Noname manuscript No.
(will be inserted by the editor)

A comprehensive review on various tools for Software Testing and Quality Assurance

Kushal Rajput · Devanshi Patel · Nirav Madhani

Received: date / Accepted: date

Abstract :

Software Testing is the most accepted and widely used methodology for finding rectifying the errors which get occurred during the execution of a program. It aims to get a zero-defect software system and at assessing a program's potential or usefulness. Software testing is an important means of determining the quality of a software system. This guarantees that the software is of high quality and meets industry standards. It is critical to be time and cost-mindful while testing software. Because of this, most testers have shifted from manual testing to software testing automation in order to save time and money. However, selecting a software testing tool for automated testing that best fits a project is an important yet difficult task. The goal of this paper is to evaluate some of the most commonly used software testing tools, identify their strengths and weaknesses, and also the field where they can be used, either for mobile testing, web service testing, or both. Software system testing is an important area of study, and much progress has been made in this subject. Testing strategies and tools are discussed in this work. Some of the most recent studies have been summarised. Software system testing will become increasingly important in the future.

Keywords Software Testing · Quality Assurance · Testing Techniques · Testing Tools · Analysis of Testing Tools.

1 Introduction

Software testing is a procedure for assessing and improving the quality of a software programme. Thus, the purpose of testing is to consistently and progressively discover numerous kinds of problems in the smallest amount of time and with the least amount of work possible. Software testing is also an important component of software quality assurance (SQA), with many software companies devoting up to one-fourth of their resources on testing. Testing comprises four major goals: detection, prevention, demonstration, and quality enhancement. [**?**][**?**].

Testing will be crucial and costly for real-time software systems, as risk analysis is also involved. As a consequence, a variety of test cases and test plans are developed during testing, implying that a program's behaviour is studied on a restricted number of test cases., i.e. inputs, execution preconditions, and additionally expected outcomes for a specific goal, such as following

Address(es) of author(s) should be given

a specific programme path or verifying compliance with a specific demand, that valued inputs are created.

With the increased functionality of software, it is vital to employ testing methodologies and tools to guarantee that testing time is minimal. When it comes to software testing, there are two types: manual and automated. The tester is in charge of manual testing. Informal review, inspection, walk-through, and technical assessment are examples of manual testing methods. Manual testing takes longer and requires more effort; these are the main disadvantages of this kind of testing. Automation testing assists in resolving and mitigating these issues.

Performance testing, safety testing, accuracy testing, and reliability testing are the four types of automated testing. Manual testing stages have been automated using automation technologies. These tools are used to test the process and are tailored to a specific testing environment.

The primary goal of software system testing is verification, validation, and error detection in order to uncover various faults and issues – and fix them. Software system testing involves more than just error detection.[**?**].

Under controlled conditions, software system testing is conducted for:

- Verification: To ensure that the system operates as expected. It is the verification and testing of objects, including software systems, for conformance and consistency of software systems by comparing the outcomes to pre-defined requirements. In verification, we tend to raise a problem, and we tend to construct the product, correct?.
- Validation: During this process, we examine the system's accuracy, which is the means of ensuring that what the user provided and what the user truly desired[**?**]. During validation, we ask ourselves, "Are we developing the right system?"
- Error Detection: Detecting previously unnoticed errors. A variety of tests are performed not just to determine what is wrong, but also to determine what will happen if software is subjected to negative testing, i.e. for purposes for which it was not designed.

1.1 Background of Software Testing and Quality Assurance:

The success of every software product is defined on its quality. This provides a significant potential for software quality assurance in the software business, and customer happiness drives it. Developing a high-quality, defect-free product within budget and schedule restrictions has become important. It is challenging to implement such goods with few or no bugs. This is the reason for the emergence of the notion of software testing[**?**].

Software testing has evolved into a comprehensive and critical step of the SDLC in the software industry. It also gives the final assessment of other operations like as requirements formulation, software design, and coding[**?**]. Software testing is a process that examines the correctness and functionality of software to ensure that user requirements and expected quality are fulfilled[**?**]. Software testing, according to the IEEE, is the process of manually or automatically analysing a system or its components to check if they fulfil user requirements or to detect the difference between actual and expected outcomes[**?**].

As a result, software testing is the act of executing software to look for flaws or missing functionality that the user needs intended. Quality assurance originated in the manufacturing business and has since moved to a wide range of industries, including software development. Any systematic method of verifying whether a product or service fulfils predetermined standards is referred to as quality assurance (QA). QA sets and maintains a set of guidelines for designing or manufacturing reliable products. A quality assurance system is designed to improve consumer trust and credibility while also enhancing work procedures and efficiency, allowing a business to compete more effectively.

The ISO (International Organization for Standardization) is a driving force behind quality assurance methodologies and procedure mapping. QA is typically used in connection with the international standard ISO 9000. Many businesses utilise ISO 9000 to ensure that their quality assurance system is in place and functioning effectively.

The concept of quality assurance as a systematic approach began in the manufacturing industry and has since spread to a variety of industries, including software development. Quality assurance aids a company in generating items and services that meet the demands, expectations, and standards of its customers. It yields high-quality outcomes.

1.2 Paper Organization

Overall structure of paper. The organization of the rest of the paper is as follows. Section 2 discusses various software testing methodologies. Section 3 discusses type of testing. In section 4 we perform analysis of various testing tools. In section 5 we talk about types of testing tool. In section 6 we discuss the results and finally conclude paper in section 7

2 Software Testing Methodologies

2.1 White Box Testing

Internal details and the structure of the system are made evident during this testing. As a result, it's incredibly cost-effective in terms of identifying and addressing issues, as bugs are often discovered before they create difficulties. This process will be described as evaluating software using data from its internal structure and code. White box testing, white box analysis, and clear box analysis are all terms used to describe white box testing. It's a technique for detecting flaws in which the tester has comprehensive knowledge of how the program's various components interact [**?**]. This approach isn't commonly used for debugging in large systems and networks, but it is in netservices applications. Basis path, control structure testing, loop testing, and more forms of white box testing exist.

2.1.1 Why perform white box testing

Because it examines and operates by internal functionality, it provides for the detection of hidden faults and internal errors. It aids in the detection of bugs and the optimization of code when using various White Box Testing approaches to test a produced application or website. Testing necessitates internal knowledge, which aids in achieving optimum code coverage.[**?**]

2.2 Black box testing

Black Box Testing is a type of testing in which no information of the system's core operation or structure is provided. The system is treated as a black box or closed box in this testing method. The tester simply knows the program's formal inputs and expected results; he or she has no idea how the program gets to those outputs. As a result, all testing must be performed in accordance with functional standards.[**?**] As a result, black box testing is sometimes referred to as functional testing, behavioural testing, closed box testing, or opaque box testing. [**?**] Although

behavioural testing is similar to black box testing, behavioural test design differs slightly since internal information may be present in behavioural testing.

2.2.1 Why perform Blackbox Testing

When utilised on big systems, it is quite efficient. Testing is balanced and unprejudiced since the tester and developer are independent of one another. A non-technical tester is possible. The tester does not need to be an expert in the system's functionality. The tests will be conducted from the perspective of the end user, because the system must be accepted by the end user. (This method of testing is also known as Acceptance testing.) Testing aids in the detection of ambiguity and inconsistencies in functional requirements. As soon as the functional requirements are established, test cases may be created.[**?**]

2.2.2 Challenges in Black-box testing

Without explicit functional requirements, creating test cases is difficult. If the test cases are not designed based on requirements, it is difficult to find challenging inputs. In a short amount of time, it's tough to detect all possible inputs. As a result, developing test cases may be time-consuming and challenging. During the testing procedure, there is a risk that unidentified routes will emerge. There's a good chance you'll be duplicating tests that the programmer has already run.[**?**]

2.3 Grey box testing

In recent years, a third testing approach, known as grey box testing, has been considered. It's defined as a software package that's being tested as well as containing some information about its core logic and underlying code. Over black box testing, it relies on internal data structures and algorithms to generate test cases, yet many people prefer white box testing. When doing integration testing across two or more modules of code authored by separate developers, where only their interfaces are available for testing, this method is required. Reverse engineering is used to calculate boundary values in this method [**?**].

2.3.1 Why perform grey box testing

When it comes to testing, both users and developers have distinct objectives. The user perspective is used for the majority of grey box testing. This testing does not need a high level of programming knowledge on the part of the tester. Gray box testing is a non-intrusive method of testing. The product's overall quality has increased. Developers have more time to resolve bugs with grey box testing. Gray box testing combines the advantages of both black box and white box testing. Gray box testing is very objective. It prevents clashes between testers and developers. In integration testing, grey box testing is far more successful. [**?**]

3 Types of Testing

In Software Engineering, there are 2 types of Testing.Which are: Automation Testing and Manual Testing.
Manual Testing :

1 Functional Testing
1.1 Unit Testing
1.2 Integration Testing
1.3 System Testing
2 Non Functional Testing
2.1 Performance Testing
2.1.1 Load Testing
2.1.2 Stress Testing
2.1.3 Scalability Testing
2.1.4 Stability Testing
2.2 Compatibility Testing
2.3 Usability Testing

Let's discuss all of them one-by-one.

Manual Testing:
Manual testing is the process of testing software or applications according to the needs of the client without the use of any automation tools.It is also known as a verification and validation technique. Manual testing is done to ensure that an application or program behaves in accordance with the requirements. The sub types of Manual Testing are as follows:

3.1 Functional Testing

Functional testing is a sort of testing that aims to determine whether each application feature functions in accordance with the software requirements. The result of each function is compared to the relevant requirement to see if it meets the end user's expectations.

It is a type of Black Box testing in which the tester have no knowledge of the internal working of the software.

3.1.1 Unit Testing

Unit Testing is a sort of software testing that examines individual software units or components. The goal is to ensure that each unit of software code works as intended. Unit testing is carried out by developers throughout the development phase of an application. Unit tests are used to isolate a part of code and ensure that it is correct. Any singular function, method, process, module, or object might be considered a unit.

3.1.2 Integration Testing

Integration Testing is a sort of testing in which software modules are conceptually connected and tested as a unit. A software product is developed by various programmers. This type of testing helps in finding loopholes when the software modules are combined to find out how they interact. It is also sometimes known as String Testing or Thread Testing.

3.1.3 System Testing

During system testing, also known as system-level tests or system-integration testing, a quality assurance (QA) team evaluates how the many components of an application interact in the

overall, integrated system or application.System testing guarantees that an application performs as intended.

3.2 Non Functional Testing

Non Functional Testing is a sort of software testing that examines a software application's non-functional elements (such as performance, usability, and dependability). It's intended to assess a system's readiness based on nonfunctional criteria that aren't covered by functional testing.Checking how many individuals can simultaneously login into a software is a great example of a non-functional test.

3.2.1 Performance Testing

Performance testing is a software testing method for evaluating a software application's speed, reaction time, stability, dependability, scalability, and resource utilisation under a specific load. Performance testing's main goal is to find and eliminate performance bottlenecks in software applications. It's also known as "Perf Testing" and is a subset of performance engineering.
Performance testing is concerned with determining a software program's ability to do certain tasks.
Speed — The application's response time is determined by this.
Scalability — The software application's ability to handle a certain amount of user traffic.
Stability — Determines whether or not the application will remain stable under varying loads.

Performance Testing can be further divided into 4 types which are:

- Load Testing:
  Load testing is a non-functional software testing procedure that evaluates a software application's performance under a certain load. It controls how a software application acts when numerous people access it at the same time. Load testing is used to identify performance bottlenecks and guarantee that software applications are stable and operate smoothly before they are deployed. Typically, this type of testing identifies –
  - An application's maximum functioning capability.
  - Check to see if the current infrastructure is adequate to run the application.
  - In terms of peak user demand, the application's viability is questionable.
  - The number of concurrent users that an application can sustain, as well as its scalability to accommodate extra users.
  - Load testing is widely used in Software Engineering for Client/Server, Web-based applications – both Intranet and Internet.

- Stress Testing:
  Stress testing is a sort of software testing that checks a software application's stability and reliability. Stress testing is used to assess software's robustness and error-handling skills under excessive load conditions, as well as to ensure that it does not crash in critical scenarios. It even goes beyond regular operating parameters to assess how software performs in difficult circumstances.
  Stress Testing is often referred to as Endurance Testing in the software engineering world. AUTs are stressed for a brief length of time during Stress Testing to determine their endurance. Stress testing is commonly used to determine the point at which a system,

software, or hardware fails. It also looks to see if the system can handle errors well in difficult situations.

- Scalability Testing:
  Scalability testing is a non-functional testing approach that assesses the performance of a system or network as the number of user requests rises or declines.Scalability testing ensures that the system can handle anticipated increases in user traffic, data volume, transaction counts frequency, and so on. It assesses the system's ability to fulfil escalating demands.
- Stability Testing:
  Stability testing is a type of non-functional software testing that assesses a software application's efficiency and capacity to work continuously over time. Stability testing is used to see if a software application crashes or fails during typical operation at any point in time.

3.2.2 Compatibility Testing

Compatibility testing examines if your programme can run on a variety of hardware, operating systems, applications, network settings, and mobile devices.

3.2.3 Usability Testing

Usability testing, sometimes called User Experience (UX) testing, is a method of determining how simple and consumer-friendly a software product is. The software application is used by a small group of target end-users to expose usability flaws. Usability testing is primarily concerned with the application's simplicity of use, flexibility in handling controls, and ability to achieve its goals.

This testing is suggested at the SDLC's early design phase, as it provides a better understanding of the consumers' expectations.

Automation Testing:
Automation testing, often known as test automation, is a software testing approach that involves the execution of a test case suite using particular automated testing software tools. Manual testing, on the other hand, is carried out by a human sitting in front of a computer, methodically carrying out the test processes. Software test automation requires significant financial and human investment. In addition, the automated testing software may insert test data into the System Under Test, compare predicted and actual outcomes, and supply comprehensive test reports
The tools used in Automation Testing are discussed below.

4 Comparative analysis of Software Testing Tools

4.1 Apache JMeter

JMeter is an Apache test tool for analysing and measuring the performance of applications, software services, and products. It's free software developed completely in Java that can be used to test both web and FTP applications as long as the machine has a Java Virtual Machine installed (JVM).

JMeter works by letting users to make and send HTTP (Hypertext Transfer Protocol) requests to the server, mimicking visitors to your application or service. The statistical data is then shown for users in the form of charts and reports based on the server response data.

4.2 Appium

Appium is a brand-new approach to automated testing that promises effective, bug-free, and high-quality apps while saving time, money, and labour for a project. Appium is a cross-platform, open source mobile testing tool that allows developers to create tests for both iOS and Android. The Appium server, the Inspector, and the Doctor are its three main components. Appium supports a wide range of languages, including java, python, ruby, and JavaScript, making it useful for developers of various skill levels. Appium is a mobile web app testing platform that focuses on user engagement with content. The accuracy and precision of a test are determined by the results.

4.3 Selenium

Selenium is most popular, widely used and essentially a free and open source tool for the testing of web application framework. It works with Windows, Mac Operating System also on Linux browsers as well as operating systems. Selenium allows testers to create tests in a variety of computer languages, including Java, PHP, C, Python, Groovy, Ruby, and Perl. The Selenium suite consists of four primary components: Selenium IDE, Selenium RC, WebDriver, and Selenium Grid. Its goal is to make automated testing of web application functional elements more engaging and enjoyable. So now It may also be used to perform the black box testing on web applications [12, 19, 20].

4.4 Test Link

The most extensively used web-based open source test management solution is Test-link. It synchronises both the requirements and the test specifications at the same time. This application allows users to build test projects and describe test cases. You may establish accounts for many users and assign different user responsibilities using Test-Link. The task of assigning test cases may be managed by the admin user.

Test cases can be executed both automatically and manually. With this tool, testers may create Test Plans and Test Reports in a fraction of the time. It accepts test reports in a variety of formats, including Excel, MS Word, and HTML.

4.5 JUnit

JUnit is a Java testing framework that is free and open-source. The establishment of test cases is required for unit testing. A unit test case is a piece of code that verifies that the programme logic functions as expected. It finds mistakes early in the code, improving the reliability of our programmes. Developers that work in a test-driven environment will benefit from JUnit. Instead of writing code, unit testing requires a developer to read it. Now, when we code in JUnit, we have the potential to improve our code such that it is more readable, dependable, and bug-free, increasing confidence and performance throughout the development process. Using JUnit, we can cut down on testing time.

4.6 Katalon Studio

Katalon Studio is a powerful automation tool based on the Selenium framework that was first published in January 2015. Katalon is primarily intended for the creation and reuse of au-

tomated UI test scripts. Katalon Studio enables automated testing of UI components like as pop-ups, iFrames, and wait time. Microsoft Windows, macOS, and Linux are all supported by the programme.

Initially available as a free solution, Katalon Studio Enterprise and Katalon Runtime Engine were introduced in October 2019 to give more feature-rich solutions for various purposes. The basic Katalon Studio for single users, on the other hand, is still available for free.

When compared to Selenium, the industry leader, Katalon's key benefit is simplicity of implementation and a larger selection of connectors. Katalon has two scripting interfaces to accommodate users with varying levels of programming experience. This implies that testers with low technical skills may utilise a user interface that does not need them to write code.

4.7 Watir

Watir (pronounced as Water) is an acronym of Web Application Testing in Ruby, is an open source programme created in Ruby that helps in the testing of web applications written in any language. Watir supports Internet Explorer, Firefox, Chrome, Safari, and Edge browsers. Watir is a Rubygems gem that may be installed.

Watir manipulates browsers in the same manner as humans do. It also performs checks on the findings, such as verifying that the desired text appears on the page. Watir supports any application, regardless of the technology used to create it. On Windows, it only works with Internet Explorer.

Some of it's properties include:

- Watir is a free Open Source programme with no fees associated with its use.
- Watir is a very active platform with a developing community.
- Rather than the proprietary vendor script, Watir utilises Ruby, full-featured contemporary scripting language.
- Watir supports any web app, regardless of the programming language used.
- Watir works with a variety of browsers and systems.

4.8 SoapUI

SoapUI is a functional automation testing tool that works across platforms. SoapUI is a free and open source tool for testing APIs like SOAP and REST interfaces to ensure application compatibility.

SoapUI is an enterprise-class functionality with an easy-to-use graphical interface. It makes it simple to develop and run automated functional, regression, and load tests. SoapUI is fully tested and supports all of the industry's standard protocols and technologies. SoapUI is a non-functional API testing tool that performs performance and security tests as well as functional API testing.

Some of its functionalities are as below:

- It's a great tool for testers that want to develop Functional API Tests.
- It has a drag-and-drop function that speeds up the script writing process.
- It supports various environments, making it simple to transition between QA, Dev, and Production settings.

- It has the ability to carry out a full vulnerability scan.
- It also protects the application's databases from SQL Injection.
- It effortlessly mimics high volume and real-world load testing.
- It enables extensive custom reporting to track performance metrics.

4.9 Espresso

Espresso is an Android native testing framework for creating dependable user interface tests. Google published the Espresso framework in October 2013, and it is now part of the Android Support Repository as of version 2.0. One of Espresso's most useful advantages is that it automatically synchronises your test operations with your application's user interface. The framework also makes certain that your action begins before the tests begin. It may also make a test wait until all observer background operations are complete, which is something that other testing frameworks struggle with.Espresso allows you to replicate user activities and test complicated intra-app user interactions programmatically. Espresso supports writing scripts in Java and Kotlin.

4.10 Ranorex

Ranorex is a robust test automation platform. It is a graphical user interface (GUI) test automation framework for online, desktop, and mobile applications. To automate an application, Ranorex does not have its own scripting language. It makes use of well-known programming languages like VB.NET and C. Any desktop, web-based, or mobile programme may be automated with Ranorex. Ranorex supports technologies like Silverlight,.NET, Winforms, Java, SAP, WPF, HTML5, Flash, Flex, Windows Apps (Native/Hybrid), iOS, and Android

The Ranorex utility may run the following sorts of automation tests:

- Cross Browser Testing: Ranorex tests and certifies web applications on browsers like Chrome, Internet Explorer, Safari, and Firefox.
- Regression Testing: In continuous build settings, Ranorex may be used for regression testing to detect new software defects much quicker. []

4.11 Kobiton

Kobiton is a cloud platform for running mobile and web tests, both automated and manually. With Selenium WebDriver (for web applications) and Appium, Kobiton can execute automated testing (for native and mobile web applications).

One may test your websites in any language you like — there's no need to learn new programming languages or scripting languages. One also don't need to install any testing SDKs, and the Appium binding might be a hassle for the test and development teams. There's no need to install Appium or configure your devices when you run Appium in the cloud. You'll get fast access to hundreds of browsers to test on. [**?**] Testing on real devices, both manually and automatically It allows for parallel testing. It allows you to record video, screenshots, and user interactions. Jenkins, JIRA, GitHub, Travis CI, and TeamCity are just a few of the prominent technologies that may be integrated. Selenium, Appium, and Katalon Studio are all supported.

4.12 k6

K6 is a load testing tool and SaaS for engineering teams that is free and open source. Often used with JavaScript but without the need for NodeJS, and CI-integrable. JMeter, HAR, Swagger scripts, and Postman scripts may all be converted to k6 scripts. JMeter isn't the only option. HTTP and WebSockets are supported. Official Docker image is available.

There are many advantages of using k6 like support for modules in ES6 JS scripting to enable code re-usability throughout an organisation. Everything is written in code: the test logic and configuration choices are both written in JS to make version control easier. Checks (such as assertions) and thresholds are automation-friendly, allowing for simple and versatile CI configuration!. It also provides support for HTTP/1.1, HTTP/2, WebSockets, and gRPC protocols.Moreover client certificates, adjustable SSL/TLS versions and cyphers are all aspects of TLS. Cookies, cryptography, custom metrics, encodings, environment variables, JSON, HTML forms, files, flexible execution control, and more were among the batteries.There is also a built-in HAR converter to save browser sessions as .har files and convert them to k6 scripts instantly. InfluxDB (+Grafana), JSON, or k6 Cloud provide flexible metrics storage and display. Cloud execution and distributed testing (currently just on their k6 cloud-managed infrastructure, with native distributed execution in k6 on the horizon!) [**?**]

4.13 AutoIT

AutoIt is an open source utility for automating Windows and desktop application activities. AutoIt is a freeware BASIC-like scripting language for automating the Windows GUI and general programming, according to the company's official website. It basically automates different operations by using a combination of keystrokes, mouse movements, and window/control manipulation. AutoIt is compatible with all Windows operating systems. It has evolved into a very strong language that supports numerous complicated expressions, user functions, loops, and other features since its original release by Jonathan Bennett and the AutoIt team in January 1999, about 17 years ago.

Its scripting languages are C and Visual Basic. AutoIt is a third-generation programming language that uses a traditional data architecture and a variety of data types to store a variety of data kinds. It may be used to automate any task by writing a few lines of script or recording the process using the AutoIt Recorder. The script is automatically created when the recording is completed, and we can use it to play back the recorded procedure. It uses numerous attributes like coordinates, the control ID, the element's name, and other smart identifiers to identify distinct pieces of the application.

With the built-in 'Send Key' feature, AutoIt can even imitate different mouse motions and keystrokes. These motions, as well as other crucial acts, can be recorded and played replayed later.

Script compilation into standalone executables: The complete AutoIt script may be saved as a.au3 executable file, which can be readily built and executed by just double-clicking the.au3 executive file.

4.14 Selendroid

Selendroid is an automation framework that operates on the user interface (UI) of Android native and hybrid applications (apps) as well as the mobile web. The Selenium 2 client API is used to

write tests. Selendroid may be used on emulators and actual devices, as well as as a node in the Selenium Grid for scalability and parallel testing.

Full JSON Wire Protocol/Selenium 3 compatibility Ready. There is no need to modify the app under test in order to automate it. Using the built-in Android driver webview app, test the mobile web. The same idea applies to automating native or hybrid apps. Different locator types can be used to locate UI components. Gestures are accepted: API for Advanced User Interactions. Selendroid can communicate with many Android devices (emulators or hardware) at the same time. Existing emulators are immediately launched. Selendroid enables hardware hot plugging. Full integration with Selenium Grid as a node for scalability and parallel testing

4.15 Tricentis Tosca

Tosca, developed by the technology firm Tricentis, was designed with Agile and DevOps in mind. It is well-known for being a comprehensive solution for all testing needs, yet it is still simple to use in conjunction with any testing software that the organisation currently has. Tosca, like TestComplete, supports mobile, web, and desktop (only Windows; Mac and Linux with virtualization tools), does not require scripting, although manual code creation is still an option. The sales team will assist you with bespoke pricing, however evaluations indicate that it is on the expensive side. Given the range of Tosca's skills, the price may be justified, but you may always start with a trial to be sure.

It has many advantages. This is ideal for Continuous Integration. Tosca is an ideal fit for the Continuous Integration methodology. This is a cutting-edge method capable of significantly reducing development time and conducting multiple tests each day. Tosca may execute test cases directly from scheduling tools and then export the results as an xml file. The learning curve is moderate. Because the tool was designed to be used by non-developers, it is simple to set up and understand, allowing you to begin automating tests straight immediately.

One of its main disadvantage is that Knowledge base is limited. Tricentis' own Knowledge Base and Forum are the only active venues to post your message, therefore one won't find any community support. Although the documentation is simple to follow, one may feel limited after interacting with much more popular solutions such as Selenium.

5 Types of Software Testing Tools:

5.1 Test Management Tools

On the market today, we have access to a multitude of package testing tools. The tools we use are fully determined by the project's requirements, and we're looking for either industrial (proprietary/commercial tools) or open-source (open-source) solutions. Of course, free Testing Tools may have certain limits in the product's options list; nonetheless, it completely supported what we're searching for and if our criteria are satisfied in the free version or by purchasing a premium package of Testing Tools. The tools are classified into the following groups:

- TET (Test Environment Toolkit): TET created a test driver that met the existing and expected testing demands of the test development community. TET's functionality and interfaces were designed and developed with feedback from a diverse group of people in order to achieve this aim. [7]
- RTH (Requirements and Testing Hub): RTH stands for "Requirements and Testing Hub." This is an open source test management application that may also be used as a requirement management tool. It also includes bug tracking capabilities. [1][3].

5.2 Functional testing Tools

- Ranorex: For automated testing, this is a simple, comprehensive, and cost-effective solution. It is a superior option to other testing tools since it evaluates applications from the perspective of the user, utilising mainstream programming languages and techniques such as C and VB.net. VB.net, C, and Iron Python are the three languages that may be employed. Many commercial software businesses and corporations utilise it all around the world.
  Rational Function Tester was created by IBM in 1999. It's an automated testing tool based on object-oriented programming. It is used with the regression based testing and nowadays it is very helpful testing tools for the findings of equipment based recording tests in a very clear, precise and a concise manner. Once recorded, these scripts are invincible against subsequent script builds of any programme to ensure that new features don't interfere with past functionality. With the help of this programme, recording equipment testing, as well as white box tests for code bottlenecks, memory leaks, and assessment code coverage, may be carried. IBM made a big change to its package creation platform in 2006 to make it more accessible.

5.3 Load testing

- Load Tracer: Load Tracer is one of Trace Technologies Pvt. Ltd.'s greatest web performance testing tools. It is a very user-friendly tool which is used for performance based and load based testing of various web applications.This programme will also be utilised efficiently for online application performance testing, monitoring, and administration. This programme employs a variety of approaches to simulate a real-world user's load on an internet server. Then we can use those performance monitoring counters and collected logs to figure out what's wrong with the internet application and where the bottlenecks are[7].

6 Results and Discussion

Following a review of the literature on the following tools, a detailed description of these tools is presented in Table 1 below, based on certain criteria selected from the literature, which are endorsed by researchers and advantageous to industry practitioners in selecting tool to be used for testing. The table shows the tools that were discussed as well as the requirements that each tool met. This research would assist industry practitioners in selecting the optimum tool needed to test software, whether it is a large-scale or small-scale project.

The opensource tools are free to use and do not require a subscription. The apps and platforms supported by each tool are also shown in the table; some tools support all three platforms (Web, Mobile, Desktop), while others only support one. These examples demonstrate the kind of applications that these tools may be used to test. Some tools are simple to use and understand, while others require programming knowledge to be useful. These characteristics should be considered when selecting tools for testing in an organisation; doing so will reduce time and money spent on testing.

	Platform	Type of Testing	Languages	Usablity	GUI	Open Source	Efficiency
JMeter	Windows, Mac, Linux	Load Testing	Java	Complex	Yes	Yes	Good
JUnit	Windows, Mac, Linux	Unit Testing	Java	Complex	No	Yes	Good
Test Link	Windows, Mac, Linux	-	-	Complex	Yes	Yes	Good
Appium	Windows, Mac, Linux	Mobile Testing Tool - automated testing for Android and iOS	Python, JAVA, JavaScript, PHP, Ruby, etc.	Complex	Yes	Yes	Good
Selenium	Windows, Mac, Linux	Functional Testing of web based applications	Java, Python, C#, Perl, PHP, and JavaScript.	Simple	Yes	Yes	Good
Katalon	Windows, Mac, Linux	Continuous Testing Tool	JAVA, Groovy	Simple	Yes	No	Satisfactory
Watir	Windows	Automated Testing Tool - functional and regression testing	Ruby	Simple	Yes	Yes	Good
Soap UI	Windows, Mac, Linux	API Testing tool	Groovy and JavaScript	Fairly Complex	Yes	Yes	Good
Ranorex	Windows, Mac, Linux	Automated Testing Tool - for all sorts of testing	C#, Python, VB.Net	Simple	Yes	No	Good
Espresso	Windows, Mac, Linux	Mobile Testing Tool - automated testing for Android	Only Java and Kotlin	Simple	Yes	Yes	Good
Kobiton	Windows, Mac	Automated Testing Tool - functional and regression	Python, JAVA, JavaScript, PHP, Ruby, etc.	Simple	Yes	No	Medium
k6	Windows, Mac, Linux	Load Testing Tool performance and load testing	Javascript	Simple	No	Yes	Medium
AutoIT	Windows	UI Testing	BASIC LIKE	Simple	Yes	Yes	Medium
Selendroid	Android	UI Testing	Java	Complex	No	Yes	Poor
Tricentis Tosca	Windows, Mac , Linux, Android, and iOS	Model-Based Test Automation	Visual Basic, Java, and C#	Complex	Yes	No	Good

Table 1 Results

7 Conclusion

In this paper, we discussed about various types of Testing Techniques and Tools that can be used to test the software. We made a thorough analysis about each tool and compared them on various parameters to find out which one is better in a particular scenario.

References

[1] R. Kaur Chauhan and I. Singh, Latest Research and development on software testing techniques and tools,International Journal of Current Engineering and Technology ,vol. 4, no. 4,2014.

[2] Testing Computer Software, by C. Kaner, J. Falk, and H. Nguyen

[3] Effective Software Testing, by E. Dustin

[4] Software testing, by Ron Patton

[5] X. Wang and G. He, “The research of data-driven testing based on QTP,” in 2014 9th International Conference on Computer Science Education, 2014.

[6] M. Monier and M. M. El-mahdy, “Evaluation of automated web testing tools,” International Journal of Computer Applications Technology and Research, vol. 4, 2015.

[7] K. M. Mustafa, et al., “Classification of software testing tools based on the software Electrical Engineering (ICCEE09), 2009, pp. 229-233. testing methods,” in Proceedings of the International Conference on Computer

[8] G. Saini and K. Rai, ”Software Testing Techniques for Test Cases Generation,” International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, 2013.

[9] pankaj. Gray box testing: Software testing (2019).

[10]. Rajkumar.Kobiton tutorial - mobile testing platform with real devices (2020).

[11]. Grafana.Grafana/k6: A modern load testing tool, using go and javascript -

[12]Kaur, H. Gupta, G. (2013). Comparative Study of Automated Testing Tools: Selenium, Quick Test Professional and Testcomplete, Int. Journal of Engineering Research and Applications, 3(5), 1739–1743.