UNIT V

CHAPTER 5

Automation Testing Tools / Performance Testing Tools

University Prescribed Syllabus for the Academic Year 2022-2023

Automation Testing : What is automation testing, Automated Testing Process, Automation Frameworks, Benefits of automation testing, how to choose automation testing tools. Selenium Automation Tools: Selenium's Tool Suite- Selenium IDE, Selenium RC, Selenium Web driver, Selenium Grid. Automation Tools: SoapUI, Robotic Process Automation (RPA), Tosca, Appium.

5.1 WHAT IS TEST AUTOMATION ?

UQ. What is automated testing in software testing? How do we decide what to automate in testing and the scope of automation ? SPPU - Aug.18(Insem), Oct. 19 (Insem)

GQ. What is Test Automation ?

(a) Different terms used in automation.

(b) Skills needed for automation.

(c) What to automate.

(d) How to define scope of automation. SPPU - Aug 18 (Insem)

5.1.1 Developing Software to Test the Software is called Test Automation

- Test automation can help address several problems. Automation saves time as software can perform test cases faster than human do this can help in operation the tests immediately or unattended. The time thus saved can be used efficiently for analysis engineers to,
- Develop other test cases to realize better reporting; Perform some obscure or dedicated tests like ad hoc difficult; or Perform some extra manual testing.
- The time saved in computerization can also be utilize to expand further test cases, thereby civilizing the reporting of testing. Moreover, the computerized tests can be run overnight, saving the elapsed time for testing, thereby enabling the product to be released frequently.
- **Test computerization can free the test engineers from mundane tasks and make them focus on more original tasks -** If present are too many designed test cases that need to be run yourself and sufficient mechanization does not exist, then the test team may expend most of its time in test implementation.
- **Automated tests can be more reliable -** When an engineer executes an exacting test case many times manually, there is a chance for human error or a bias because of which some of the defect may get missed out.
- As with all machine-oriented activities, automation can be expected to produce more reliable results every time, and eliminates the factors of boredom and fatigue.

5.1.2 Automation Help in Instantaneous Difficult

Automation reduces the time gap between increase and testing as scripts can be execute as soon as the product build is ready.

Automation can be designed in such a way that the tests can be kicked off automatically, after a successful build is over.

- **Automation can protect an association against attrition of test engineers -** Automation can also be used as a information transport tool to train test engineers on the product as it has a depository of dissimilar tests for the product.
- Test computerization opens up opportunity for better operation of global property
- Manual testing require the attendance of test engineers, but automatic tests can be run round the clock, twenty-four hours a day and seven being a week.
- This will also allow team in dissimilar part of the world, in dissimilar time zone, to observe and organize the tests, thus given that round the clock exposure.
- Certain types of difficult cannot be execute without computerization

- Test cases for certain types testing such as reliability testing, stress testing, load and performance testing, cannot be execute exclusive of automation.
- For instance, if we desire to study the performance of a scheme with thousands of user logged in, there is no method one can achieve these tests exclusive of using automatic tools

5.1.3 Automation Makes the Software to Test the Software and Enables the Human Effort to be Spent on Original Testing

- **Automation means end to end, not test completing alone** Automation does not end with developing programs for the test cases. In fact, that is where it starts.
- Automation should believe all behavior such as alternative up the right produce build, choosing the right design, the stage installation, organization the tests, generate the right test data, analyze the results, and filing the defect in the imperfection repository.
- Automation is supposed to have scripts that produce examination data to make the most of coverage of permutation and combinations of inputs and predictable output for product contrast. They are called *test data generator*
- It is not constantly easy to see coming the output for all input situation.
- Even if all input situation are known and the predictable results are met, the error produced by the software and the functioning of the product after such an error may not be predictable.
- The automation script be supposed to be able to map the error patterns dynamically to conclude the result.

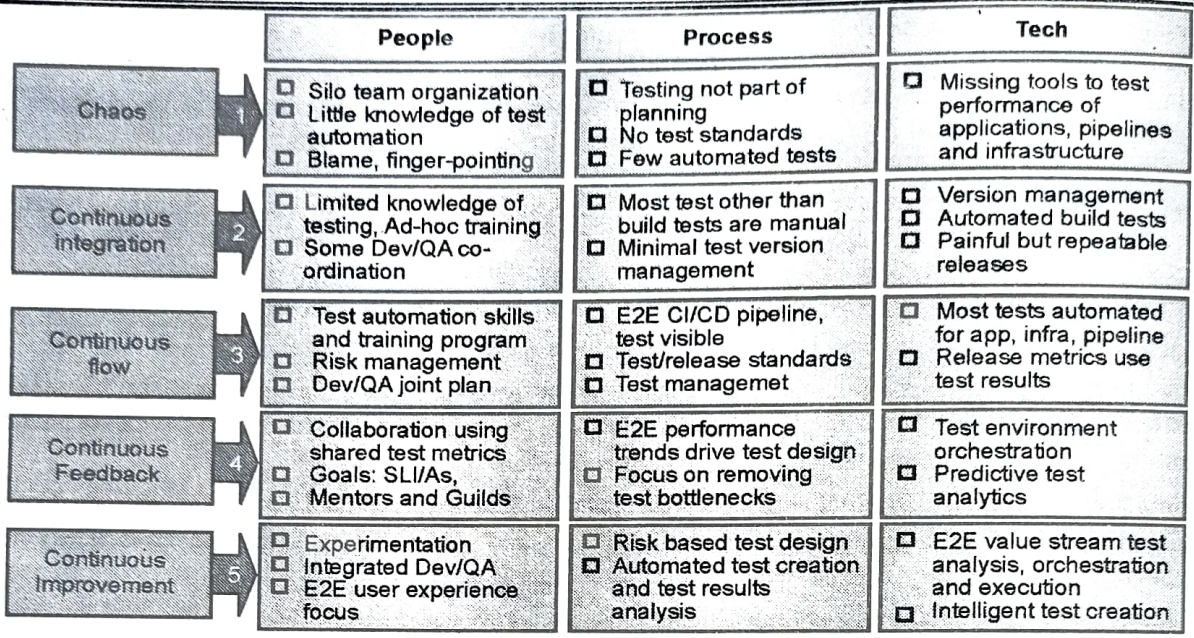
5.2 AUTOMATED TESTING PROCESS

A ***Test Case*** is a set of sequential steps to execute a test in use on a set of predefined inputs to produce assured predictable outputs.

5.2.1 There is Two Types of Test cases Automated and Manual

1. A test case (manual or automated) can be represented in many forms.
2. It can be documented when a set of simple steps, or it could be an assertion statement otherwise a set of assertions. An example of an assertion is **"Opening a file, which is already opened should fail."**
3. An assertion statement includes the expected result in the definition itself, as in the above example, and makes it easy for the automation engineer to write the code for the steps and to conclude the correctness of result of the analysis case.
4. Testing involves several phases and several types of testing.
5. This presents an opportunity for the automation code to be reused for different purposes and scenarios.

- The Fig. 5.2.1, **Continuous Test Automation Maturity Model** is a helpful tool to establish the best fit for the maturity of an organization or application within an organization.
- By advertising the kind that best match, give a visual picture of the leading level of maturity. This also is a fast way to decide areas to address to recover the stage of maturity.

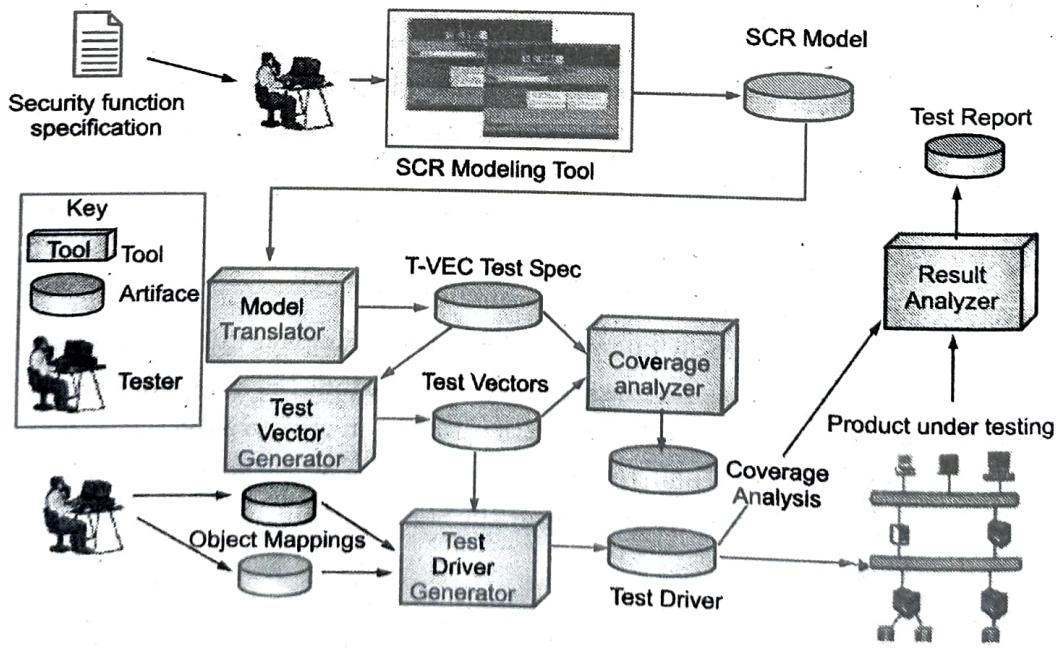


(1C1)**Fig. 5.2.1 : Constant Test Automation Maturity Model**

5.2.2 There are Two Important Dimensions

"What operations have to be tested," and "how the operations have to be tested."

- The how portion of the examination case is called scenarios (shown in Italics in the above table). "What an operation has to do" is a product-specific feature and "how they are to be run" is a framework-specific requirement.
- The automation belief is based on the fact that product operations (such as log in) are repetitive in nature and by automating the basic operations and leaving the different scenarios (how to test) to the framework/test tool, great progress can be made.



(1C2)**Fig. 5.2.2 : Framework for test automation**

- When a locate of test cases is combined and associated with a set of scenarios, they are called "test suite." A Test suite is not anything but a place of test cases that are automatic and scenarios that are associated with the test cases.
- This provides an overview of Model-Based Testing (MBT) and its activities. A categorization of MBT based on dissimilar criterion is also presented. additionally, several problems of MBT are highlighted.
- A review that provide a thorough report of how MBT is efficient in difficult dissimilar excellence attributes of distributed systems such as protection, presentation, dependability, and precision is given.
- In private communication, it would be fairly straightforward to model distributed system relationships in TTM(T-VEC Tabular Modeller) and generate vectors that could test those relationships, provided you had a tested/test environment designed to set up, control, and record your distributed systems environment in the manner of the test vectors that would result from such an approach.

5.3 AUTOMATION FRAMEWORKS

There are different *"Generations of Automation."* The skills required for automation depends on what generation of automation the company is in or desires to be in the near future.

5.3.1 The Automation of Testing is Mostly Classify into Three Generation

1. First generation: evidence and Playback
2. Second generation: Data-driven
3. Third generation: Action-driven

1. First generation : Record and Playback

- Record and playback avoid the repetitive nature of executing tests. Almost all the test tools accessible in the marketplace have the evidence and playback feature.
- A test engineer *records* the succession of actions by keyboard typeset or mouse click and those record scripts are *played back* later, in the same order as they be record.
- Since a record script can be play back multiple times, it reduces the tedium of the testing function.

2. Second generation : Data-driven

- This method helps in developing test scripts that generates the set of input conditions and corresponding expected output. This enable the tests to be frequent for unlike input and output conditions. The approach takes as much time and effort as the produce.
- However, change to request does not need the automatic test cases to be changed as long as the input conditions and expected output are still valid.

3. Third invention Action motivated

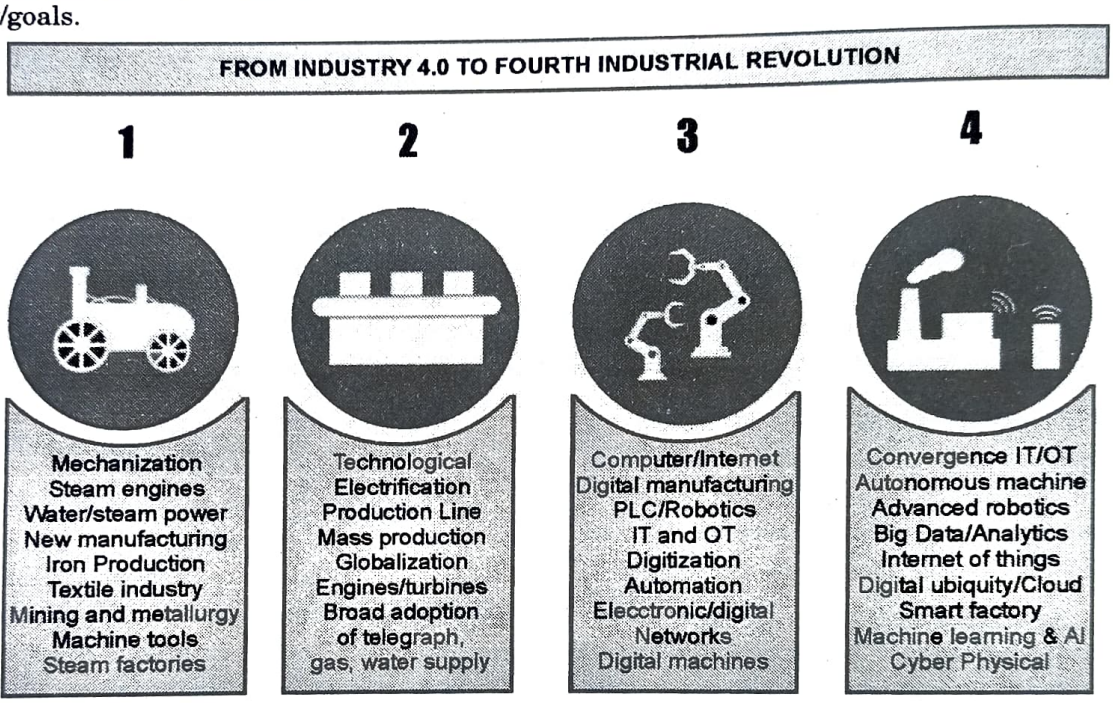
- This technique enables a layman to create automatic tests. Present are no input and expected output situation necessary for organization the tests.
- All events that appear on the request are mechanically tested, based on a general set of control defined for automation.
- The set of actions are represented as objects and those objects are reused. The user wants to specify only the operations (such as log in, download, and so on) and everything else that is needed for those actions are automatically generated.
- The input and output situation are automatically generated and used.

5.3.2 Automation in the Third Generation Involves Two Major Aspects

"Test case automation" and "framework design."

Industry 4.0 is a vision and concept in motion, with reference architectures, standardization and even definitions in flux.

Most Industry 4.0 initiatives are early-stage projects with a limited scope. The majority of digitization and digitalization efforts, in reality, occur in the circumstance of third and even second industrial rebellion technology/goals.



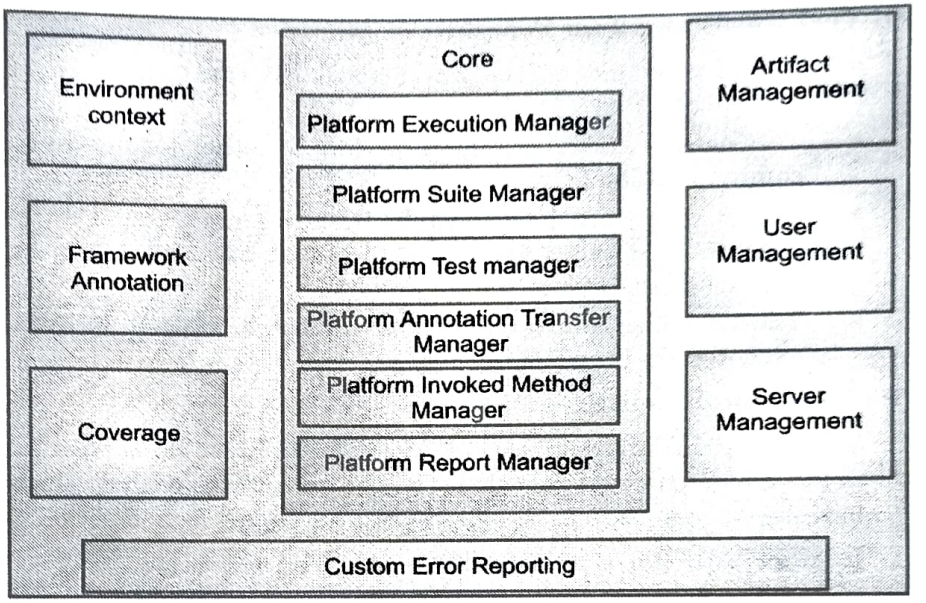
(1C3) **Fig. 5.3.1 : Industry 4.0 - digital alteration of developed in the fourth manufacturing revolution**

- In spirit, the technology creation Industry 4.0 probable control existing data and sufficient additional data sources, counting data from associated assets to increase efficiencies on numerous levels, change existing developed processes, produce end to end in sequence stream across the value chain and understand new forces and business models.
- To **know Industry 4.0,** it is necessary to observe the full value chain which includes supplier and the origins of the resources and mechanism essential for a variety of forms of smart manufacturing, the end-to-end digital supply chain and the final purpose of all developed/production, despite of the number of mediator ladder and company: the ending customer.
- Enabling more straight models of modified production, service, as glowing as purchaser/customer interaction *(including gaining real-time data from actual product usage)* and cutting the inefficiencies, irrelevance and costs of intermediaries in a digital supply chain model, where possible, are some goals of **Industry 4.0** in this customer-centric sense of progressively more difficult clients who value speed, *(cost)* efficiencies and value-added innovative services.
- In the end, it remainders business with the inventive twist of novelty and alteration of business models and process: amplify profit, reduce costs, enhance customer experience, and optimize customer lifetime value and where possible customer loyalty, sells more, and innovates to produce and stay applicable.
- The **Automation Framework Core** is the main module and contains the major structure blocks of the framework. It uses Testing spectators to systematize the test flow and execute the following operation as suitable for each phase of the test implementation :

- Server start up and minimize
- Emma instrumentation, report production, and organization
- Artifact operation and organization
- resident creation and registering
- Report generation

The **Automation Framework API** provides a test pleasant, united advance for confirmation and statement of each service. It includes the following functionality:

- Test friendly covering of the back-end service stubs.
- Test pleasant functions for confirmation and declaration of functions.
- Updates with the relevant release version.
- Encapsulate the complexity of altering all available tests in the case of a remains change.
- Maintain page object classes for Selenium mechanization.



(1C4)**Fig. 5.3.2 : Automation Framework Core**

The **Automation structure Utils** are utility mechanism that provides regularly used functionality, such as distribution SOAP and REST needs to a recognized endpoint or monitor a port to decide whether it's obtainable. Developers can just add the Utils module to their test case to get the functionality they need. Some example utilities would be:

- Axis2 client
- Wire message monitor
- Custom server starts up scenario (Axis2, Tomcat, Qpid)
- Concurrency test scenario

Testing team and developers classically preserve a set of automation tests by means of well-known mechanization tools like Selenium (for UI automation), J Meter, and Soap UI.

The **Automation Framework Tools** module supports by means of these existing scripts in your test cases and allow you to confirm and report the result of the implementation with other test cases handle by the mechanization Framework.

5.3.3 Modules

(1) **External Module :** There are two module that are external module to automation - TCDB and defect DB. To recall, all the test cases, the steps to execute them, and the history of their execution (such as when a exacting test case was run and whether it passed/failed) are stored in the TCDB.

(2) **Defect DB or *defect database* or *defect repository*** contains details of all the defects that are found in various products that are tested in a particular organization.
It contains defects and all the related information (when the defect was found, to whom it is assigned, what is the current status, the type of defect, its impact, and so on).

(3) **Scenario and Configuration File Modules :** As we have seen in earlier sections, scenarios **are nothing but information on "how to execute a particular test case."**

- **A *configuration file* contain a set of variables that are used in automation.** The variables could be for the test framework or for other module in automation such as tools and metrics or for the examination suite or for a set of test gear or for a exacting test case.
- A configuration file is important for organization the test cases for various execution conditions and for organization the tests for various input and output situation and states.

(4) Test Cases and Test Framework Modules

- A test case means the automated test cases that are taken from TCDB and executed by the framework.
- Test case is an object for completing for other module in the planning and does not symbolize any communication by itself.
- A test structure is a module that combines "what to perform" and "how they contain to be executed." It picks up the explicit test cases that are automatic from TCDB and picks up the scenario and execute them.
- The variables and their defined values are chosen up by the test structure and the test cases are executed for those values.

(5) Tools and Results Modules

When a test framework executes a deposit of test cases with a set of scenarios for the different values provided by the configuration file, the results for each of the analysis case along with scenarios and variable values have to be stored for future analysis and action.

(6) Report Generator and Reports / Metrics Modules

Previously the outcome of a test run is available; the next step is to arrange the test reports and metrics. Preparing reports is a complex and time-consuming effort and hence it should be part of the automation design.

5.4 BENEFITS OF AUTOMATION TESTING

The precise supplies can vary from product to product, from situation to situation, from time to time.

5.4.1 We Present below Some Generic tips for Identifying the Scope for Automation

1. **Identifying the Types** of Testing Amenable to Automation
2. **Certain types of tests** automatically lend themselves to automation.
3. **Stress, reliability, scalability, and performance testing**
4. **These types of testing** require the test gear to be run from a large numeral of different machines for an extended period of time, such as 24 hours, 48 hours, and so on.
5. **It is just not likely** to have hundreds of users trying out the product day in and day out
6. **Test cases** belonging to these testing types become the first candidates for automation.
7. **Regression Test :** are repetitive in nature. These test cases are executed multiple times during the product growth phases. Given the tedious nature of the test cases, mechanization will save significant time and effort in the long run.
8. **Functional tests :** These kinds of tests may require a complex set up and thus require specialized skill, which may not be available on an continuing basis.
9. **Automate these once,** with the specialist ability sets, can allow with less expert natives to run these tests on an continuing basis.
10. **This provides** an opening to automate test cases and execute them multiple times during release cycles.

5.4.2 Automating Areas Less Prone to Change

- Automation should consider those areas where requirements go through lesser or no changes. Normally change in requirements cause scenarios and new features to be impacted, not the basic functionality of the product.
- While automating, such basic functionality of the product has to be considered first, so that they be able to be used for *"regression test bed"* and *"daily builds and smoke test."*
- The non-user interface portions of the product can be automated first. While automating functions involving user interfaces-and non-user interface-oriented ("backend") elements, clear demarcation and "plug ability" have to be provided so that they can be executed together as well as executed independently.
- This enable the non GUI portions of the mechanization to be reuse even when GUI goes during change.

5.4.3 Automate Tests that Pertain to Standards

- Automating for standards provides a dual advantage. Test suites developed for standards are not only used for product testing but can also be sold as test tools for the market.
- A large number of tools obtainable in the commercial market were internally developed for in-house usage.
- Hence, automating for standards creates new opportunities for them to be sold as commercial tools.
- The certification suites are executed every time by the supporting organization before the release of software and hardware. This is called *"certification testing"* and requires perfectly compliant results every time the tests are executed.

5.5 HOW TO CHOOSE AUTOMATION TESTING TOOLS

5.5.1 Selecting the Test Tool is an Important Aspect of Test Automation for Several Reasons as given below

1. **Free tools are *not well supported and get phased out* soon.** It will be extremely dangerous to see a release stalled because of a problem in a test tool.
2. **Developing in-house tools *takes time*.** Even though in-house tools can be less expensive and can meet needs better, they are often developed by the personal interest shown by a few engineers
3. **Test tools sold by vendors are expensive.** In absolute dollar terms, the standard test automation tools in the marketplace are expensive.
4. **Test tools require strong training.** Test automation cannot be successful unless the people using the tools are properly trained.
5. **Such training usually involves** getting familiar with the scripting languages that come with the tool, customizing the tool for use, and adding extensions or plug-ins for the tool.
6. **Test tools generally** do not meet all the requirements for automation. Since tools are meant to be generic, they may not fully satisfy the needs of a particular customer. That is why customization and extensibility become key issues.
7. **Not all test tools run on all platforms.** To amortize the costs of automation, the tools and the automated tests should be reusable on all the platforms on which the product under test runs. Portability of the tool and the scripts to multiple platforms is therefore a key factor in deciding the test automation tool.

5.5.2 Criteria for Selecting Test Tools

UQ. What are the selection criteria of automatic difficult tool?

SPPU - Nov./Dec.18(Endsem), Oct. 19(Insem)

The categories are:

1. Meeting requirements	2. Technology expectations
3. Training/skills	4. Management aspects

1. Meeting Requirements

- Firstly, there are plenty of tools obtainable in the marketplace but rarely do they meet *all* the requirements of a given product or a given organization. Evaluating different tools for different requirements involves important effort, money, and time.
- Given of the plethora of alternative accessible (with each choice meeting some part of the requirement), huge delay is concerned in select and implanting test tools.

2. Technology expectations

- Firstly, test tools in universal may not agree to test developers to extend/modify the functionality of the structure. Therefore extending the functionality requires going back to the implement dealer and involves further cost and attempt.
- Test tools may not provide the same amount of SDK or exported interfaces as provided by the products. Very few tools obtainable in the marketplace supply source code for extend functionality or fixing a number of problems. Extensibility and customization are significant potential of a test tool.
- Secondly, a good quality number of test apparatus need their libraries to be connected with creation binaries. When these libraries are connected by means of the source code of the product, it is called *instrumented code.*
- This causes portions of the testing be repetitive after those libraries are distant, as the results of confident types of difficult will be dissimilar and better when those libraries are impassive.

3. Training skills

- While test tools require plenty of training, very few vendors provide the training to the required level. Organization level teaching is wanted to organize the test tools, as the user of the test suite are not only the test team but also the progress team and other area like configuration management.
- Test tools suppose the user to study new language / scripts and may not use standard language / script.
- This increases skill necessities for mechanization and increase the need for a learning curve inside the organization.

4. Management aspects

- A test tool increases the system requirement and requires the hardware and software to be upgraded. This increases the cost of the already-expensive test tool.
- When selecting the test tool, it is important to note the system requirements, and the cost involved in upgrading the software and hardware needs to be included with the cost of the tool.
- Let's talk more about evaluating test automation tools for Object Recognition for testing web applications. You'll need to evaluate which browsers are supported by each tool one by one. We normally check the top 3 popular browsers – Firefox, IE (Edge), and Chrome. Each uses a different rendering engine – Gecko, Trident, and Webkit. Because of this, automation tools need separate methods while working with them.

- The Table 5.5.1 only gives a very basic and a quick glance of the capability for Object Recognition for desktop and web, but you may require to include mobile test automation as well. For the full evaluation, we are more specific and expand into more details that are more contextual and targeted toward the software under test.

Table 5.5.1 : Glance of the capability for Object Recognition for desktop and web

	Desktop Application		Web based Application	
	Rating	Comment	Rate	Comment
Visual Studio Coded UI	Good	Can recognize most controls, except some special tabs.	Good	Can recognize almost all controls, except some special objects, for example a map. The recording feature is good. It supports IE and Firefox, but supports IE better than Firefox.
HP QTP/UFT	Poor	Can recognize few controls	Bad	Cannot recognize controls needed for meaningful scripts
TestComplete	Average	Can recognize some controls. More than QTP but less than Coded UI.	Poor	Can recognize some controls, Many use coordinate positions
Selenium (Webdriver)	Not supported	Does not support	Excellent	With Selenium, almost all controls can be recognized
Ranorex	Average	Can recognize few controls.	Poor	Can recognize some controls. Many use coordinate positions
Silktest	Poor	Can recognize few controls	Good	Same as Coded UI, recognized most controls, The recording feature is good with IE

5.6 INTRODUCTION TO SELENIUM TESTING ?

UQ. What Is Selenium Tool ? SPPU - Aug 19 (Insem), May 18 (Endsem)

- **Selenium** is a free Open source functional Testing tool used for testing web applications on multiple browsers and multiple operating systems (Platforms). It is used for Functional and Regression Testing. Testing done by the selenium tool is usually referred to as Selenium Testing.
- SQA includes all software growth procedures starting from important necessities to coding until issue. Its prime goal is to ensure quality.

5.6.1 Brief History of the Selenium Project

UQ. Brief History of Selenium Project ? SPPU - Aug 2017(Insem), May 18(endsem)
UQ. What is Selenium Project ? SPPU - Nov./Dec.19(Endsem)

- Selenium (the testing framework, not the mineral you get from eating clams) came from? Here's a short history of the technology, from its origins more than a decade ago as a proprietary tool through the present era of Webdriver.

- Selenium originated in elder days - by which I mean 2004 - as a tool for testing web applications. It was developed by Jason Huggins, a programmer at Thought-Works.
- That Selenium originated at Thought-Works is interesting. While no one in 2004 was talking about Agile infrastructure, Thought-Works was the place where Martin Fowler made his career. Fowler went on to become one of the major thought leaders behind the migration to micro services.
- While Fowler can't take credit for Selenium, it seems fitting that the tool, which is an important part of automated testing for Develops-inspired workflows today, originated in the same place from which the Agile infrastructure revolution later emerged.
  - **Open Source Selenium :** At first, Selenium was used only internally by Thought Works employees. But that changed by the end of 2004, when the tool was open-sourced. I don't know exactly when Selenium became an open source tool, since the earliest emails relating to the open-sourcing do not seem to exist anymore (at least not publicly). But it was apparently no later than late November 2004.
  - That's when the first extant Selenium development emails were being exchanged, and people were talking about checking out the Selenium code via Subversion. Selenium at this point remained an imperfect tool. It suffered from some bugs that affected testing for certain browser environments.
  - And thanks to the inefficiencies of the waterfall-style software development practices of the time, bug fixes were slow to reach users. As a Selenium user noted on November 29, 2004.Last but not least, the move placed Selenium within the rapidly growing stack of open source applications at the time.
  - The early 2000s were the era when companies like Red Hat were showing that Linux and other software that was given away for free could have huge commercial value.
  - It was also when open source web browsers, namely Mozilla Firefox, and word processors, such as Open Office, were giving closed-source tools a run for their money. And open source web browsers like Apache HTTP had already held a majority of market share for years. Against this backdrop, open-sourcing Selenium only made sense.
  - Selenium Grows Up : Within a year of its release as an open source tool, Selenium had evolved significantly. By October 2005, developers were dreaming up ambitious "grand plans" for the tool. They envisioned adding sophisticated new features that would extend Selenium beyond its original mission as a basic web app testing tool. Those included things like support for testing framed applications and cross-platform testing. The evolution of Selenium during this period was helped by the fact that Jason Huggins, the original Selenium developer, moved in 2007 to Google, where he was able to continue work on the tool.
  - Selenium Meets Web-driver Simon Stewart developed another testing tool for web apps called WebDriver. WebDriver's debut signaled a desire for features that were not available in Selenium. And for a short time, the two tools competed.
  - But in 2011, the projects were merged to form one web testing tool to rule them all. The combination of Selenium and Web Driver became Selenium 2.0, which debuted in July 2011. The new release paired the Web Driver APIs that are familiar to Selenium users today with the original Selenium feature set. Jason Huggins & Simon StewartS
  - Selenium Present, Selenium Future : The demands of automated testing continue to change. It's a safe bet that Selenium will, too.
  - One important trend that is likely to shape Selenium development going forward is the demand for ever-more efficient automated testing. The Selenium ecosystem has offered some automation options for a while thanks to Selenium Grid and other tools.

- But as the movement increases pressure on development teams to test and deliver software even faster than they already do, techniques for speeding tests, such as by offloading them to the cloud and running them in parallel, will remain key.
- Shift-left testing has also become an important part of the automated testing conversation. Selenium is already well suited for shift-left testing, which refers to the practice of performing tests earlier in the development cycle, in order to identify bugs before they slow development. But Selenium users have to choose to take advantage of Selenium in the right way for this purpose.
- Optimizing Selenium today is easier thanks to a rich ecosystem of plugins and integrations that simplify the task of working Selenium into the software delivery pipeline.
- Since development workflows are now more complex than they have ever been, and will probably grow yet more complex over time, the ecosystem surrounding Selenium is poised to remain essential in helping Selenium to remain relevant for modern application testing.

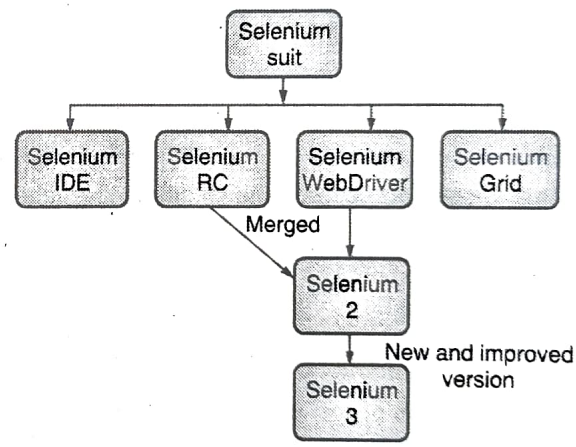
5.7 SELENIUM'S TOOL SUITE

UQ. What is Selenium's IDE explain in detail. SPPU - Oct. 18 (Insem)

UQ. Define Selenium's Tool Suite. List and explain Core Components ? SPPU - Oct. 19 (Insem)

5.7.1 Selenium IDE

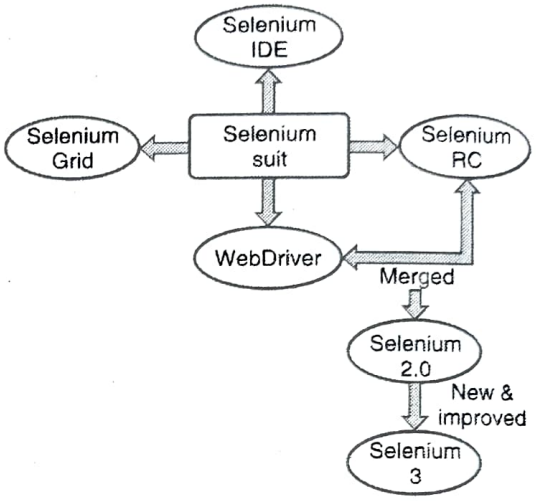
- Selenium IDE (Integrated Development Environment) is an open source web automation testing tool under the Selenium Suite.
- Unlike Selenium WebDriver and RC, it does not require any programming logic to write its test scripts rather you can simply record your interactions with the browser to create test cases. Subsequently, you can use the playback option to re-run the test cases.
- Selenium Integrated Development Environment (IDE) is the simplest framework in the Selenium suite. It is a browser plugin to record and playback the operations performed on the browser.



(1D1)**Fig. 5.7.1 : Selenium Suite Tree Diagram**

- Selenium IDE plugins are available for Chrome and Firefox browsers. It doesn't support the programming features. Selenium is the language which is used to write test scripts in Selenium IDE.
- As a Firefox plugin, Selenium Integrated Development Environment (IDE) can be used to create a test script prototype quickly and easily. It can record human testers' actions as a script while the tester runs the test case manually.
- Selenium IDE is a rapid prototyping tool for building test scripts within very less amount of time.
- It allows you to record, edit and debug the test case by providing the very simple to use components. This tool will be most helpful for beginners to learn the commands used by selenium while recording the test case. Although it was available as Firefox addon for a long time, it also available on chrome recently.
- With this tool, you can easily record the test case and able to play back any number of time whenever you will require.

- You can easily export the recorded scripts as reusable scripts in one of any programming languages that support.
- Although it was very simple and easy to use tool for beginners, it's having some limitations. It only allows you to record and playback very simple test cases.
- It will not possible to test dynamic websites or web applications. It's neither be scripted using any programming logic not support data-driven testing.
- The recorded test script can be executed at a later point in time for the regression test automatically. This tool can access the browser's DOM elements with the use of JavaScript. It also provides a flexible interface for testers to create or update test cases.
- Thought Works Company introduces selenium IDE in 2006 andimplemented in the Firefox browser, which provides record and playback functionality to the test scripts.
- Selenium-IDE is the simplest tool of Selenium community. Selenium-IDE allows software testers to export recorded scripts in many languages like HTML, Java, Ruby, PHP, Python, C#, and Test-NG.
- Selenium-IDE supports six locators, i.e., - Id, Name, X Path, CSS Selector, Link Text, DOM.



(1D2)**Fig. 5.7.2 : Selenium Suite**

☞ **Pros**

1. It is simple, easy to install and use.
2. Built-in test results reporting and help modules
3. Test Cases can be exported to usable formats in the Selenium RC and WebDriver
4. No previous knowledge of programming needed, though basic knowledge of HTML and DOM required.
5. Can easily export the recorded tests in different programming languages such as Ruby, Python, etc.

☞ **Cons**

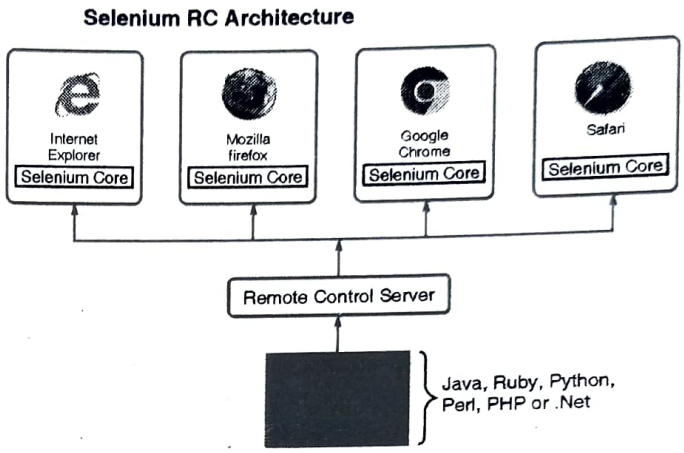
1. It is only available for Firefox.
2. The execution of test cases is slow as compared to RC and WebDriver.
3. Data-driven testing is not supported.
4. It is not able to test dynamic web applications.

5.7.2 Selenium RC

UQ. What is Selenium's RC explain in detail. **SPPU - May 18(Endsem)**

- Selenium RC is the main feature in the Selenium. A tester can use it to simulate user actions such as input data, submit a form, and click a button in web browsers.
- Selenium RC was the first tool used on selenium project. It was the core application written in Java as a programming language.
- This tool will accept commands for the browser via HTTP request. It consists of two components which are selenium RC server and RC client.
- Where RC server will communicate with HTTP/GET/POST request while the RC client will include programming codes.

- You will be able to write an automated test in most of the programming languages such as Java, JavaScript, Ruby, PHP, Python, and Perl and C #.
- Selenium RC is the first open-source tooling in selenium community which is introduced by Thought Works Company in 2004.
- Selenium RC doesn't have a record and playback features. Selenium RC cannot execute test script with selenium server.
- Selenium-RC supports multiple languages (java, C++, python), multiple operating systems (Windows, Linux), and multiple browsers (internet explorer, Google Chrome).
- Although selenium RC was the main project for a long time until the selenium 2 was released. It was officially deprecated when selenium version 2 was released. It will support mostly in maintenance mode that will provide some feature that is not available in Selenium 2.
- When a web browser is being loaded in a test script, it injects a suite of JavaScript (JS) into it. Then use those JavaScript's programming to interact with the different web browsers.



(1D3)**Fig. 5.7.3 : Core Components of Selenium RC**

☞ **Usage of Selenium RC**

1. Tester writes a test case script with the supported programming language API.
2. The test script sends a command to the RC server.
3. RC server receives these commands and triggers selenium core from executing the commands and interacting with the browser page web elements.

☞ **Pros**

1. It supports cross-browser testing.
2. It supports data-driven testing.
3. Execution speed is more as compared to IDE.
4. It supports conditional operations and iterations.

☞ **Cons**

1. Slower execution speed as compared to Web-Driver.
2. Browser interaction is less realistic.
3. Programming knowledge required.

5.7.3 Selenium Web-Driver

UQ. What is Selenium's Web-Driver explain in detail. SPPU - May 19(End sem)

- Web-Driver is the new feature added in the Selenium 2. It aimed to deliver an easy and helpful programming interface to resolve the limitations of Selenium RC programming API.
- Different from RC, Web-Driver uses browser native support to interact with the web pages.
- So different browsers have different web driver libraries and different features too. All these are decided by the web browser that runs the test cases.
- The implementation of Web-Driver is much more related to the web browser.

So, are the following Web-Driver Drivers ?

- **HttpUnit Driver :** This is one of the fastest and reliable Web-Driver implementations. Based on the HttpUnit, it can run across Linux, Windows, and Mac because of its pure java implementation.
- **Firefox Driver :** It is easy to configure and use. It is being used to run the test scripts in the Firefox web browser and does not require extra configuration to use.
- **Chrome Driver :** It is being used to run a test script on the Google Chrome web browser that needs more configurations to use.
- **Internet Explorer Driver :** It is being used to run the test script in the Internet Explorer web browser that needs more configurations to use. It can only run in Windows OS, slower than the Chrome and Firefox Web Driver.
- Selenium Web-Driver is also called Selenium-2, and Google introduced it in 2008. Selenium Web-drivers is just a collection of core java interface.
- In comparison to Selenium RC, Selenium web driver is more powerful and faster tool because it directly calls to the web browser. Web-driver supports multiple browsers, multiple operating systems, and multiple languages.
- Selenium WebDriver (Selenium 2) is the successor to Selenium RC and is by far the most important component of Selenium Suite.
- Selenium WebDriver provides a programming interface to create and execute test cases. Test scripts are written in order to identify web elements on web pages and then desired actions are performed on those elements.
- Selenium WebDriver performs much faster as compared to Selenium RC because it makes direct calls to the web browsers. RC on the other hand needs an RC server to interact with the web browser.
- Since, WebDriver directly calls the methods of different browsers hence we have separate driver for each browser. Some of the most widely used web drivers include:
  - Mozilla Firefox Driver (Gecko Driver)
  - Google Chrome Driver
  - Internet Explorer Driver
  - Opera Driver
  - Safari Driver
  - HTML Unit Driver (a special headless driver)

☞ **Pros**

1. No separate components such as the RC server are needed.
2. Execution time is faster as compared to Web-Driver and RC.
3. It supports testing on different platforms such as Android, iOS, Windows, Mac, and Linux.

☞ **Cons**

1. No mechanism to track runtime messages.
2. Image testing is not available.
3. Prior knowledge of programming required.

5.7.4 Selenium Grid

UQ. What is Selenium's Grid explain in detail. **SPPU - May 18(End sem)**

- With the Selenium Grid feature, test scripts can run on multiple machines at the same time, which reduces the total test scripts run time.

- Thus helps to find the bug more quickly because the test cases run more quickly. This is suitable for a large application with too many test scripts to run.
- You can also choose to run test scripts on different web browsers and on different machines. You can configure the browser version, Operating System, and machine to run the test case by using the Selenium RC capabilities.
- Selenium grid is the part of selenium version1 that combined with selenium RC to scale for large test suit and able to run tests on remote machines.
- We can execute multiple test cases at the same time on different remote machines. If you run your test cases on multiple environments, you will use the different remote machine to run the tests at the same time.
- While testing with selenium grid, one server acts as the hub where other machines contact the hub for obtaining browser access.
- This ability to run test cases on remote machine may be most helpful for spreading the testing load across several machines having different environments or platforms. So this feature of selenium grid will help you to speed up your test automation process.
- Selenium web driver is the latest addition among the tools in selenium tool suite. It is the upgraded version of Selenium RC. So it is much faster than selenium RC since it makes direct calls to the browsers.
- Unlike the selenium RC, selenium webdriver does not require any special server for running or executing test cases.
- Selenium webdriver supports all of the main browsers such as Mozilla Firefox, Google Chrome and Internet Explorer and Safari. It allows you to write test scripts on different programming languages such as Java, C#, Ruby, Python, Perl etc.
- Although it overcomes all the limitations of Selenium RC, generating a detailed test report is not possible with selenium webdriver yet.
- Selenium Grid is also an important tool of Selenium Suite, which allows us to run our tests script on different machines against different browsers simultaneously.
- Selenium Grid proceeds from the Hub-Node Architecture to achieve parallel execution of test scripts.
- Selenium Grid is divided into two parts:
  - **Grid-1 :** Grid-1 introduced by Thought works company in 2004.
  - **Grid-2 :** Google Company introduced it in 2008.

☞ **Pros**

1. Selenium Grid offers tools needed to diagnose the failures and rebuild a similar environment for the new test execution.
2. Selenium Grid saves time extremely as it uses the Hub-Node design.
3. It supports the simultaneous execution of test cases in multiple browsers and environments.

☞ **Cons**

1. The code executes only on the local machines where the test cases are launched.
2. Considerable efforts and time are must for the initial operation of parallel testing.
3. The remote machine only receives the browser control commands.

5.8 TEST DESIGN CONSIDERATIONS

UQ. What are Selenium's Test Design conditions in detail. **SPPU - May 18(Endsem)**

- The information we provide in this chapter is useful for newcomers and experienced veterans in the field of automated testing.
- This article describes the most common types of automated testing and also describes "design patterns" that can enhance the maintainability and scalability of your automated test suite.
- Experienced automated test engineers who have not yet used these technologies will be more interested in these technologies.

☞ Test type

- What parts of the application should you test? It depends on the various influencing factors of your project: user expectations, time limit, priorities set by the project manager, etc.
- However, once the project boundary is defined, as a test engineer, you must make a decision about what to test.
- In order to classify the type of testing of web applications, we have created some terms here. These terms do not imply standards, but these concepts are typical for web application testing.

☞ Test static content

- Static content testing is the simplest test used to verify the existence of static, unchanged UI elements. E.g. :
  - Every page has its expected page title? This can be used to verify that the link points to an expected page.
  - Does the application's homepage contain an image that should be at the top of the page?
  - Does each page of the website include a footer area to display the company's contact information, privacy policy, and trademark information?
  - Is the <h1> tag used for the title text of each page? Does each page have the correct header text?
- You may or may not need to perform automated tests on the content of the page. If your web content is not easily affected, it is sufficient to test the content manually. If, for example, the location of your application files is moved, content testing is very valuable.

☞ Test link

- A common error on Web sites is invalid links or links to invalid pages.
- Link testing involves clicking on individual links and verifying that the intended page exists. If the static link does not change often, manual testing is sufficient.
- However, if your web designer frequently changes links, or files are redirected from time to time, link testing should be automated.

☞ Function test

- In your application, you need to test specific functions of the application, require some type of user input, and return a certain type of result.
- Usually a functional test will involve multiple pages, a form-based input page, which contains several input fields, submit "and" cancel "operations, and one or more response pages.
- User input can be through text input fields, check boxes, Drop-down list, or any other browser supported input.
- Functional testing is usually the most complex type of test that requires automated testing, but it is also usually the most important.

- Typical tests are logging in, registering website accounts, user account operations, account setting changes, complex data retrieval operations, etc.
- Functional testing usually corresponds to your application's description of application characteristics or designed usage scenarios.

☞ Test dynamic elements

- Usually a web page element has a unique identifier, which is used to uniquely locate the element in the web page. Usually, the unique identifier is implemented with the HTML tag's 'id' attribute or 'name' attribute.
- These identifiers can be a static, i.e. immutable, string constant. They can also be dynamic production values, which change on every page instance.
- For example, some web servers may name the displayed file as doc3861 on a page instance, and display it as doc6148 on other page strengths, depending on the 'document' the user is retrieving.
- The test script that verifies the existence of the file may not be able to locate the file with the same ID.
- Normally, dynamic elements with varying identifiers exist on the results page based on user actions, however, obviously this depends on the web application.

  Below is an example.

```
<input id="addForm:_ID74:_ID75:0:_ID79:0:    checkBox" type="checkbox" value="true"/>
```

This is an HTML marked checkbox,

Its ID (addForm: _ID74: _ID75: 0: _ID79: 0: checkBox) is a dynamically generated value. The next time this page is opened, the ID of the check box may be a different value.

☞ Ajax test

- Ajax is a technology that supports dynamically changing user interface elements. Page elements can be changed dynamically, but do not require the browser to reload the page, such as animations, RSS feeds, other real-time data updates, etc.
- Ajax has countless ways to update elements on a webpage. But the easiest way to understand AJAX is to think of it.
- In Ajax-driven applications, data can be retrieved from the application server and then displayed on the page without reloading the entire page. Only a small part of the page, or only the element itself is reloaded.
- When to use assert command and when to use verify command? It depends on you. The difference is what you want the test program to do when the check fails.
- Do you want the test to terminate, or do you want to continue and simply log the test failure?
- This needs to be weighed. If you use assertions, the test will stop when the check fails, and no subsequent checks will be run.
- Sometimes, perhaps often, this is what you want. If the test fails, you will immediately know that the test failed.
- Test engines such as TestNG and JUnit provide plugins commonly used when developing test scripts, which can easily mark those tests as failed tests.

☞ Advantages

1. You can see directly whether the inspection passed.

☞ **Disadvantages**

1. When the inspection fails, subsequent inspections will not be performed and the result status of those inspections cannot be collected.
2. In contrast, the verify command will not terminate the test. If your test only uses verification, you can be assured that-assuming no unexpected exceptions-the test will be executed regardless of whether a defect is found.

☞ **Cons**

- You have to do more work to check your test results. In other words, you will not get feedback from TestNG and JUnit. You will need to view the results in the printout console or log file.
- Every time you run the test, you need to spend time to check the output. If you are running hundreds of tests, each has its own log, which will take time. Getting feedback in time is more appropriate, so assertions are often used more often than verification.

☞ **Trade-offs**

- assertTextPresent, assertElementPresent and assertText
- You should now be familiar with these commands and the mechanisms for using them. If not, please refer to the relevant chapter. When building your test, you need to decide
- Only check the text on the page? (Verify/assertTextPresent)
- Only check if there are HTML elements on the page? That is, text, images, or other unchecked content, as long as they are related to HTML tags. (Verify/assertElementPresent)
- Need to check both the element and its text content? (Verify/assertText)
- There is no correct answer. It depends on your testing requirements. If in doubt, use assertText, because this is the most stringent type checkpoint. You can change it later, but at least you will not miss any potential failures.
- Verify/assertText is the most special type of test. Inconsistent HTML elements (tags) or text will cause the test to fail.
- Maybe your web designer often changes the page, and you don't want your test to fail when they change the page, because this is the expected periodic change.
- However, if you still need to check things on the page, such as paragraphs, title text, or images. In this case, you can use verify/assertElementPresent. This will ensure that a certain type of element exists (if you use XPath, you can ensure its existence relative to other objects on the page).
- But you don't care what the content is, you only care about a specific element, for example, a picture is in a specific position.
- With the passage of time and the accumulation of experience, how to decide to use is still very simple.
- Using the ID or name locator of an element is the most effective way in terms of test execution. It also makes your test code more readable, if the ID or name attributes in the page source code are friendly named.
- XPath statements take longer to process because the browser must run its XPath processor. In Internet Explorer 7, XPath is notoriously slow.
- It is very convenient to use the linked text for positioning, and it works well. This technique only applies to links.
- In addition, if the link text is likely to change frequently, using the <a> tag to locate the element will be a better choice.

- However, sometimes you must use XPath positioning. If a page element does not have an ID or name attribute, there is no choice but XPath positioning. (DOM locators are no longer commonly used because XPath can do better. DOM locators simply exist for legacy testing).
- Compared with ID or name attribute positioning, using XPath for positioning has a unique advantage. Using XPath (DOM), you can find an object relative to other objects on the page.
- For example, if there is a link that must exist in the second paragraph of the tag, you can use XPath to locate it. Using ID and name attribute positioning, you can only conclude that they exist on the specified page, without knowing the specific page location.
- If you have to test the image showing the company logo at the top of the page, XPath positioning may be a better choice.

Position dynamic elements

- As mentioned earlier in the Test Type section, the page IDs of dynamic elements is listed differently in different page instances. E.g,

```
<a class="button" id="adminHomeForm" onclick="return SubmitForm('adminHomeForm',
'adminHomeForm:_ID38');" href="#">View Archived Allocation Events</a>
```

- This HTML anchor tag defines a button with an ID attribute of "adminHomeForm".
- Compared to most HTML tags, this is a fairly complex anchor tag, but it is still a static tag. Each time the page is loaded by the browser, the HTML will remain unchanged.
- Its ID remains the same in all page instances, that is, this UI element always has the same identifier when the page is displayed. Therefore, the test script (Selenium Server) that clicks this button is as follows:

```
selenium.click("adminHomeForm");
```

- However, your application may generate dynamic HTML identifiers. In different web page instances, the identifier changes.
- For example, the HTML element of a dynamic page might look like this:

```
<input id="addForm:_ID74:_ID75:0:_ID79:0:checkBox"
type="checkbox" name="addForm:_ID74:_ID75:0:_ID79:0:checkBox"
value="true"/>
```

- This is a check box, both id and name attributes are addForm: _ID74: _ID75: 0: _ID79: 0: checkBox. In this case, using standard positioning, the test script should look like this:

```
selenium.click("addForm:_ID74:_ID75:0:_ID79:0:checkBox");
```

- For dynamically generated identifiers, this approach does not work.
- The next time the page loads, the identifier will be a different value, and you will encounter an "element not found" error when executing the above script.
- To correct this problem, a simple solution is to use XPath positioning instead of ID locator. Therefore, for this check box, you can simply use

```
selenium.click("//input");
```

- Or, if it is not the first text input field on the page, try a more detailed XPath statement.

```
selenium.click("//input[3]");
or
selenium.click("//div/p[2]/input[3]");
```

- However, if you really need to use the ID to locate the element, you can change to a different solution. You can capture this ID of the website before using it, for example:

```
String[] checkboxids  = selenium.getAllFields();//Collect all input IDs on page.
for(String checkboxid:checkboxids) {
if(checkboxid.contains("addForm")) {
selenium.click(expectedText);
        }
        }
```

- This method works if the ID text of only one check box on the page is "expectedText".

☞ Positioning Ajax elements

- The best way to locate and verify AJAX elements is to use the Selenium 2.0 webdriver API, which specifically addresses some of the limitations of Selenium 1.0 testing AJAX elements.
- In Selenim 2.0, you can use the waitfor() method to wait for a page element to become available.
- This parameter is a By object used by WebDriver to achieve positioning. This is explained in detail in the WebDriver chapter.
- In Selenium 1.0 (Selenium-RC), to do this requires more coding, but it is not difficult. Check the element first, if it exists, wait for a predefined period of time, and then check again.
- This is performed within the loop. If a predetermined timeout is exceeded and the element does not exist, the loop is terminated.
- Let us consider a link on the page that implements the AJAX effect (link = ajaxLink), which can be handled using a loop:

```
//Loop initialization.
for (int second = 0;; second++) {
   //If loop is reached 60 seconds then break the loop.
if (second &gt;= 60) break;
   //Search for element "link=ajaxLink" and if available then break loop.
try { if (selenium.isElementPresent("link=ajaxLink")) break; } catch (Exception e) {}

//Pause for 1 second.
Thread.sleep(1000);
}
```

- This is certainly not the only solution. Ajax is a common topic. On the user forum, look up the previous discussion and see how others have done it.

☞ Encapsulate Selenium calls

- As with any programming, you need to use tool functions to handle functions that are repeated in the test code.
- One way to avoid duplication is to encapsulate the commonly used Selenium method calls. For example, when testing, you often click on elements on the page and wait for the page to load.

```
selenium.click(elementLocator);
selenium.waitForPageToLoad(waitPeriod);
```

- In order not to repeat the above code, you can write a wrapper method to achieve these two functions.

```
/**
* Clicks and Waits for page to load.
*
* paramelementLocator
* paramwaitPeriod
*/
public void clickAndWait(String elementLocator, String waitPeriod) {
selenium.click(elementLocator);
selenium.waitForPageToLoad(waitPeriod);
}
```

"Safe operation" to determine the existence of an element

- Another common method of encapsulating Selenium is to check the presence of elements on the page before performing further operations. This is sometimes referred to as "safe operation".
- For example, the following method can be used to implement a safe operation that depends on the presence of the desired element.

```
/**
* Selenum-RC -- Clicks on element only if it is available on page.
*
* paramelementLocator
*/
public void safeClick(String elementLocator) {
if(selenium.isElementPresent(elementLocator)) {
selenium.click(elementLocator);
    } else {
        //Using the TestNG API for logging
Reporter.log("Element: " +elementLocator+ ", is not available on page - "
                +selenium.getLocation());
    }
}
```

- The above example uses the Selenium 1.0 API, Selenium 2.0 also supports secure operations.

```
/**
* Selenium-WebDriver -- Clicks on element only if it is available on page.
*
* paramelementLocator
*/
public void safeClick(String elementLocator) {
```

```
WebElementwebElement = getDriver().findElement(By.XXXX(elementLocator));
if(webElement != null) {
selenium.click(elementLocator);
    } else {
        //Using the TestNG API for logging
Reporter.log("Element: " +elementLocator+ ", is not available on page - "
                + getDriver().getUrl());
    }
}
```

- In the second example, the 'XXXX' method is a placeholder and can be replaced with an element positioning method.
- The use of a safe method depends on the test developer's decision. Therefore, if the test needs to be continued, even if you know that some elements on the page are not found, you can use the secure method and send a message with missing elements to the log file.
- This is basically equivalent to implementing verification with a reporting mechanism, rather than an assertion that terminates execution upon failure.
- However, if the element must appear on the page in order to be able to perform further operations (such as a login button on the homepage of a portal), then the security method technique should not be used.

5.9 AUTOMATION TOOLS: SOAPUI, ROBOTIC PROCESS AUTOMATION (RPA), TOSCA, APPIUM.

5.9.1 SoapUI

SoapUI is the world's leading open-source testing platform. It is the most widely used automation tool for testing web services and web APIs of SOAP and REST interfaces. It is a boon for testers to test functional and non-functional testing, such as automated testing, functional, load testing, regression, simulation and mocking without hindrance because its user interface is very simple to use. It supports various standard protocols such as HTTP, HTTPS, REST, AMF, JDBC, SOAP, etc., that exchange information in structured data such as XML, plain text or JSON, etc. with the help of network services or web APIs in a computer.

Why we use SoapUI ?

- It is an important tool to test the Web domain, and it is an open-source, cross-platform as well as language independent that supports Eclipse, NetBeans, and IDEA. It allows the testers to test functional, non-functional testing, performance testing, regression testing, compilation, and load testing on various Web services and APIs.
- It is an important tool to test the Web domain, and it is an open-source, cross-platform as well as language independent that supports Eclipse, NetBeans, and IDEA. It allows the testers to test functional, non-functional testing, performance testing, regression testing, compilation, and load testing on various Web services and APIs.

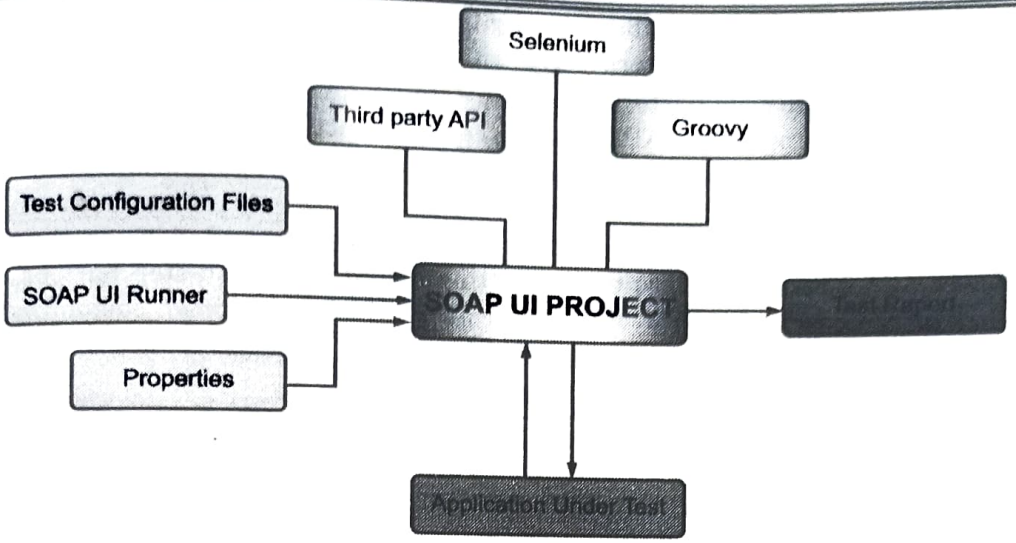


Fig. 5.9.1 : Soap-UI Architecture

☞ **Advantages of SoapUI**

(1) It provides a simple and user-friendly Graphical User Interface (GUI).
(2) Cross-platform desktop-based application.
(3) It supports all standard protocols and technologies such as HTTP, HTTPS, AMF, JDBC, SOAP, WSDL, etc.
(4) SoapUI costs less than all other test tools available in the market.
(5) It is also used as message broadcasting.
(6) It provides a fast and well-organized framework that generates lots of web services tests.
(7) It creates mocks where testers can test real applications.
(8) It supports drag and drops features to access script development.
(9) Transferring data from one response or source to different API calls without manual interaction in the SoapUI tool.
(10) It facilitates tester and developer teams to work together.
(11) SOAPUI tool provides the facility to get data from various sources of web service without developing any code.

☞ **Disadvantages of SoapUI**

(1) Security testing requires enhancements.
(2) The Mock response module should be more enhances and simplified.
(3) It takes longer to request big data and dual tasks to test web services.

5.9.2 Robotic Process Automation (RPA)

- Robotic Process Automation (RPA) is software technology that's easy for anyone to use to automate digital tasks. With RPA, software users create software robots, or "bots", that can learn, mimic, and then execute rules-based business processes.
- RPA automation enables users to create bots by observing human digital actions. Show your bots what to do, then let them do the work. Robotic Process Automation software bots can interact with any application or system the same way people do except that RPA bots can operate around the clock, nonstop, much faster and with 100% reliability and precision.

- Robotic Process Automation bots have the same digital skillset as people—and then some. Think of RPA bots as a Digital Workforce that can interact with any system or application.
- For example, bots are able to copy-paste, scrape web data, make calculations, open and move files, parse emails, log into programs, connect to APIs, and extract unstructured data. And because bots can adapt to any interface or workflow, there's no need to change business systems, applications, or existing processes in order to automate.
- RPA bots are easy to set up, use, and share. If you know how to record video on your phone, you'll be able to configure RPA bots. It's as intuitive as hitting record, play, and stop buttons and using drag-and-drop to move files around at work. RPA bots can be scheduled, cloned, customized, and shared to execute business processes throughout the organization.

☞ **Advantages of Robotic Process Automation (RPA)**

(1) Improved quality

- Research shows that performing the same tasks continuously leads to boredom, low concentration levels, and poor quality of products and services.
- Robots save employees from performing these repetitive tasks, by taking over production and accurately checking that all the products maintain high standards. Better quality products, in turn, create new and repeat opportunities for the business.

(2) Cuts down on costs

- Businesses lose a lot of income due to employees taking time off to attend to their issues, lunch breaks, sick leaves, and holidays among others.
- Robots, on the other hand, can work all year round without taking any breaks, which helps to raise production at much lower costs. They also help to save from employing other staff members to fill up positions left vacant by absent employees.

(3) Increase in sales

Freeing employees from performing repetitive tasks raises their morale making them more productive, with a better environment at the workplace and higher energy levels, there is more motivation to improve on their input, which helps to increase sales and have more satisfied clients.

(4) Eliminate risks

- Some industries have dangerous work environments that lead to injuries from chemical spills and others.
- Such industries also spend large amounts of money to ensure that the employees do not risk their health and lives while at work. By using robots to take over the dangerous and risky applications, you not only save your employees from risk, but you also save money from buying lots of security and safety equipment.

☞ **Disadvantages of Robotic Process Automation (RPA)**

1. High investment costs

- Robots do not come cheap, and the initial cost of deploying them may not be an easy thing to do. It is crucial to ensure that you have a comprehensive business plan before considering the deployment of automation at your workplace.
- You need to consider if deploying robots will increase your output and reduce expenses before deciding to put some income aside for capital expenditure.

2. **Job losses**
   - One of the main reasons people are resistant to robotic automation is the fear of job losses. On the contrary, robots do not take jobs away as there are always new roles created for the staff members.
   - Some of the most innovative robots, such as the Universal Robots cobots come with specialized programs that allow them to work alongside humans without replacing them. Robots, in most cases, take over the high risky applications while humans take on other, less risky tasks that benefit the business.
3. **Upskilling or sourcing for skilled staff**
   - Introduction of robots in the workplace comes with the need to hire skilled personnel to enable smooth operation of the machines.
   - In the majority of the cases, robot-manufacturing companies carry out the initial installation process and take the staff through training and commissioning. After that, it is either upon the business to employ skilled technicians or upskill the existing ones to keep up with the flawless working of the machines, which may attract extra costs.

5.9.3 Tosca

Tosca Automation Tool

- Being a test tool, Tosca has the ability to automate the functional and regression testing scenarios. It is also capable of mobile and API testing, which is now mandatory for any product delivery in AGILE mode.
- Tosca supports scripts less automation i.e., scripts and coding is not required to automate any scenario. So, anyone can learn the tool easily and start developing test cases. TOSCA supports its users to build efficient test cases in a methodologically way and provide detailed reports for management.

Tosca Key Features Are :

1. **Model-Based Testing Approach :** It's one of the significant features of Tosca as a test automation tool. It helps Tosca to gain leverage over other automation tools. Tosca creates a model of AUT (application under test) to create the test case without using of scripts.
2. **Risk-Based Testing Approach :** As per the name explains, this feature helps users to assess the risk with respect to test cases and allows them to identify the right set of test scripts to minimize the risks. Following different black box testing approaches such as boundary testing, equivalence partitioning, decision box, linear expansion, etc. are utilized to reduce the test script count by ensuring the functional risk coverage. After completion of test execution, risks are measured based on the test results and risk coverage.
3. **Script less test cases :** Tosca allows script less automation. It means test cases are created based on the modules which are added by drag and drop method, test data parameters, etc. after carefully incorporating the checkpoints. So, anybody can develop the test suite with minimum programming knowledge.
4. **Dynamic test data :** Test data can be stored separately in a central repository.

   Easy to the maintenance of test cases: In case of a change in application or data, it can be easily incorporated in the test suite by updating the centrally stored modules, library, and data. Distribute Execution : Tosca also provides a great feature to schedule and execute the test cases in different distributed systems in an unattended mode. It reduces the human efforts for testing.

Advantages of Tosca

1. Multiple Features in One Tool
2. No Scripting Required
3. Testing Methodology
4. Supports Multiple Platforms
5. Quality Vendor Support

Disadvantages of Tosca

1. It is highly expensive when compared with other automation tools
2. It is a heavy tool to maintain.
3. Provides less performance while scanning the application

5.9.4 Appium

- Appium is an open-source automation mobile testing tool, which is used to test the application. It is developed and supported by Sauce Labs to automate native and hybrid mobile apps. It is a cross-platform mobile automation tool, which means that it allows the same test to be run on multiple platforms.
- Multiple devices can be easily tested by Appium in parallel. In today's development area, the demand for mobile applications is high.
- Currently, people are converting their websites into mobile apps. Therefore, it is very important to know about mobile software automation testing technology and also stay connected with new technology. Appium is a mobile application testing tool that is currently trending in Mobile Automation Testing Technology.

Features of Appium

- Appium does not require application source code or library.
- Appium provides a strong and active community.
- Appium has multi-platform support i.e., it can run the same test cases on multiple platforms.
- Appium allows the parallel execution of test scripts.
- In Appium, a small change does not require re-installation of the application.
- Appium supports various languages like C#, Python, Java, Ruby, PHP, JavaScript with node.js, and many others that have Selenium client library.

Advantages of Appium

1. Appium is an open-source tool, which means it is freely available. It is easy to install.
2. It allows the automated testing of hybrid, native, and web applications.
3. Unlike other testing tools, you do not need to include any additional agents in your app to make Appium compatible with automation. It tests the same app, which is going to upload in App Store.
4. An additional feature added to Appium. Now it would support desktop application testing for windows as well along with mobile application testing.
5. Appium is a cross-platform, freely available mobile testing tool, which allows us the cross-platform mobile testing. This means you can test on multiple platforms (single API for both Android and IOS platforms).

Disadvantages of Appium

1. Along with some features and advantages, Appium has some drawbacks too, which are as follows-
2. Lack of detailed reports.
3. Since the tests depend on the remote web driver, so it is a bit slow.
4. It is not a limitation, but an overhead that Appium uses UI-Automator for Android that only supports Android SDK, API 16, or higher.

...Chapter Ends

□□□