International Journal of Research
Available at https://edupediapublications.org/journals

p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 10
June 2016

Automation Framework for Network Management Applications

Renukaprasad P S
M. Tech Student, Dept. of IS&E,
Sri Jayachamarajendra College of Engineering,
Autonomous under VTU, Mysore, Karnataka, India

Prathibha R J
Assistant Professor, Dept. of IS&E,
Sri Jayachamarajendra College of Engineering,
Autonomous under VTU, Mysore, Karnataka, India

ABSTRACT: In today's software development lifecycle, the mantra is to deliver working and stable product over every iteration cycle. It's a challenge to meet without automation. It implies the importance of automation for the successful working products. There are different types of automation for different needs though and no single automation approach works everywhere.

The objective is to deliver complete automation coverage. This would involve designing and developing test automation suite framework for Network management application.

Design and development of framework to automate the application is carried out.

KEY WORDS: QA – Quality Assurance, AT – Automation Tool, GUI – Graphical User Interface, TS- Testing System, AUT- Application under Test, SOA- Service Oriented Architecture

INTRODUCTION:

In today's software development lifecycle, the mantra is to deliver working and stable product over every iteration cycle. It's a challenge to meet without automation. It implies the importance of automation for the successful working products. There are different types of automation for different needs though and no single automation approach works everywhere.

There is a simplistic perception about test automation with user groups not belonging to quality assurance (QA). But the Test Automation is more than a set of tests run to generate apparent results. It includes picking right automation methodology, designing test harness, implementing automated test cases, monitoring and interpreting a broad range of results. Automation by simply running test cases without human interaction is not going to catch any application issues.

Enterprise, Telecom and Finance software are complex in nature. Few of the factors which give rise to this complexity include but not limited to:

i. User interface with disparate 3rd party and legacy systems

ii. Multiple releases of systems in product families.

iii. Multiple platform and browser types and versions support

iv. Use cases are inherently complex comprising of many steps on the user interface

The number of combinations of test environment is simply enormous and many times manually testing the software is just not feasible, even if the time permits it. Test automation can automate some repetitive but necessary tasks in a formalized testing process already in place, or add additional testing that would be difficult to perform manually.

This paper provides the detailed automation implementation of Network Management a web based GUI application that manages complex network.

LITERATURE SURVEY

The literature related to the research topic has been reviewed for last few years in order to find out work carried out by various researchers. It is noticed that most of the research carried out belongs to the following categories:

1. Network Management Application:

Network management vision for the enterprise calls for a new level of synergy between people, the collaborative real-time applications they use, and the underlying, enabling network. A key building block for this vision is the foundational network. As real-time communications continue the evolution to IP, the data network becomes completely integrated into the delivery of communications-enabled business services and mission critical business applications. Networking provides advanced enterprise-class reliability, performance, and security that organizations throughout the world depend on to run their businesses. Because solutions are streamlined to better utilize and manage networking resources, a data network can uniquely deliver both mission critical dependability and superior return on investment.

- Centralized Provisioning for Networking Devices
- Automatic Detection of changes and re-provisioning of Network Devices
- Simplified Device set-up and Administration
- Enhanced Application Performance and Troubleshooting

The Right Tool for the Job

- Configuration and Orchestration Manager
- Virtualization Provisioning Service
- WLAN Management Software
- IP Flow Manager
- Visualization Performance and Fault Manager

2. Decision to automate

As per the automation industry "Involve early and test often" so it is better to identify automation at the earliest of development cycle. There are many factors which can influence the decision to automate. The key factors which are considered during the decision process are

What to automate?

A high expectation is set to automate everything as soon as an investment made on automation tool. It may sound great idea but is it achievable? Again automation must be looked at an addition to the existing testing not a replacement. With that it is imperative to know what to automate and what to leave out. While analyzing what to automate, keep it in mind that the test program must not duplicate AUT program logic. If a test is repeated only a few times, it must be left out. The criteria to pick automation candidates are as given below:

- Is TC (Test Case) repetitive?
- Is TC executed successfully?
- Does TC involve any manual steps for end-to-end automation?
- Is TC steps are readable and executable?
- Is TC test data available?
- Is TC time consuming one to execute manually?
- Is TC fit for Data Driven approach?
- Is TC covers core functionality?

When to automate?

There are certain times that automation is not advantageous at all whereas manual testing is. For an instance:

- User interface (GUI) under constant changes
- User interface (GUI) is not stable
- Short deadline for release
- In these times manual testing is best bet to take the product to market quickly. Because initial investment in automation is high and need ample time to build up a framework. Here is the list of criteria considered to decide when to automate:
- Products with long lifespan
- Project follows Agile process which insists on extreme programming
- For continuous integration and nightly builds
- Multiple and parallel releases
- Cross Browser and Platform support

METHODOLOGY

1. Automation Framework architecture

An automation framework is a set of functions, methods, objects, classes, constants, and records. To simplify this it is a collection of building blocks for an automation regression suite. It contains conceptual ideas, tools, data source, suite structures, coding standards and assumptions defined. The regression suite will be built on top of the frameworks. There are many benefits can be stated depends on the framework, but the key ones are speed and reusability. Yes it truly improves the pace of the automation development as it provides all basic functionalities required to create test scripts. There are different type of approaches can be considered while designing framework. However, it all depends up on the schedule, resources and application types. The Data Driven approach is chosen for Network Management application automation as they are naturally driven by data. The other reasons are it takes less time to develop compare to other approaches and can increase the test coverage rapidly.

2. Design Principles

The proposed design was based the following principles.

- **Plug and Play architecture**: The framework designed can be used for automating multiple applications. The user needs to get the designed framework and do some scripting to perform automating particular functionality. User should not be worrying about taking care of whole application, generating report and logs .
- **Highly portable and configurable**: Framework is highly transferrable. It can be used for many applications. It can be customized as well based on the user requirement. So configuration based on the need is provided in the designed framework.
- **Data Driven**: Suppose similar kind of test cases are to be repeated multiple times with just the change in the data. Such cases can be data driven from the xls sheet. Framework should handle those things.
- **Readability and Maintainability**: The scripting done with framework is readable. i.e step by step execution of test cases is done guided by logs.

If any changes to the application are done, the framework contents have to be maintained. It can be effectively handled in our framework.

- **Single suite for multiple solutions:** The suite can be used for multiple solutions. This suite can be used for regression testing, sanity testing etc.
- **Consistency and Reliability**: The suite is reliable and consistent in the operation. It has got recovery system as well.
- **Scalable:** The suite is scalable to new features added to the application. It can also be scaled to load, longevity testing calls.

3. Framework components

This phase will discuss about automation components which are put together to create an automation framework. At high level it" s completely black box and it" s important to decompose to understand its architecture and other design concepts. It consists of:

- ❖ Configuration manager
- ❖ Data Driven functionality
- ❖ Automation approach and coding standards
- ❖ Class, objects and methods for CLI apps - is based on SOA (service oriented architecture)
- ❖ Utilities for various tasks conceived – is again based on SOA
- ❖ 3rd party Tools identified
- ❖ Functional architecture
- ❖ Recovery system
- ❖ Result Managements
- ❖ Error handling
- ❖ Object repository specific to apps
- ❖ Key constants

The more granular level details of framework is presented the figure 1

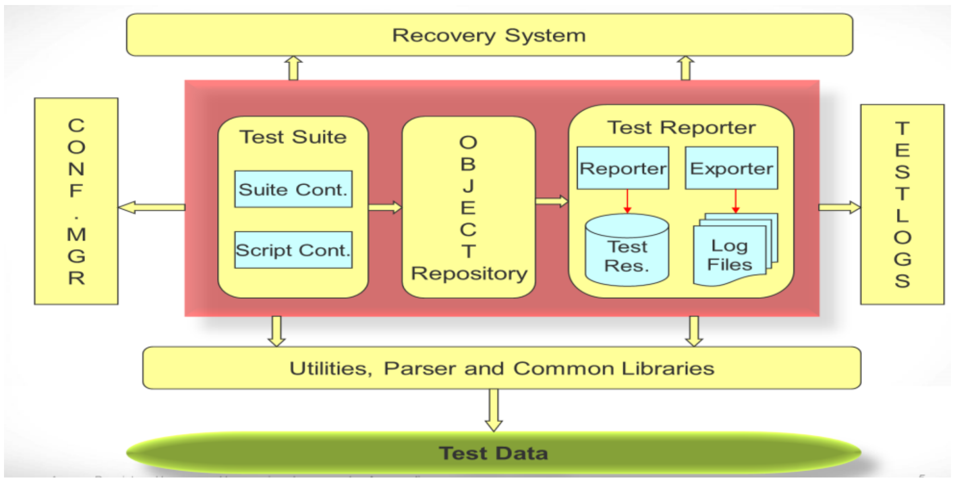

Figure 1: Automation Framework architecture

IMPLEMENTATION:

This phase will outline how to approach implementation and phases involved with. Since High level test plan and test script are designed, it is now just to follow test development process to implement those designs. As we know that the test script development process is almost similar to Development cycles. So, introducing the test script development phases might help automation team to speed up the implementations.

- Design test plan and test script
- Develop functions/methods and test case
- Do unit test of functions/methods and test case from test script
- Debug and fix issues if any found
- Integrate test cases with test plan
- Do integration testing by selecting and executing set of test cases in unattended manner
- Debug and fix if any issues found
- Add the sub-plans into regression master plan
- Do system testing by executing regression suite overnight
- Debug and fix if any issues found

1. **Development**

It is now matter of converting manual steps into automated scripts by developing methods and functions identified to execute the steps and verify the expected results automatically.

2. **Test Execution and Management**

After creating a robust regression suite, it's time to execute as many times as possible and manage the automated suite with the help of Test management tool or some other means.

3. **Automation Tasks**

Executing an automated regression suite is typically a time consuming and interactive in nature. There are various factors involved though. Like the size of the suite, robustness of the suite, number of AUT changes, number of h/w resources availability and the resource expertise level with the regression suite. Hence, identifying the tasks involved with an end-to-end automated regression suite run may be significant for sizing activity.

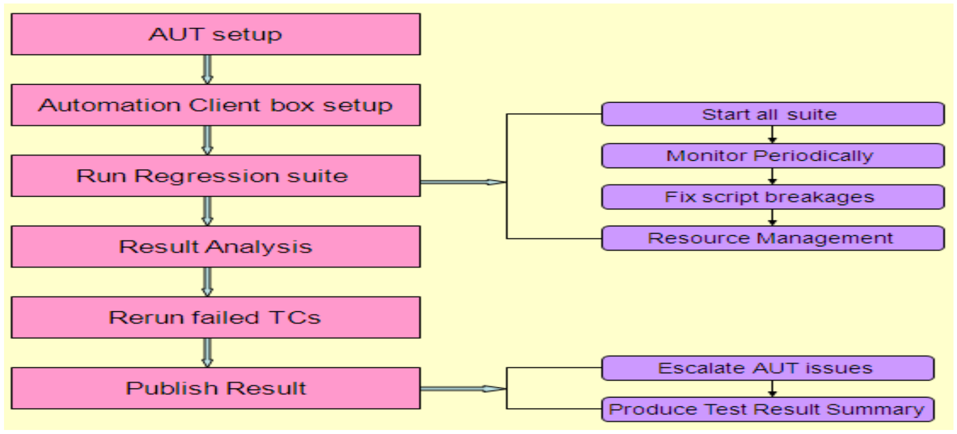

Figure 2: Automation tasks

The figure 2 clearly depicts the tasks involved with its flows. Here few tasks are one time activities and few are repetitive in nature.

4. **Execution**

After preparing both server and Automation client boxes, now simply open the regression master plan and select the particular functionality to be executed. The test suite provides options to run either from command line or IntelliJ Idea IDE. It also provides options to create a schedule job for repetitive runs. The daily sanity deployment is one of them which must be scheduled as its continuous one. Please refer the figure 2 Tasks diagram for tasks involved during execution. It's estimated that a complete UI regression run might take 3 to 5 man days with 3 servers and 8-10 client boxes setups

TEST RESULTS AND OBSERVATION

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner.

Once our framework is ready, the suite is used to test the test cases mentioned in the excel sheet. The test cases and its steps are mentioned in the excel sheet. Step by step execution of test cases is achieved. Verification of the test case execution is performed using testing assert statements and a report is generated.

As regression suite is broken and started on multiple client boxes, the completion time of each suite run is short and quick. So the test results may be available for analysis the very next day. As mentioned earlier without human interactions the results make no sense/difference. So, it" s one of the important phases of whole automation process. In different words creating, maintaining and running scripts are only one part of the overall automated testing process. However, analyzing the result and isolating script and application failures usually takes much longer than the actual test itself. So it is so critical for successful runs. If the failures are isolated as script issues, it must be fixed if it s simple and not time consuming ones. Select all failed scripts and rerun them after the quick fix. Finally merge the successful scripts with master regression result files. Its one of the ways to reduce number of failures for analysis.

REFERENCES

1. https://en.wikipedia.org/wiki/Microsoft_UI_Automation
2. [Website]http://www.oracle.com/technetwork/articles/entarch/shrivastava-automated-frameworks-1692936.html
3. [Website] Selenium [OB/HB methodology]. http://http://www.seleniumhq.org
4. HOFFMAN, D.: Test Automation Architectures: Planning for Test Automation. 1999,[online],[cit.2013-03-at:http://www.softwarequalitymethods.com/Papers/autoarch.PDF.