	Firefox, IE, Chrome, Opera, Safari	Firefox, IE, Chrome, Opera, Safari	IE, Firefox, Chrome	IE, Firefox, Chrome
Ease of learning	Experience needed	Experience needed	Easy to learn	Easy to learn
Ease of installation and use	Require installing and integrating various tools	Require installing and integrating various tools	Easy to setup and run	Easy to setup and run
Record-Playback	Support	Support	Support	Support
Data-driven framework	Excel, Csv, Xml	Xml, Excel	Csv, Excel, Sql	Excel, Text file, Db files, Xml
Product support	Open source community	Limited support on open suorce community	Dedicated staff, Community	Dedicated staff, community
Cost	Free	Free	7000$	8000$

(8)

Table 1 Software tools with their criteria [source: own]

4.1.2 Weight of criteria (importance)

The several criteria have their own importance which helped to select the right tool for the specific requirements. There is comparison chart of tools listed below based on the most important parameters for automation project (8). For example, for each criteria author put 5%, 10% or 15%, if percentages add up to 100%

4.1.3 Rate of options

The best performing options gets the highest points and the worth performing options the lowest. There are three points which will start from 0-2.

Weight	Features	Selenium	Watir	TestComplete	QTP
5%	**Test development platform**	(Supports many platforms) 2	(Supports many platforms) 2	(Supports one platform) 0	(Supports one platform) 0
5%	**Application under test**	(Few apps) 2	(Few apps) 2	(Many apps) 2	(Many apps) 2
10%	**Programming skills**	(High) 0,75	(Medium) 1	(High) 0,75	(Low) 2
10%	**Scripting languages**	(Many) 2	(Few) 0,5	(Many) 2	(Few) 0,5
15%	**Browser support**	(Many) 2	(Many) 2	(Many) 1	(Many) 0,75
10%	**Ease of learning**	(High) 0,5	(Medium) 1,75	(High) 0,5	(Low) 2
10%	**Ease of installation and use**	(Good) 1	(Good) 0,75	(High) 2	(High) 2
5%	**Record-Playback**	(Good) 2	(Good) 2	(Good) 2	(Good) 2
10%	**Data-driven framework**	(Medium) 1,5	(Medium) 1,5	(Medium) 1,5	(High) 2
5%	**Product support**	(High) 2	(Good) 1	(High) 2	(High) 2
15%	**Cost**	(Low Cost) 2	(Low cost) 2	(High Cost) 0	(High Cost) 0

Table 2 Weight and rate of options [source: own]

Weight	Features	Selenium	Watir	TestComplete	QTP
5%	**Test development platform**	2	2	0	0
5%	**Application under test**	2	2	2	2
10%	**Programming skills**	1	1	1	2
10%	**Scripting languages**	2	0,5	2	0,5
15%	**Browser support**	2	2	1	0,9
10%	**Ease of learning**	0,75	1,5	0,75	2
10%	**Ease of installation and use**	1	0,75	2	2
5%	**Record-Playback**	2	2	2	2
10%	**Data-driven framework**	1,5	1,5	1,5	2
5%	**Product support**	2	0,5	2	2
15%	**Cost**	2	2	0	0

Table 3 Calculation [source: own]

The author multiplies weight of each criteria to the rates. After that she sum them up to get total score of each option. There is a formula how to get total score of each tool.

Formula of score

Score = Rating * Weight

Sum = Total score of each option (12)

Weight	Features	Selenium	Watir	TestComplete	QTP
5%	Test development platform	0,1	0,1	0	0
5%	Application under test	0,1	0,1	0,1	0,1
10%	Programming skills	0,1	0,1	0,1	0,2
10%	Scripting languages	0,2	0,05	0,2	0,05
15%	Browser support	0,3	0,3	0,15	0,135
10%	Ease of learning	0,075	0,15	0,075	0,2
10%	Ease of installation and use	0,1	0,075	0,2	0,2
5%	Record-Playback	0,1	0,1	0,1	0,1
10%	Data-driven framework	0,15	0,15	0,15	0,2
5%	Product support	0,1	0,025	0,1	0,1
15%	Cost	0,3	0,3	0	0
	Highest Score	**1,225**	**1,125**	**1,075**	**1,185**

Table 4 Calculation and selection [source: own]

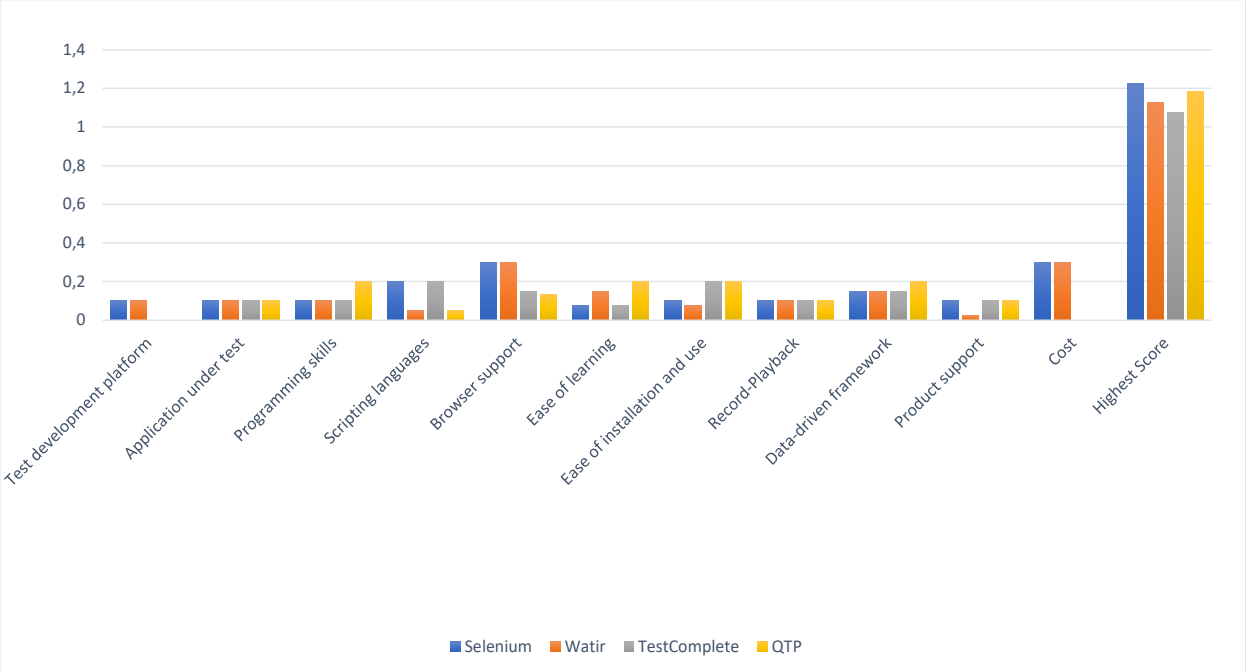


Table 5 Table of rate [source: own]

Final rating:

1. Selenium – 1,225
2. Watir – 1,125
3. TestComplete – 1,075
4. QTP – 1,185

The Selenium got highest score which is 1,225. The tool work with different platforms that ensure comfortable environment for testing. Watir, same as Selenium, supports cross-platform features. But commercial tools like TestComplete and HP QTP/UFT supports only one platform that is Windows. It can pose difficulties for instance organizations based on Linux platform. The thesis research requirements that the tool should test only web applications the Selenium is more usable for that. Watir also works well for web applications. On the other hand, commercial tools like TestComplete and HP QTP/ UFT are expensive, but they offer professional support, lots of learning material are provided and easy to use (8).

Cost is one of the important factors in automated testing. The TestComplete and QTP/UFT have two different fees like license and maintenance. QTP/UFT is highly costly. License costs around 7000$ and there is additional fee for upgrading to the latest features. Installing the tool is possible only for a month. TestComplete as QTP/UFT has free trial. The tool for web application with node-locked license user cost around 2400$ if user wants additional applications like desktop or mobile options it costs around 4200$. Moreover, it has training availability for scripted testing live online or keyword-driven testing online training which costs 743$ for each (13). The comparison shows that Selenium and Watir fits great for small businesses instead of commercial tools.

Another important factor is scripting languages. Java, JavaScript and Python are the most popular languages for web testing. Selenium supports all three languages while Watir and QTP/UFT can support only one language (8). The organization should have tester which they can use and understand these scripting languages. TestComplete as Selenium support multiple popular languages.

QTP/UFT has high point then others in data-driven framework that supports more data sources like Excel, text, Xml, and database files. However other can-do same functionalities with other tools

Selenium and Watir support all browser. One of the features of the tools that can make parallel execution of tests in multiple browsers. For using selenium needs to install various tools like documentational tool Maven, Selenide and Allure. Also, ease of learning takes more time. However, Selenium has more features for functional web testing despite it is open source.

After selected tool the author tested makeup.cz

4.2 Website test with Selenium

Java

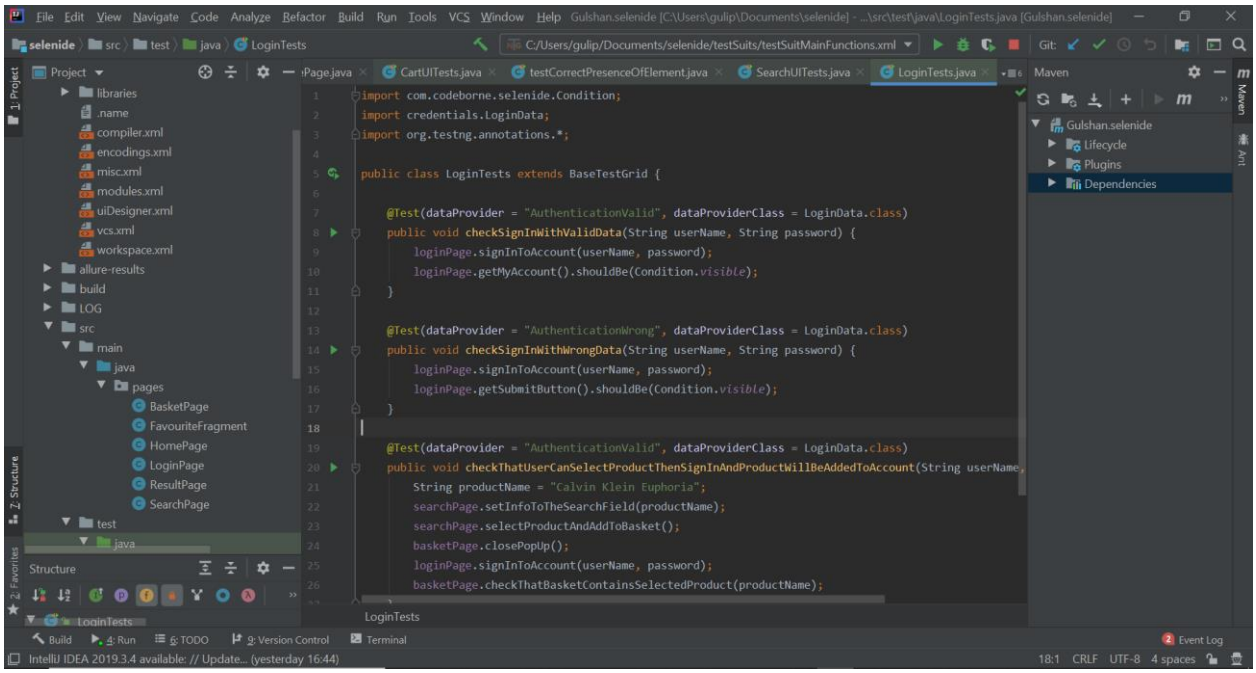
Java is a programming which used for testing for this project. Java is used to develop mobile apps, web apps, desktop apps, games and much more. Many large corporations use Java to maintain their back-end systems. There are more than 3 billion devices that are running applications built using Java. Though JUnit is a popular unit testing framework, a number of open-source automation testing frameworks have been developed using Java. Automated browser testing for a web product (website/web application) can be performed using JUnit with Selenium WebDriver. (9)

Using Selenium

The author tested pages were as:

Tested pages

- Basket page
- Favorite fragment
- Home page
- Login page
- Result page
- Search page




```
import com.codeborne.selenide.Condition;
import credentials.LoginData;
import org.testng.annotations.*;

public class LoginTests extends BaseTestGrid {

    @Test(dataProvider = "AuthenticationValid", dataProviderClass = LoginData.class)
    public void checkSignInWithValidData(String userName, String password) {
        loginPage.signInToAccount(userName, password);
        loginPage.getMyAccount().shouldBe(Condition.visible);
    }

    @Test(dataProvider = "AuthenticationWrong", dataProviderClass = LoginData.class)
    public void checkSignInWithWrongData(String userName, String password) {
        loginPage.signInToAccount(userName, password);
        loginPage.getSubmitButton().shouldBe(Condition.visible);
    }

    @Test(dataProvider = "AuthenticationValid", dataProviderClass = LoginData.class)
    public void checkThatUserCanSelectProductThenSignInAndProductWillBeAddedToAccount(String userName,
        String productName = "Calvin Klein Euphoria";
        searchPage.setInfoToTheSearchField(productName);
        searchPage.selectProductAndAddToBasket();
        basketPage.closePopUp();
        loginPage.signInToAccount(userName, password);
        basketPage.checkThatBasketContainsSelectedProduct(productName);
```

Picture **3** Tested pages **[source: own]**

Maven

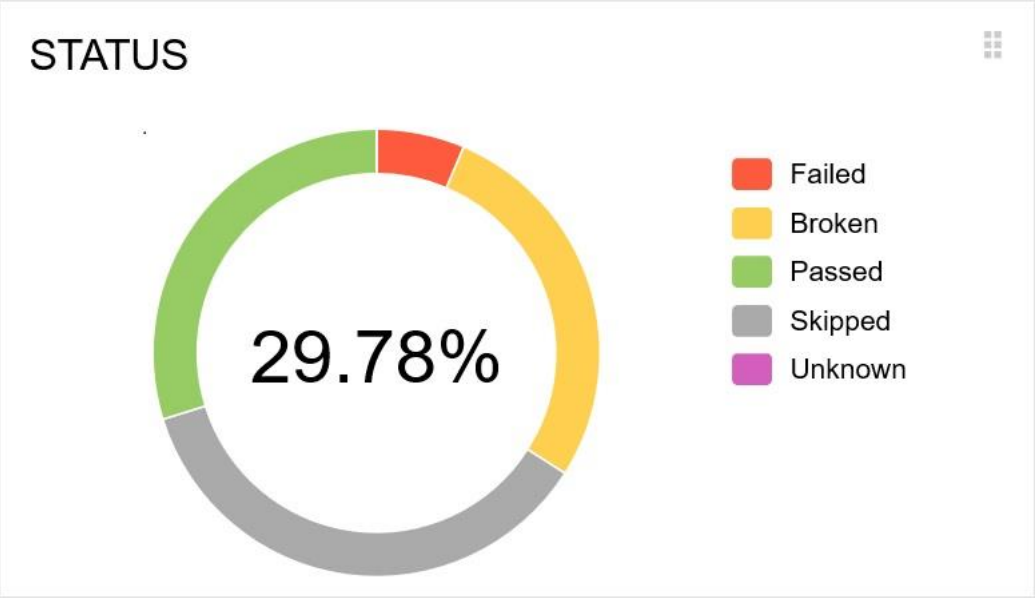
Maven is documentation tool. It is a framework to manage a project's build, reporting and documentation from a central piece of information any Java-based project. The framework based on project object model (POM). (14)

Selenide

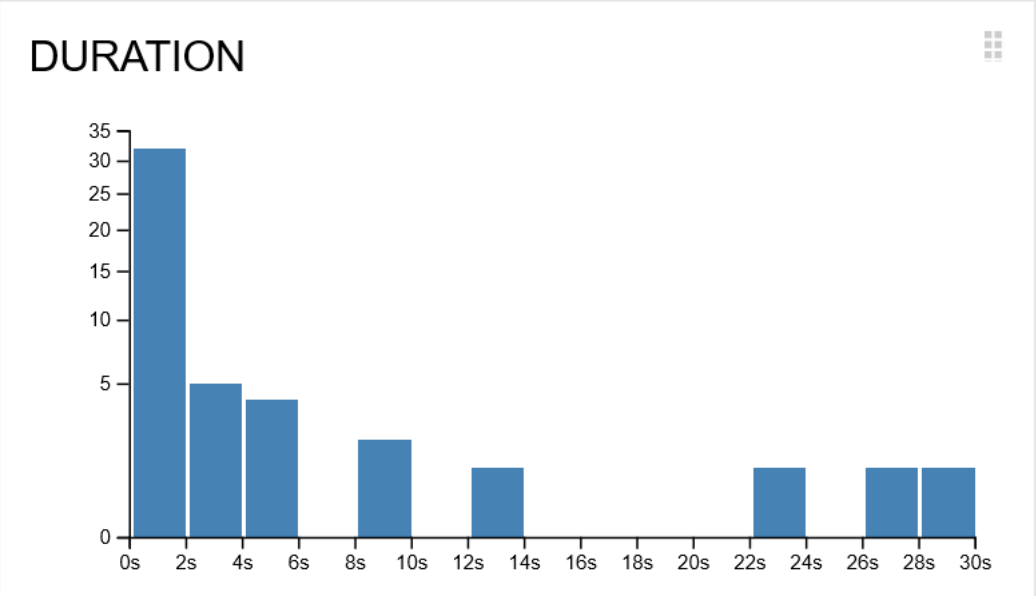
Selenide uses with Selenium WebDriver that consents to write easier and faster UI tests. It has methods which developers cannot do only with Selenium WebDriver. Selenite takes automatic screenshots of failure, clearing browser cache and so on. [(15)]

Allure Results

The Allure framework is a tool for building comprehensible auto-test reports. Alure shows what is happen in test with beautiful diagrams and which class and method the error occurred. (16)



Picture **4** Test Reports **[source: own]**



Picture 5 duration of test cases [source: own]

The tests were done in makeup.cz. Duration of the test was approximately 30 seconds in every browser like Chrome, Firefox, and Explorer. The tool automatically logged to the web site by putting username and password. After the tool chose a product “Calvin Klein Euphoria “then selecting product it was put in a basket. 47 test cases were done. The Allure result shows that:

- 3 (6.38%) tests failed.
- 13 (27.57%) tests broken.
- 14 (29.78%) tests passed.
- 17 (36.17%) tests skipped.

5 Results and Discussion

For a context analysis, the best available literature was combed through to learn about commonly used testing methods and their roles in software testing. Faiz Shaikh (17) on his blog tried right automation tool. In their case the consulting team concluded that manual and automation training would be required to validate the web, mobile, and desktop applications following a detailed review of the project specifications. As a result, they wanted a platform that worked seamlessly with their project preparation and test management software and was quick to understand and use. He put key criteria as vital parameters (Picture 7), his team used Pugh Matrix to choose the best tool for mid-scale web, desktop, and mobile test automation. The team tested Selenium Web Driver, Ranonex, TestComplete, Ghost Inspector, Test Studio. In the matrix, Ranonex got more points than others. Ranorex is a versatile and effective platform for web, laptop, and smartphone program automation. It has a very simple, easy-to-use user interface as well as recording and replay functionality. Screen scraping technology is used by the instrument, and it is independent of the Application Under Test (AUT) that is running on the screen. In the report written that "Selenium Web Driver needs more skillful resources for coding. his surpasses the cost of any other tool in terms of Pricing and ease of complexity of maintenance. However, Selenium Web Driver is the most effective tool for extensive scale web applications. (17)

Authors of a comparative review of the features of automated software testing tools (10) Heidilyn V. Gamido and Marlon V. Gamido also tried to compare various features automation testing tools like TestComplete, Ranonex, Sahi, SoapUI, Watir QTP/UFT, Selenium. They cited if project costs are a priority, open-source tools like Selenium are a smarter option. If help, ease of learning, and report generation are key considerations, licensed resources like QTP/UFT are a better choice.

Are the free automated website testing tools capable to do the same as commercial paid tools?

Unfortunately, open-source tools cannot do all functions that do commercial tool. For open-source tool the organization should have skilled experts on it otherwise it can consume more time to learn.

Commercial tools work well with database application. Supports all tests like, web based, desktop based, and mobile based applications. QTP/UFT and TestComplete works fast and uses

less CPU and RAM. Functions as web, functional testing, regression, unit, load, and other testing can be done.

Which automated website testing tool would be optimal for a small startup company?
organization budged is limited, open source as Selenium or Watir are good choice to consider. For example, Selenium the tool was developed in 2004 therefor it has good product support from users and the professional community. The beneficial parts of this tool are it supports widely used languages such as Java, C# and more. Selenium integrates with various frameworks such as Allure, Maven, Selenide. As commercial tool open-source tools have feature like record and playback. It makes learning process easy and helps scenarios to be easily automated. Supports all browsers and its free.

6 Conclusion

The theoretical part of the thesis had the task of defining elementary theoretical knowledge. The author tried to illustrate some process life cycle of software developments. The thesis describes manual, automated testing and several types of software testing. The theoretical part also focused on automation testing tools which are popular nowadays and used by many big companies. All theoretical knowledge was obtained based on analysis literature and resources available.

The automation testing is an important part of software testing. Test automation has been a vital component of successful program testing. Test automation, according to the new World Quality Report 2018–2019, is the greatest bottleneck in providing “Quality at Speed,” as it saves time, resources, increases performance, and improves accuracy (11). As a result, without the correct audacity, accurate and efficient test automation is impossible.

The aim of the practical part of this thesis was to define what is great tool for startups and small businesses developers. In this thesis, the author compared and analyzed commercial paid and open-source tools (Selenium, Watir, TestComplete, UFT/QTP) on various quality factors. The comparison was performed by multi-criteria analysis based on selected criteria and set weights. Every tool has cons and pons. The author put rating to characteristics by their importance After the overall analysis, it is not easy only to rank these tools based on number of factors. Then the author tests the tool which has more points than other whether to know how it work. The installation of various tools for it and use was difficult for author. The learning of tool took significant time. But, in recent years, Selenium has been the chosen tool for most computing industries, and testers and developers use it more than other tools. The simplicity of Selenium's features and functionalities, as well as its capacity to interact with a range of structures and the number of different languages it supports programming languages and the fact that it is online has substantial benefits. The testing was performed in makeup.cz

References.

1. *Choosing the right automation tool and framework is critical to project success.* **Bajaj, Harsh.** Bengaluru : Information Technology consulting company, 2018, p. 8.

2. **Homès, Bernard.** *Fundamentals of Software Testing.* London : John Wiley & Sons, Incorporated, 2012. ISBN: 978-1-84821-324-1.

3. **Krishna Rungta.** How to select best automation testing tool. *Guru99.* [Online] [Cited: March 1, 2021.] https://www.guru99.com/testing-automation-why-right-tools-are-necessary-for-testing-success.html.

4. **Testing Go Easy.** Testing go easy. *Sites google.* [Online] Testing go easy. [Cited: March 3, 2021.] https://sites.google.com/site/testipscenter/validation/Black-box-testing.

5. **Savin, Roman.** *Testing dot com.* Moscow : "Дело", 2007. ISBN 978-5-7749-0460-0.

6. **Diffrerence between Black Box and White Box Testing.** ***Testing genez.*** **[Online] Testing Genez, october 12, 2019. [Cited: March 1, 2021.] https://testinggenez.com/black-box-and-white-box-testing/.**

7. Mahajan, Amod. Selenium vs Testcomplete. ***knowlaehutdge.*** **[Online] June 20, 2019. [Cited: February 1, 2020.] https://www.knowledgehut.com/blog/software-testing/selenium-vs-testcomplete.**

8. STH. How to choose the best automation testing tool. ***Software testing help.*** **[Online] Software Testing Help, february 18, 2021. [Cited: March 2, 2021.] https://www.softwaretestinghelp.com/automation-testing-tutorial-4/.**

9. Himanshu Sheth. Top 7 Programming Languages For Test Automation In 2020. ***https://www.lambdatest.com/.*** **[Online] January 23, 2020. [Cited: Februay 1, 2020.] https://www.lambdatest.com/blog/top-7-programming-languages-for-test-automation-in-2020/.**

10. Gamido, Heidilyn V. and Gamido, Marlon V. ***Comparative review of the features of automated software testing tools.*** **Tarlac : International Journal of Electrical and Computer Engineering (IJECE), 2019.**

11. Mubarak, Albarka Umar and Chen, Zhanfang. ***A Study of Automated Software Testing: Automation Tools and Frameworks.*** **Jilin : International Journal of Computer Science Engineering, 2019. ISSN : 2319-7323.**

12. North Carolina Cooperative Extension Service. Multi-Criteria Decision Analysis. ***The Natural Resources Leadership Institute.*** **[Online] 2011. [Cited: February 1, 2020.] projects.ncsu.edu/nrli/decision-making/MCDA.php.**

13. Pricing. ***Smart Bear.*** **[Online] SmartBear Software. [Cited: March 10, 2021.] https://smartbear.com/product/testcomplete/pricing/.**

14. Apache Maven Project. What is Maven? ***Apache Maven Project.*** **[Online] The Apache Software Foundation. [Cited: February 1, 2020.] https://maven.apache.org/what-is-maven.html.**

15. Selenide. What is Selenide. ***Selenide.*** **[Online] Codeborne well-crafted software, April 23, 2019. [Cited: February 1, 2020.] https://selenide.org/2013/04/23/what-is-selenide/.**

16. Allure. Allure Test Report. ***Allure. qatools.*** **[Online] Allure. [Cited: februry 1, 2020.] http://allure.qatools.ru/.**

17. Shaikh, Faiz. 4 Simple Steps to Select the Right Test Automation tool for your Project. ***Saviant consulting.*** **[Online] Saviant itelligent solution. [Cited: March 1, 2021.] https://www.saviantconsulting.com/blog/4-steps-select-test-automation-tool.aspx.**

18. Glenford J.Myers, Corey Sandler, Tom Badgett. -3rd ed. ***The art of software testing.*** **New Jersey : Word association, 2012.**

19. Alexsoft. The Good and the Bad of Selenium Test Automation Software. ***https://www.altexsoft.com/.*** **[Online] Alexsoft, February 12, 2021. [Cited: January 28, 2021.] https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-selenium-test-automation-tool/.**

20. Tutorials point. SDLC Tutorial. ***Tutorials point website.*** **[Online] Tutorials piont, June 12, 2014. [Cited: 03 2021, 14.] https://www.tutorialspoint.com/sdlc/index.htm.**

21. Abhik Khandelwal. Didderence berween BlackBox and White Box Testing. ***Testing Genez.*** **[Online] Testing Genez, October 12, 2019. [Cited: February 28, 2021.] https://testinggenez.com/black-box-and-white-box-testing/.**

22. Angie Jones. Which programming language is most popular for UI test automation in 2019? ***Applitools.*** **[Online] Applitools, January 26, 2019. [Cited: January 28, 2020.] https://applitools.com/blog/language-software-test-automation?utm_referrer=https%3A%2F%2Fwww.google.com%2F.**

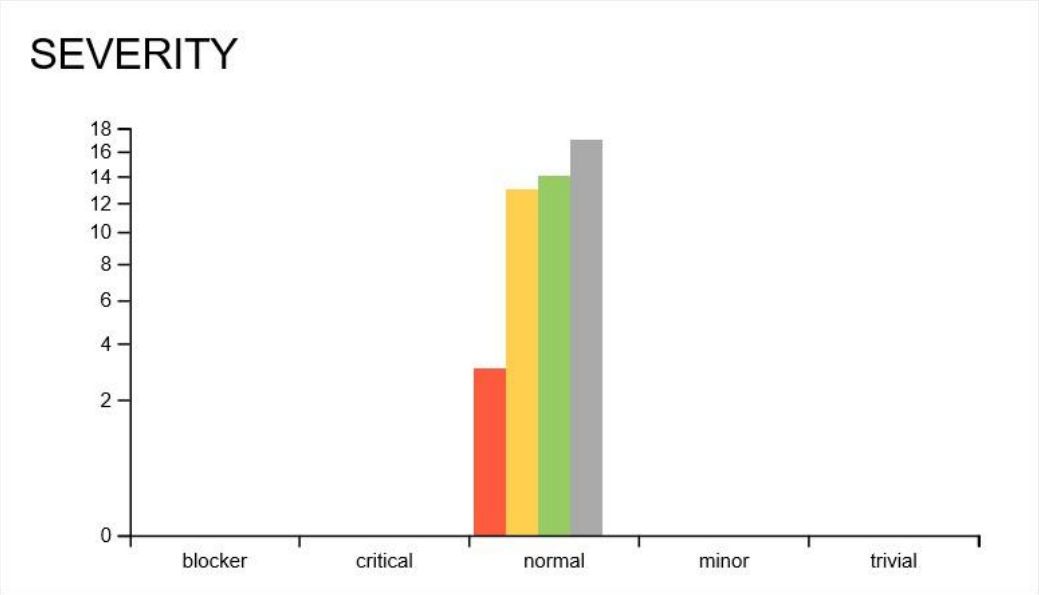
23. Guru99. What is Non Functional Testing? ***guru 99.*** **[Online] guru 99, March 27, 2019. [Cited: 03 14, 2020.] https://www.guru99.com/non-functional-testing.html.**

24. Katalon. A Comparison of Automated Testing Tools. ***Katalon website.*** **[Online] Katalon. [Cited: January 10, 2021.] https://www.katalon.com/resources-center/blog/comparison-automated-testing-tools/.**

25. TestComplete features. ***Smartbear.*** **[Online] SmartBear Software. [Cited: February 1, 2021.] https://smartbear.com/product/testcomplete/features/.**

26. selenium.dev. ***selenium.*** **[Online] Selenium, February 20, 2021. [Cited: February 1, 2021.] https://www.selenium.dev/documentation/en/introduction/the_selenium_project_and_tools/.**

7 Appendix



Picture 6 Severity of tests [source: own]

Pugh Matrix						
			Solution Alternatives			
Concept Selection Legend: Better +, Same S, Worse -						
Key Criteria	Importance Rating	Selenium Web Driver	Ranorex	Test Complete	Ghost Inspector	Test Studio
Ease of Developing and Maintaining the scripts	5	S	+	+	-	+
Ease of Test Execution for Non-technical user	5	S	+	+	+	S
Support to Web, Desktop & Mobile Application	5	S	+	+	-	S
Intuitive Test Report Generation	4	S	+	+	S	S
Cross Browser Testing	4	S	+	+	-	+
Support to Keyword & Data Driven Testing	4	S	+	+	S	+
Technical support and assistance	3	S	+	+	S	S
C# Language Support	3	S	+	+	-	+
Pricing	3	S	-	-	+	-
TFS DevOps Integration with Builds	3	S	S	S	-	S
Sum of Positives			8	8	2	4
Sum of Negatives			1	1	5	1
Sum of Sames			1	1	3	5
Weighted Sum of Positives			33	33	8	16
Weighted Sum of Negatives			3	3	20	3
TOTALS			30	30	-12	13

Picture 7 matrix technique to select right test tool (17)