10th Annual International Software Testing Conference in India – 2010

Test Automation using Open Source Tool - Selenium

By:

Sachin Kumar

sachink@corbus.com

Author is employed with
Corbus India Pvt. Ltd.

SDF # C-5,

Noida Special Economic Zone,

Noida – 201305. India.

Abstract

Test Automation today is developing as a separate discipline and organizations are now thinking towards automating application, to have a better cost benefit in the long run. But there are many challenges which an organization has to face when they want to automate their application. This paper outlines the below mentioned areas which will help an organization in making decisions on automation using the Open Source Tool, Selenium:

1. Challenges of Test Automation
2. Points to consider while calculating the ROI for automation
3. Can automation suite be used beyond Regression testing?
4. To what extent can you automate?
5. When to start automation?
6. Automation Life Cycle
7. Advantages and disadvantages of using Selenium
8. Selenium Vs other commercial tools
9. Test case template and checklist for writing effective test cases
10. Hybrid framework on Selenium using C#.

Challenges of Test Automation

1. Selection of Automation Tool – Today there are n number of automation tools available in the market and choosing a good automation tool is one of the major challenges that an organization often faces. This is majorly because there are several commercial tools which are expensive, there are open source tools which might not be reliable and there are tools of which an organization may not have sufficient expertise to make optimum use of.
2. No Defined Process for Executing Automation Project within an organization – Automation is like a project execution ,wherein you start with Requirement, then you design the framework as per your requirements and finally you roll it out for Test Case Execution. Lack of systematic approach and process will make successful automation really tough. As we have processes, guidelines and checklist defined for our development project; we should also have guidelines, processes and checklist available for Automation as well.
3. Availability of Right Resources – The right set of resources is a must when you are doing automation. This means the resources should be skilled enough to design and code robust scripting, so that it requires minimum time for debugging during maintenance.

4. Commitment from Customer or Management – Automation is time consuming and resource intensive task. Customer or management commitment is required if one wants to get the real benefit of automation. This paper will highlight the approach wherein you can maximize your benefits within reasonable period of time.

Overcoming the Challenges Using Selenium

Selenium is an Open Source Tool developed by ThoughtWorks. There are a set of Selenium tools which when combined provides you the power to automate simple to complex web application. Some of the Selenium Tool under ThoughtWorks umbrella are:
Selenium IDE – It is a Firefox add-on that makes it easy to record and playback tests in Firefox 2+. You can even use it to generate code to run the tests with Selenium Remote Control.
Selenium Remote Control (RC) – It is a client/server system that allows you to control web browsers locally or on other computers, using almost any programming language and testing framework.
Selenium Grid – It takes Selenium Remote Control to another level by running tests on many servers at the same time and cutting down on the time it takes to test multiple browsers or operating systems.
Bromine – It is a web-based QA tool of Selenium that enables you to easily run Selenium RC tests and view the results. It scales beautifully, from the single tester who just wants to run some tests without all the hassle; to the corporate solution, with multiple user-groups and hundreds of test cases.

Selenium is the best open source tool for doing automation for web based application and it does not have any cost attached to it. The only cost is the effort which will go for designing and developing the script for the application.

There is no need to define or develop separate Life Cycle for doing automation in Selenium. If you have an existing automation process, Selenium Automation Life Cycle will fit into that well. If you are using Selenium as an automation tool, you will find a lot of information online for the best process or Life Cycle to be adopted for automation.

Selenium scripting can be done in a number of programming languages like C#, Perl, PHP, Ruby etc, unlike other commercial tools which support single scripting language. Since Selenium scripting can be done in any language of choice, one can easily find right resources for the programming language chosen for Selenium Automation.

Last but not the least, since this tool comes at ZERO PRICE, an organization's management will find that the only investment they have made is on the infrastructure and human effort and not on the heavy licensing cost.

One of the important decision points for GO NOGO for automation is the Return On Investment factor. The next section will discuss the important aspect which you might consider while calculating your automation ROI.

Points to be considered while calculating Automation ROI

1. Number of Test Cases Planned for Automation – This is the total number of test cases that have been identified as an Automation Candidate. The test cases that should be executed multiple times with different sets of data, should be counted as number of iterations needed for such test cases. For e.g. if Security Related Test Cases has to be run for 10 roles, then test case count should be taken as 10 and not 1. The reason being, if a manual tester will execute the same set of test cases, he/she will run it for 10 roles. This is one of the very important factors as you will save a lot of effort when such test cases will be executed through automation tool. The tool will execute the above 10 roles in not more than 15 Min while a manual tester might take 50 to 60 Min.
2. Manual Execution Effort Required for Identified Test Cases – This is how much time it will take to manually execute one test case. If you multiply the time taken in executing one test case with the total number of test cases, you will have the Total Manual Execution Effort.
3. Automation Execution Effort for Automated Test Cases – This is the time taken for the tool in successfully executing one test case. You should benchmark the execution time taken by the automation tool while you take the effort. Normally simple to medium test cases will not take more than 2 Min for execution.
4. Number of Test Iteration Planned in a Year – This is one of the deciding factors for automation. As an organization how many Iteration you are planning for the application in a Year. This number when multiplied with Manual Execution Effort and Automation Execution Effort will provide the Effort on saving in a Year.
5. Automation Effort required for Test Cases Automation – This is one time effort which will go into the Automation Investment. Estimate the Automation Effort estimation on the basis of identification of Simple, Medium & Complex Test Cases and the effort required for automating each of the Simple, Medium & Complex modules. The Effort Estimation process for automation should be carried out based on an organization's standard.
6. Automation Cost – After you complete the estimation of automation, you can arrive at the cost by multiplying the effort with the cost per hour.

7. Maintenance/Support Cost - This again is based on an organization's standard. Take 5 – 10% of automation cost as maintenance or support cost when using Selenium.
8. Net Savings in a Year - The difference between the effort of manual execution and automation execution will provide the savings on effort. While calculating the Net saving in Dollar, calculate the difference between saving and spending. In first year your Net saving might be negative depending on the automation effort and cost but from the 2nd year onwards you should have a positive ROI from your automation investment and your ROI will increase year by year as your application and automation capability gets matured.

The major advantage of using Selenium is the saving in the support/maintenance cost. Since there is no License Cost involved with Selenium, so your Recurring Cost for License will be zero and your ROI will be more. The maintenance or support cost while using Selenium as an automation tool will vary between 5% - 10% of the automation effort.

The FTE cost per hour will also be less when using Selenium since the resources availability for Selenium programming language is ample.

Sample Illustration of Calculating Automation ROI (Table 1)
The below number might vary depending on an organization's productivity and the per resource cost for manual testing and automation.

1st Year Projection

Table 1

Application Name	ABC
Number of Test Cases	2000 (Sample Number)
Manual Effort (hours) to execute	168 (Considering 5 Min is required for one Test Case Execution)
Automated Effort (hours) to execute	66 (Considering 2 Min for one Test Script)
Number of Test Iteration Planned Yearly	10 (10 Iteration are planned in a Year. Each iteration will require execution of 2000 Test Cases).
Total Projected Hours Saved	(168*10) – (66*10) = 1680 – 660 = 1020
Total Projected FTE Savings Annually ($)	1020 * 60 =61,200 (Considering 60$ per hour is one FTE Rate)
Savings %	(1020/1680)*100 = 61%
Automation Effort Estimate	4000 (Considering 2 hours for Scripting one Test Case)
Automated Effort FTE Cost ($)	4000 * 60 = 2,40,000 (Considering 60$ per hour is one

	FTE Rate)
Maintenance/Support Cost ($)	5% of 2,40,000 = 12,000 (Considering 5% of Automation Effort is the maintenance/Support Cost)
Net Savings Annually	(61,200 – 2,40,000) - 12,000) = -1,90,800

From 2nd Year, you will save on Automation Effort and you have only the Maintenance Cost. The ROI from 2nd Year will be something like 61,200 – 12,000 = 49,200. Apart from that, if you are planning to increase the number of iterations, you will have more saving on effort. As you mature more in the Application Life Cycle, you have more savings on effort and overall quality of the project. The existing automation code can be reused for similar project wherein you have more ROI since your Automation get matured and productivity get increased.

Today we are majorly working on Web based applications wherein automation will really provide you good benefit not only in terms of Quality but also in terms of Effort and Cost. Since major applications are built on Web, you can really leverage the Selenium tool for driving your automation goal.

Can Automation Suite be used beyond Regression Testing

The answer to above question is "YES". You can make use of Automation Suite as a substitute for 1st Round Manual Execution not only for Regression Testing.
Many of us know that the major usage of automation is only for Regression testing which is not completely true. This paper will highlight the areas wherein you can make optimum usage of your Automation Suite not only for Regression Testing but also for Smoke Testing, Sanity Testing and even as a substitute for Manual Testing. If 50 - 80% of the test cases do not require Manual Execution even for the first time, then imagine the cost and effort benefit due to automation.
How you can substitute the Manual Execution at the very first time is elaborated in the coming section of this paper?

How much can you automate?

This is a very basic question of management. How much can we automate? The answer to this question depends on the type and complexity of the application. If it is a Simple Web based application not having much interaction with other system or environment, then you can achieve 100% of automation. If it is a Medium web application like Online Reservation System for Rail, Hotel etc, you can set automation target of 60-80%. If it is a Complex Web application, wherein it is interacting with other system or environment, then you can target to automate at least 40% to 60% of application.

Automation %age Metrics based on Complexity – Table 2

Table 2

Complexity	Target %age for Automation
Simple	100%
Medium	60 – 80%
Complex	40-60%

Simple - The Web based application is a static site with Navigation, Content Rendering, Login Feature, Role wise access, Simple Transaction and a Content Management System with well defined workflow.

Medium - The application is dynamic in nature like Online Booking System or Rail Reservation System.

Complex - The application built for Banking domain, Ecommerce Domain wherein integration with other enterprise system is involved. The data is coming through various layers and the business logic is complex.

Note: The above metrics are prepared based on our organization benchmark and may vary depending on the nature of application and underlying business.

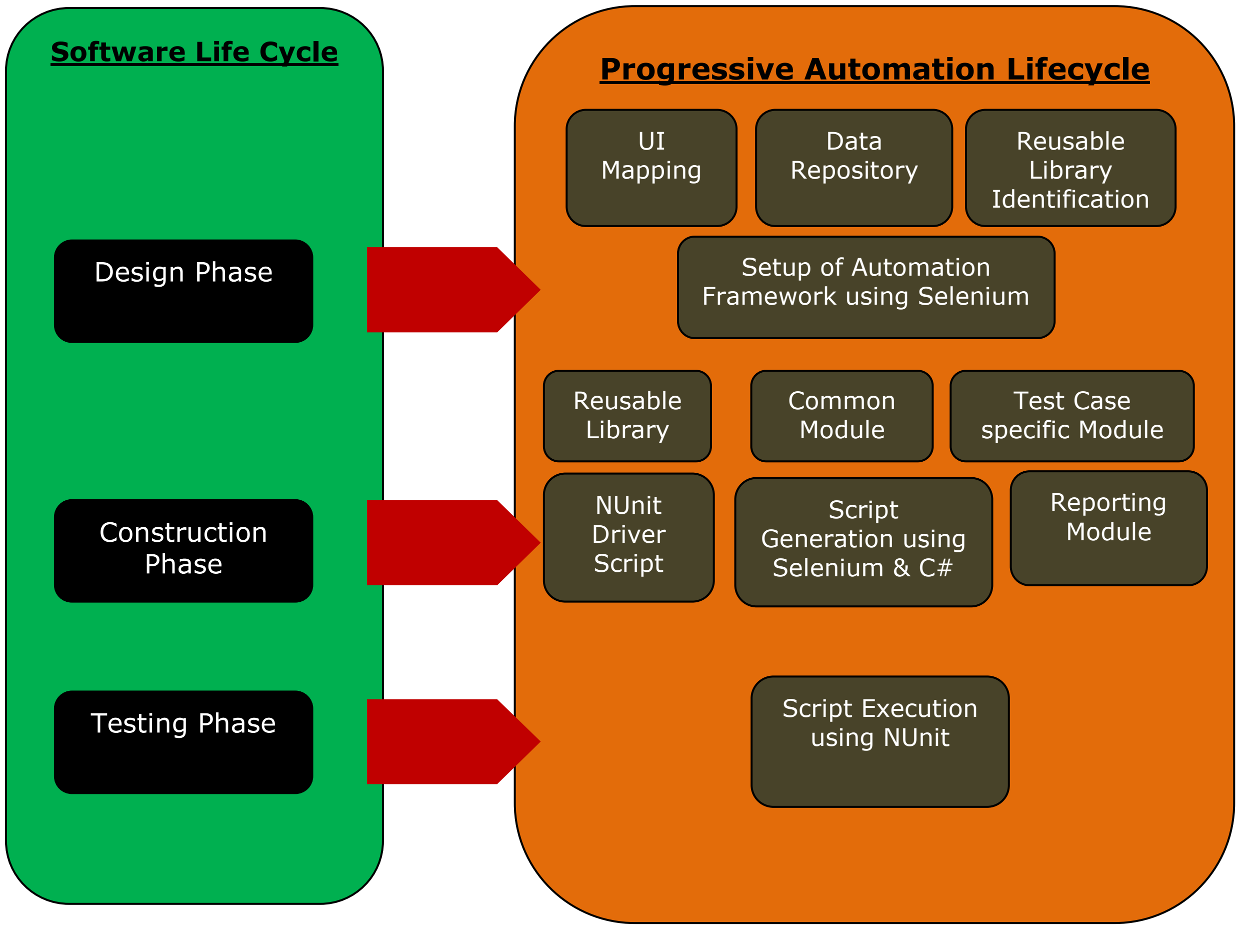
When to start Automation?

As I mentioned earlier that you can use your Automation Suite beyond regression testing. To achieve that benefit, you have to start your automation as early as possible in the Development Life Cycle (Design or Construction Phase). This again goes against normal practice when we say "Start Automation when the Application is Stable."

Progressive Automation as explained below:

The best phase to start your automation planning is during the Design Phase of the Development Life Cycle. This is during this phase that you can know about the application design in terms of screen, user control, and their properties, which make the building block of your Automation Script. During design phase you will be able to identify the UI Control and build your Object Repository for later phases. You also start looking into the test cases that have been written during Scope phase or Design phase, so that you can design your framework depending on the functionality of the system. When Construction phase comes and you have got an access to User Interface, you can start developing your Test Script which will later be updated based on any changes in the UI. This way you build your Test Script and once the Construction phase is completed and Testing phase get started, you have your Test Script ready for execution. You now require Test Execution Engineer who can run your script on the AUT. A Life Cycle for Progressive Automation is shown below: (Diagram 1)

Diagram 1


Software Life Cycle
Design Phase
Construction Phase
Testing Phase
Progressive Automation Lifecycle
UI Mapping
Data Repository
Reusable Library Identification
Setup of Automation Framework using Selenium
Reusable Library
Common Module
Test Case specific Module
NUnit Driver Script
Script Generation using Selenium & C#
Reporting Module
Script Execution using NUnit

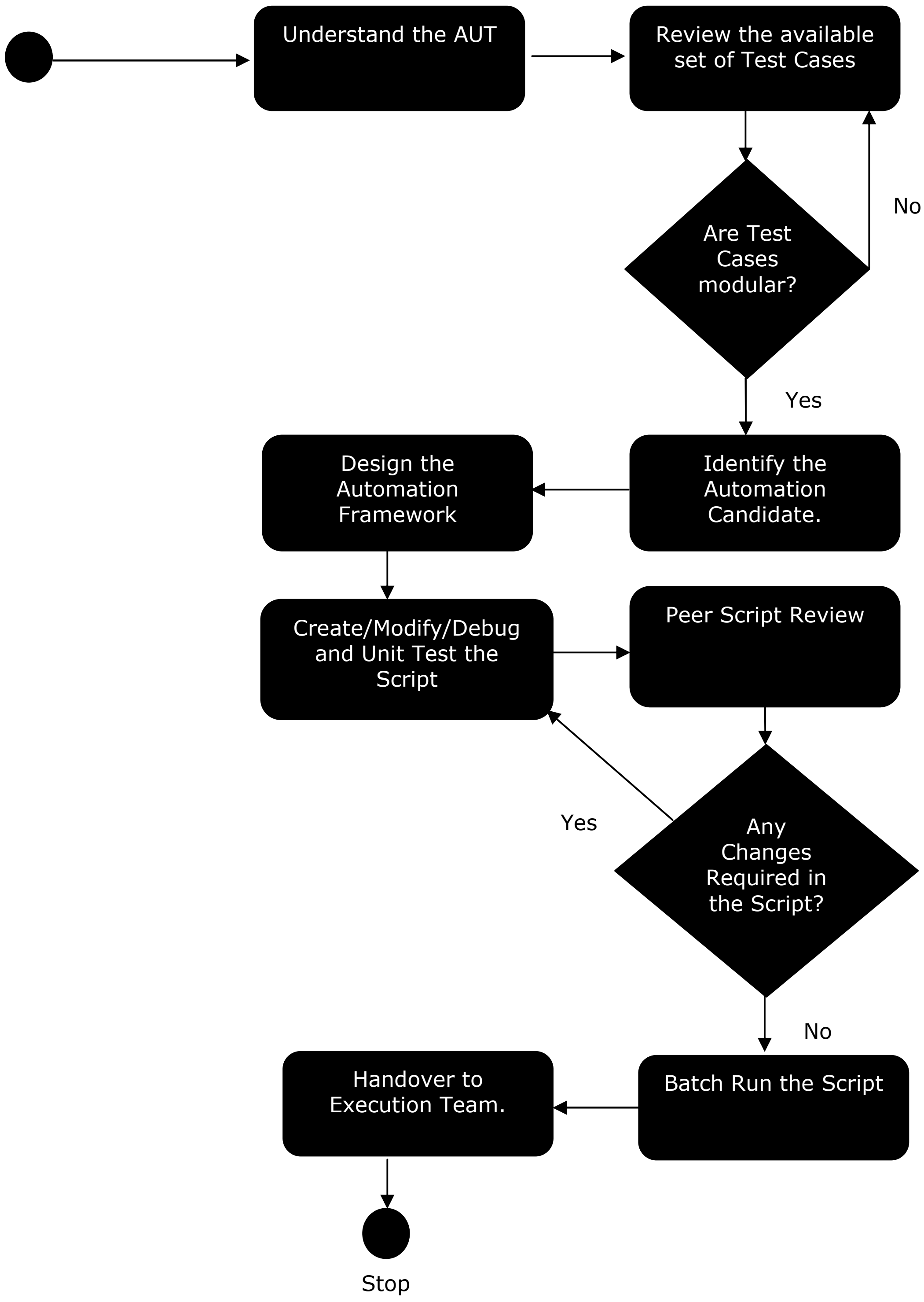
To achieve above approach we have to follow an Automation Life Cycle so that the test script delivered during Testing phase will be robust enough to run without any interruption. To ensure the robustness we have to strictly follow the below Life Cycle.

1. Understand the Application Under Test
2. Review the available test cases
3. Modify the Test Cases to make it modular
4. Identify the Automation Candidate from the List of Test Scenarios
5. Design the Framework – Data Driven, Keyword Driven, Modular or Mix of all these Hybrid Framework.
6. Start Generating the Script for AUT
7. Debug the Script and apply error handling for uninterrupted Test Execution.
8. Peer Code Review of the Script
9. Unit Test the Script

10. Batch Run the Script
11. Analyze the Result
12. Make necessary Changes based on Result Analysis.
13. Handover the Script to Test Execution Team.

An Automation Flow Life Cycle is depicted below: (Diagram 2)

<u>Diagram 2</u>


Understand the AUT
Review the available set of Test Cases
Are Test Cases modular?
No
Yes
Identify the Automation Candidate.
Design the Automation Framework
Create/Modify/Debug and Unit Test the Script
Peer Script Review
Any Changes Required in the Script?
Yes
No
Batch Run the Script
Handover to Execution Team.
Stop

While following above Lifecycle, follow the Script Design Guideline, Script Review Checklist, and the Test Case Review Checklist as per Organization Standard. This way we ensure that all the script developed follow similar approach and style.

The next sections of this paper will also cover sample of the Test Cases and Review Checklist. The Script Guideline can be developed based on the Scripting language chosen. For e.g. If one is using C# coding, then C# Coding Guideline specific to Selenium and Checklist need to be followed. Similarly in other specific programming language guideline, different checklist should be followed.

There are many advantages and disadvantages or limitation while using Selenium. Major advantages and constraint are discussed in the below section.

Advantages of Selenium

1. One of the major advantages of Selenium is that it's an open source tool without any cost.
2. It supports many languages like Java, Groovy, C#, Perl, PHP, Python and Ruby which can be used to develop your script. This support gives an edge to leverage existing skillet in any of the above language to design and develop the script.
3. Support for multiple browser and platform. You can run the same version of the script on multiple browsers and platform by designing a framework around it.

Disadvantages of Selenium

- Object scanning feature is not available, so one has to record the application in Selenium IDE and check the Object detail from the generated Selenese code.
- You cannot drive your test cases from Test Management Tool other than Bromine. You have to manually drive the test cases from the Framework using NUnit
- Selenium can only be used for automating Web Based System and cannot be used for Desktop application.

Note: Please check the Selenium release note for any limitation and work around to overcome those constraints.

Comparison between Selenium and QTP

Refer the below Table 3 for seeing the comparison details:

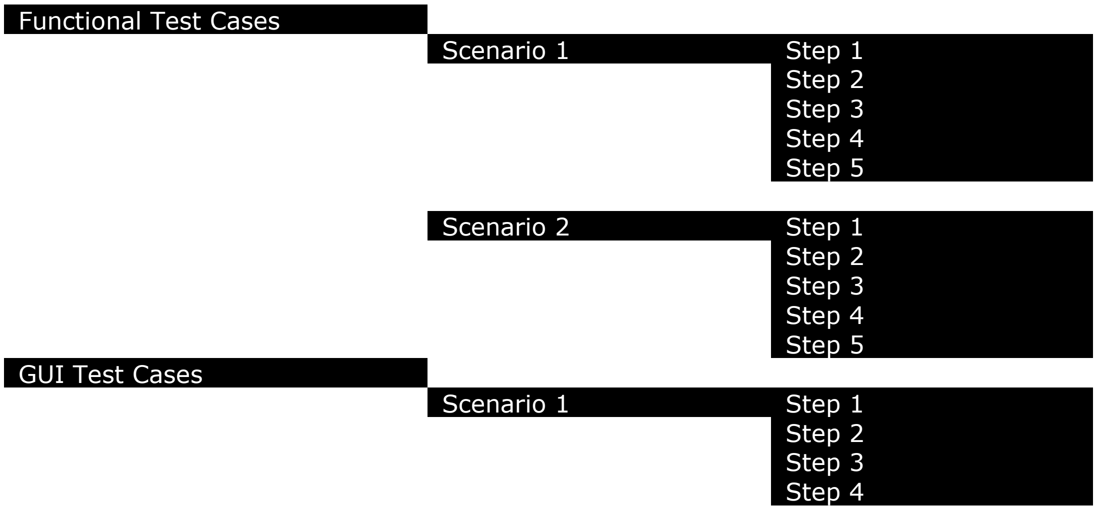
Table 2

Feature	Selenium	QTP	Comments
Recording Feature	X	X	Use Selenium IDE for recording
Playback Feature	X	X	Use Selenium IDE for Playback
Checkpoints	X	X	Use Inbuilt Selenium Checkpoints, function, methods etc for implementing verification point

In Built Report	-	X	Build Report Module in Selenium to get an Excel based Report.
Script Customization	X	X	You can add conditional logic, loops in Selenium Script when used with language of your choice.
Multiple Browser Support	X	X	Selenium provides inherent capability for running the same script on multiple browsers which is supported in QTP but complex to implement.
Multiple Language Support for Scripting	X	-	Selenium scripting can be done in C#, Perl, PHP, Ruby, Groovy etc while QTP scripting can be done only in VB Script. This 1 feature will provide the Scripter the power of Object Oriented Programming which can be used to design a robust framework even for complex application.
Modularity	X	X	QTP does support limited modularity as VB scripting is the core language use for Automation.
OOPS	X	-	Selenium choice of language gives the power of OOPS.
Extensibility	X	-	Since selenium is an open source tool and its core library file is exposed to user, so you can extend the Library to make amendment as per your requirement. QTP base code cannot be extended being a legacy tool.

Test Case Template for Effective Automation

The test case writing should follow the same pattern as followed in any Test Management Tool such as Quality Center, Salome TMF, Bromine etc. You should divide your test cases in Functional & GUI. Add the specific scenario and Test Steps to these logical grouping and also identify the Reusable piece to make your automation really effective. A sample depiction is shown below: **<u>Diagram 3</u>**



Once you are through with your Test Case Development, please follow the standard organization checklist for review. The one very important point while reviewing the test cases is to check the Modularity of the test cases. The test cases should be written in a modular way. The reusable test cases such as Login, Navigation etc should be marked as Template Test Case which will be referred while writing the test cases for other Business Function. These Reusable Test Cases can be maintained separately to for easy reference while doing automation.

In the last section of this paper, I will show you the approach of creating a Hybrid Framework in Selenium. The illustration of the Framework is shown below with explanation wherever required. This approach will give you an idea how powerful tool Selenium is, provided you know the robust implementation of it. You can make Selenium work for any type of complex scenarios by changing its Core Library file to fit your requirement.

Develop a Hybrid Framework in Selenium using C#

The framework is digrammatically shown below (Diagram 4). Normally while designing the below framework, you logically group the Data Set, Reusable Methods, Common Utility and your core C# Test Script file. The pre-requiste and steps to start with building a framework in Selenium is outlined below:

Pre-requisite

- Visual Studio 2003 with C#
- NUnit Tool
- Selenium .NET Client Driver
- Selenium IDE in Firefox
- Selenium RC
- JRE for Selenium RC
- Excel Report Module

Automation Steps

- Record the Test Steps for each Scenario in Selenium IDE
- Insert required validation point while recording the script
- Convert the Selenese Script into C#
- Copy the C# Script
- Open Visual Studio and create a project
- Add C# File and Paste the copied Script
- Add Conditional Statement as per Test Case Steps
- Call the Reporting Module wherever you are checking the Test Condition for Pass & Fail
- Compile the code and fix any compilation error.
- Add the Compiles DLLs in NUNIT
- Unit Test the Test Script
- Analyze the Result

- Turnover the Test Script for execution.

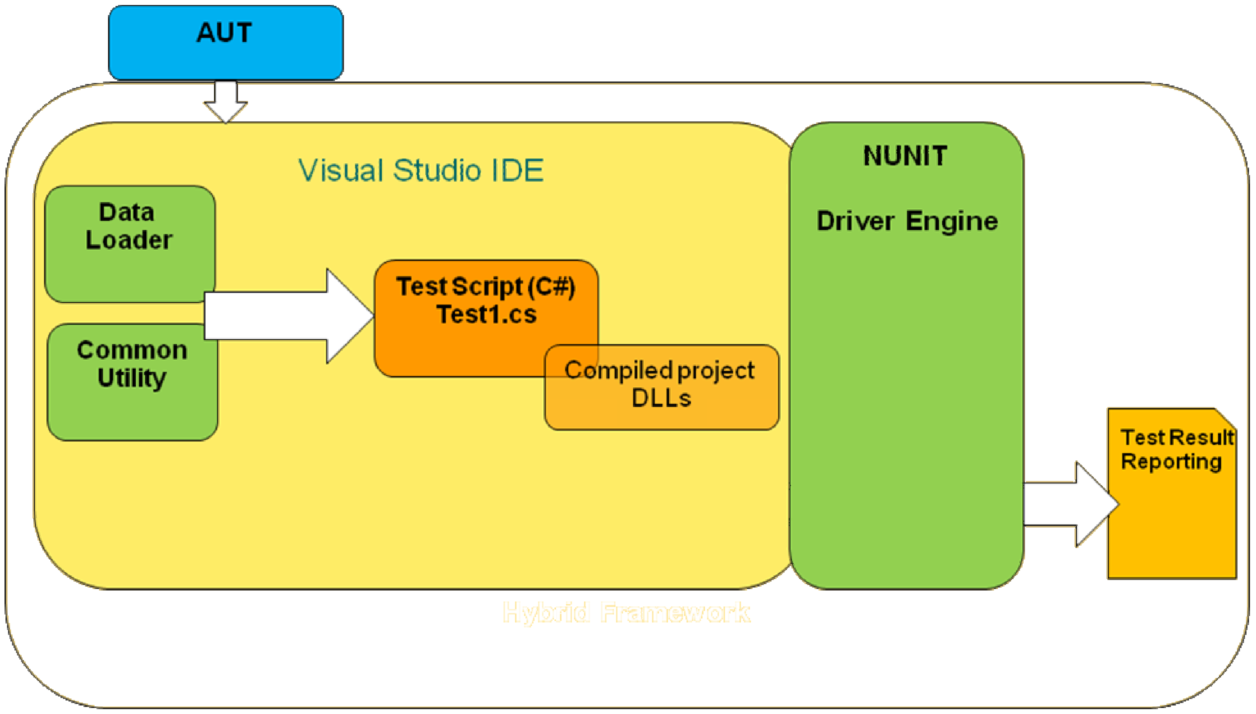


Diagram 4

References

- Selenium HQ Website

About the Author

Sachin Kumar
Project Manager Testing
sachink@corbus.com

Sachin Kumar is a Science (Mathematics) Graduate from Delhi University. He is PMP Certified and has a total experience of 9+ Years in the field of Software Testing. His experience includes both Manual and Automation Testing besides Test Management. He has played different roles like that of a Test Manager, Test Lead, Automation Architect, Onsite Coordinator, Training new personnel and member of Testing COC etc. He has designed Automation Framework for a very complex Banking Domain project which provided a good ROI to Client. He has excellent project management abilities such as handling projects, customer interaction skills, team management and is flexible in his approach. He has worked as Test Automation Architect for IBM-Hyderabad prior to Corbus. He is technically very sound and has always delighted the customers with his testing solution. He has shared many tutorial & white paper on his blog for other fellow members of Testing Community.

XXX