International Journal of Advance Research in Science and Engineering
Vol. No.5, Issue No. 06 , June 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

AUTOMATION FRAMEWORK FOR DATABASE TESTING

Praveen Kumar[1], Dr. Kavita[2]

[1]Research Scholar, [2]Associate Professor, Department Of Computer Science,
Jayoti Vidyapeeth Women's Universiy,Jaipur, India

ABSTRACT

In today's environment, test automation becomes an increasingly critical and strategic necessity. Assuming the level of testing in the past was, how do we possibly keep up with this new explosive pace of web-enabled deployment while keeping satisfactory test coverage and reducing risk? The answer is either more people for manual testing, or an advance level of test automation. Reduction in project cycle times correlates to a reduction of time for test.

In every organization, database applications play an important role, but little has been done on testing of database applications. They have become complex and are subject to constant change. Testing of database application is of utmost importance to avoid any errors encountered in the application, since a minor fault in database application can result in unrecoverable data loss. Several tools and frameworks for performing testing of database applications has been introduced to populate the test database and create test cases which checks the validity of application

In this paper we present different automated framework for database application testing. Finally we present strategies for performing efficient regression tests by reducing the re-tests that may occur while testing database applications. We will talk about different frameworks used across organizations while automating database test cases.

Keywords: Data Warehouse Testing, Techniques, Automation Framework, Database Testing.

I. INTRODUCTION

What is Framework?

A framework is a combination of set protocols, standards, rules and guidelines that can be integrated or followed as a whole so as to have the benefits provided by the Framework.

Test Automation Framework

A "Test Automation Framework" provides an execution environment for the automation scripts. The framework provides user with various benefits that helps them to execute, develop and report the automation test scripts efficiently. It is like a system that has built specifically to automate our test cases.

In a very simple language, we can say that a framework is a mixture of different guidelines, concepts, coding standards, processes, modularity, reporting mechanism, practices, test data injections etc. to pillar automation testing. So, user can follow these guidelines while automating application to take advantages of various results.

The advantages can be in different forms like ease of scripting, modularity, understandability, scalability, process definition, maintenance, re-usability, cost etc. Thus, to be able to achieve these benefits, developers are directed to use one or more of the Test Automation Framework.

Moreover, the need of a standard Test Automation Framework arises when you have developers working on the several modules of the same application and when we do not want to have situations where each of the developer implements his approach towards automation.

Types of Test Automation Framework

Now as we have basic idea of what is an Automation Framework, we will check with the various types of Test Automation Frameworks those are available in the market. We would also try to shed lights over their pros and cons and usability recommendations.

There is a different range of Automation Frameworks available now days. These frameworks may differ from each other based on different key factors to do automation like ease of maintenance, reusability etc.

Most popularly used Test Automation Frameworks are:

- Module Based Testing Framework
- Data Driven Testing Framework
- Keyword Driven Testing Framework
- Hybrid Testing Framework
- Behavior Driven Development Framework

II. MODULAR FRAMEWORK

This is one of the most basic type of automation framework. In modular framework, a test scripts are written to match a functionality that correspond to modules of the application. These modules are used in a hierarchical fashion to build large test cases.

Modular framework assigns independent test scripts to specific software components, modules or functions. Removing script inter dependencies is crucial to framework maintainability and stability.

Each module script embeds both data and actions. Any change in the test data requires script changes or a new script. Where data changes are common, a data-driven testing framework would be preferable.

Advantages:

- Maintenance is easy because of modular division of scripts and also the scalability of the automated test Scripts are independent to write.

Disadvantages:

- The main problem with this frameworks is that the test script have data embedded in them, which will become problem when updating the code /script.
- Whenever a test step fails, it is difficult to find out by debugging where the test case failed.

Data-Driven Testing Framework:

In this framework, test data is separated from test scripts and results are returned against the test data. Finally if all the test data are pass, then only the test case is treated as "PASS". If any one of the test data is failed, the entire test case will be treated as "FAIL".

Where tests use varied sets of input and output test data, this approach is more easily managed and more efficient. Test scripts get all input/output data from an external database, which improves their re-usability. Test data are stored as key-value tuples in a uniform fashion. Data extraction within scripts is facilitated by a common library.

Data-driven testing is creation of test scripts where test data are read from data files or tables instead of using the same hard-coded values each time the test runs. This way all testers can test how the application reacts on various inputs effectively with different data.

Advantages:

- It is suitable for test cases with different test data combinations.
- It reduces the number of test scripts needed to implement all the test cases.
- The test data can be prepared before test implementation is ready.
- The most important feature of this framework is that it reduces the total number of scripts required to cover all the combinations of test scenarios. Thus lesser code is required to test a complete set of scenarios.
- Increases flexibility and maintainability

Disadvantages:

- It is not suitable for test cases having simple actions which doesn't include any test data.
- Need to write different scripts to understand different sets of data.
- The process is complex and requires an additional effort to come up with the test data sources.
- Requires proficiency in a programming language that will be used to develop test scripts.

Keyword Driven:

In this framework, keywords are written which are equal to a unit level functionality. It is an application independent framework which utilizes data tables methods and keywords to perform the actions.

The Keyword driven testing framework is an extension to Data driven Testing Framework; it not only segregates the test data from the scripts, it also keeps the code belonging to the test script into an external data file.

Keyword driven framework is also called table-driven framework. It separates test automation from test case design, it adds an extra layer of abstraction via keyword-value tuples in the data matrix. Different keywords are used to drive tests independently. This framework enables less technically skilled staff to create test scripts and also improves test script readability.

Advantages:

- This framework provides high re-usability; this can be achieved by re-using script across multiple test cases.
- Maintainability is easy which doesn't require any coding or Automation expertise.
- Debugging is very easy.

- Keyword driven framework not require the user to know scripting knowledge unlike Data Driven Testing.
- A single keyword can be used across multiple scripts.

Disadvantages:

- It is a complicated framework than the data driven framework.
- Test cases become complex and grow longer sometimes.
- Takes time to build and stabilize the framework.
- The user should be well versed with the Keyword creation mechanism to be able to leverage the benefits provided by the framework.
- The framework becomes complicated eventually as it grows and a number of new keywords are introduced.

Hybrid Framework:

This framework is the combination of data-driven and keyword driven testing frameworks.

The Hybrid Test Automation Framework incorporates features of Script modularity, Data Driven framework and Functional Decomposition.

For instance, hybrid approach can be the use of a common library attached with a data repository that combines both action keywords and test I/O data. Each tuple in the repository may contain a name, description, UI locator, action keyword and text string.

Naturally, a hybrid approach may be more complex to set up initially. However, it could provide the greatest flexibility if it is carefully evaluated and implemented.

Advantages:

The Hybrid framework is built with several reusable modules / function libraries that are developed with the following features in mind:

- Maintainability – hybrid framework extensively reduces maintenance effort
- Re-usability – test cases and library functions can be reused.
- Manageability - effective test design, traceability and execution.
- Accessibility – easy to develop, design, modify and debug test cases while executing
- Availability – allows to schedule automation execution
- Reliability – advanced error handling and scenario recovery
- Flexibility – framework independent of system under test
- Measurability – customizable reporting of test results ensure the quality output

Behavior Driven Development Framework:

In simple terms, it is a bridge between business language (User Stories) and automatic tests (TestNG, JUnit). BDD framework allows automation of different functional validations in easily understandable and readable format to Developers, Business Analysts, Testers, etc. Such frameworks do not necessarily require the user to be familiar with programming language. There are several tools available for BDD like cucumber, Jbehave etc.

A key drawback to this approach is that BDD scenarios are so vague that they are often different with technical reality. In practice, it is difficult to get participation from business stakeholders in creating or evaluating BDD scenarios.

III. CONCLUSION

The frameworks demonstrate above are the most popular frameworks used by the testing fraternity. For efficient and fast product development and delivery it is crucial to implement reusable and effective test automation. Properly planned and implemented automated testing can extensively lower project risk and cost. If automation has been well designed and implemented, it will not only reduce the risk of the current project, but can actually reduce the risk and cost of future projects. The ROI for test automation is easy to establish.

An automation framework can be created around model based testing that begins with automating the test design exercise and with the development of manual test procedures. It further uses automation frameworks to achieve modularity in automated test script design, and combine with test execution tools for automated test execution.

REFERENCES

[1] NASSCOM Research & Intelligence - Software Testing: Shifting from Functional to Business Assurance http://survey.nasscom.in/sites/default/files/researchreports/softcopy/ Software%20Testing%20Report_Secured.pdf

[2] MBT: A superior alternative to traditional software testing - HCL Blog by Ambica Jain on 11 Oct 2012 http://www.hcltech.com/blogs/engineering-and-rd-services/modelbased-testing-%E2%80%93-superior-alternative-traditionalsoftware-te

[3] Modeling Introduction by Hani Achkar http://testoptimal.com/ref/TO_Presentation/Modeling_101.pdf

[4] Model-Based Testing (MBT) – HCL Whitepaper by Naveen Jain published on 14 Jun 2013 http://www.hcltech.com/whitepapers/engineering-services/model-based-testing

[5] Selenium – Web Browser Automation. (2013). Retrieved from Seleniumhq.org: www.seleniumhq.org

[6] TestNG. (n.d.). Retrieved from TestNG: www.testng.org What is Software Architecture? (2013).

[7] Retrieved from Microsoft Developer Network: http://msdn.microsoft.com/en-us/library/ee658098.aspx

[8] 2010 paper by Ashutosh Jha “DEVELOPMENT OF TEST AUTOMATION FRAMEWORK FOR TESTING AVIONICS SYSTEMS” Digital Avionics Systems Conference (DASC), 2010 ieee.

[9] D.R. Kuhn, D.R. Wallace, and A. Gallo, “Software Fault Interactions and Implications for Software Testing,” IEEE Trans. Software Eng., vol. 30, no. 6, 2004, pp. 418–421.

[10] Glicker, S.; Hosch, F., "A design approach for a distributed test automation system," Applied Computing, 1990., Proceedings of the 1990 Symposium on , vol., no., pp.9,11, 5-6 Apr 1990, doi: 10.1109/SOAC.1990.82132.

[11] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.

[12] Caniupan, M.; Placencia, A., "Data Warehouse Fixer: Fixing Inconsistencies in Data Warehouses," Computer Science Society (SCCC), 2011 30th International Conference of the Chilean , vol., no., pp.28,32, 9-11 Nov. 2011 doi: 10.1109/SCCC.2011.5.

[13] Ramachandran, M., "Testing software components using boundary value analysis," Euromicro Conference, 2003. Proceedings. 29th, vol., no., pp.94,98, 1-6 Sept. 2003 doi: 10.1109/EURMIC.2003.1231572.