An Automated Testing Tools Activity for Software and Methodologies

P.Venkata Soma Sekhar#1, K.Venkateswarlu *2
#1 M. Tech Student, Dept. of CSE, GIT, GITAM UNIVERSITY, Visakhapatnam
*2 Assistant Professor, Dept. of CSE, GIT, GITAM UNIVERSITY, Visakhapatnam
1987.somu@gmail.com , venkat.cst@gmail.com

Abstract : Software producing companies want to increase their quality and efficiency. They often look at automated test tools as a part of a solution. Not many company's use test tools, the time it takes to evaluate which test tool that suites the company best costs much in both money and time. An automated test takes a lot of time to implement and introduce in the company To get the whole picture when to automate test we have looked at different test methodologies. Step-by Step Method, Product Life Cycle Method. We have also investigated some general information about automated test like cost and when to automate. ATLM (Automated Test Life- Cycle Methodology) is a methodology that is used in order to take the correct decisions such as when to automate or not. (ATLM) is a structured methodology aiming toward ensuring a successful implementation. We also have some methods of automated tools. The Record/Playback Method, The functional Decomposition Method and The Key- Word Driven Method are a data-driven automated testing methodology. That allows developing automated test scripts that are more "generic". It requires only that the input and the expected results have been updated.


I. Introduction

It is common practice for hardware to be designed for testability; however, the testability of software is rarely considered. When software testability is addressed, the resultant design often does not readily support full coverage automated testing. Since software products must be tested to verify requirements are met, it only makes sense to consider software testability from day one of a project. Once the decision has been made to embrace the concept of designing testable software, there are best practices that enable a lean software development process. This paper will discuss 1) The methodologies [4] implemented during software testing 2) General facts about automated tests; 3) Evaluation of Automated Tool methods 4) Lean pros and cons.

II. Different Test methodologies

Test designers who have worked with tests a long time has created their own test methodologies [4]. This is a presentation of some methods that can be a help to create own method or just get some ideas or tips. The reason we chose these particular methodologies is because they are easy to follow and well structured.

The Step – by – Step Method

The method can be used for some parts of the product that is developed. It works for both black box and glass box testing. It can be used by an individual or by a small group. It creates lists and tables that are easy for the designer to handle.

This basic test design method consists of the following six steps:

- List test requirements based on the specifications
- Add test requirements for a range of inputs
- List a test type for each test requirement
- Review test types and fill in the holes
- Write a test case for each test requirement
- Group test cases into test scripts

Product Life Cycle Method

The implementation of the PLC[5] varies widely from company to company. You can use this as a guide for future reference to assist you in your automation efforts. Your implementation will vary from the ideal PLC that is discussed here, but your software's success may depend on how well you've implemented its PLC. If your PLC [5] is to include automated testing you should pay attention to which automated tasks are performed during each phase

- Design Phase
- Code Complete Phase
- Alpha Phase
- Beta Phase
- Zero Defect Build Phase
- Green Master Phase

III. General facts about Automated Tests

Automated Test Lifecycle Methodology (ATLM) This is a structured methodology geared toward ensuring successful implementation of automated testing.

International Journal of Internet Computing (IJIC)2011
When should a Test be automated?

To decide whether a test should be automated or not you need to look at the cost and design of the test. Both automation and manual testing are plausible. That is not always the case. For example, load testing often requires the creation of heavy user workloads. Even if it were possible to arrange for 200 testers to use the product simultaneously, it is surely not cost-effective. Load tests need to be automated.

You first design the test and then decide whether it should be automated. It is common for the needs of automation to influence the design. This sometimes means that tests are weakened to make them automatable.

To introduce Automatic Tools

1. Determine your budget for automated tools.
2. List the various testing functions of your group or department. Briefly outline how these are undertaken currently.

The list can look something like this:

- Test Planning
- Defect Tracking
- Performance Testing
- Functional Testing
- Regression Testing

3. Against each category compile a list of functions that could be automated.
4. Determine what it would cost to supply your automated testing needs based on the above assessment.
5. Implement them.

Instigate a planned review process over the next two to three years to ensure that the adoption and uptake of tool is appropriately maintained.

What is required to Successfully Implement Automated Testing?

Automated testing is automating the manual testing process currently. This requires that a formalized "manual testing process" exists in the company and includes such a process:

- Detailed test cases, including predictable "expected results", which have been
developed from *Functional Specifications* and *Design documentation*.

- An adequate Test Environment department, including a test database that is restorable to a known constant, such that the test cases are able to be repeated each time there are modifications made to the application.

### The Costs to Automated Test

Automating a test and running it once will cost more than simply running it manually one time. How much more? An automated test has a finite lifetime, during which it must recoup that additional cost. Is this test likely to die sooner or later? What events are likely to end it? The essential question is: During its lifetime, how likely is this test to find additional bugs (beyond whatever bugs it found the first time it run)? How does this uncertain benefit balance against the cost of automation?

### Cost Effective Automated Testing

Automated testing is expensive and it does not replace the need for manual testing or "down-size" the testing department. Automated testing is an addition to the testing process. It can take between 3 to 10 times as Long (or longer) to develop, verify, and document an automated test case than to create and execute a manual test case. This is especially true if the "record/playback" feature (contained in most test tools) is elected as the primary automated testing methodology.

### Automated Tests Survival

Automated tests produce their value after the code changes. Except for rare types of the tests, Re-running a test before any code changes is a waste of time. It will find exactly the same bugs as before. (The exceptions, such as timing and stress tests, can be analyzed in roughly same way.) But a test will not last forever. At some point, the product will change in a way that break’s the test ex, changing database etc. The test will have to either be repaired or discarded. To a reasonable approximation, repairing a test costs as much as throwing it away.

### Losing with Automation

Creating an automated test is usually more time-consuming (expensive) than running it manually one time. The automated tests are not free from bugs, which the tester must have in mind. The cost varies, depending on the product and the automation style.

### IV. Preparation before installing an Automated Testing Tools

An adequate test environment must exist that accurately replicates the production environment. This can be a small-scale replica, but it must consist of the same types of hardware, programs and data. The test environment's database must be able to be *restored* to a known baseline otherwise tests performed against this database will not be able to be repeated, as the data have been altered. Part of the test environment includes hardware. The automated scripts must have dedicated PC's on which to run. If one is developing scripts, then these scripts themselves must be tested to ensure that they work properly. It takes time to run these scripts, especially after a number of them have been developed.

Detailed test cases that can be converted to an automated format must exist. If they do not then they will need to be developed, adding to the time required. The test-tool is not a thinking entity. You must tell it exactly what to do, how to do it, and when to do it. Data to be entered and verified must be specific data. The person or persons who are going to be developing and maintaining the automated scripts must be hired and trained. Normally, test tool vendors provide training courses.

### V. Evaluation of Automated Tool methods

There are three different interfaces available for testing. Some products may have all three, but many will have only one or two. **API** (application programming interfaces) and **CLI** (command line interface) are easier to automate than **GUI** (graphical user interfaces) because the GUI test automation requires manual scripting. In this essay we have only looked at GUI.

- **The Record/Playback Method*[2]*
- **The “Functional Decomposition” Method*[3]*
- **The Key-Word Driven or Test Plan Driven Method*. [4]

### VI. Automated Tools

Here we describe the automatic tools we have investigated. There are many companies who sell test tools on the market. We have chosen some of the biggest and the companies with the best reputation, which are offering test tools that are interesting for future testing environment.
An Automated Testing Tools Activity for Software and Methodologies

Rational Robot

One automated test tool system that we will examine more carefully is the one provided by Rational Robot [9]. This offers a different set of tools to suite your company concerning testing and quality. No matter what you buy from Rational [9], you have two options: one is to put the system on a PC or on a server. It uses VB scripting language, which must be learned by those writing the test scripts for which some degree of previous programming experience is required. This provides the building blocks for test automation.


Mercury Interactive WinRunner and Quick Test Professional (QTP)

Mercury Interactive's WinRunner is an enterprise functional testing tool that verifies if applications work as expected by capturing and replaying user interactions automatically. Quick Test Professional [14] is a regression testing tool. WinRunner's development environment provides the foundation for developing effective test automation. It uses scripting language (TSL), which must be learned by those writing the test scripts for which some degree of previous programming experience is required. This provides the building blocks for test automation.

WinRunner identifies defects and ensures that business processes, which span across multiple applications and databases, work flawlessly the first time and remain reliable throughout the lifecycle. WinRunner and QTP [14] support an extensive range of Java environments, including industry leading virtual machines such as: Internet Explorer, Netscape, Sun's Java Plug-in, Applet Viewer, Oracle Initiator, JDK/JRE, Microsoft's view, Java toolkits including Java Foundation Classes, Symantec Visual Cafe, KL Group Class, Oracle Developer, Sun's AWT and among others.

Silk Test

SilkTest [15] is a regression-testing product for e-business applications. It can be used to test a Web, Java or traditional client/server application. SilkTest tests entire applications end-to-end from front-end clients to back-end web, database and application servers. It can drive scripts from a central point of control, even when they are operating on entirely different platforms; it gives an accurate picture of how well the system components are performing together. SilkTest [15] recognizes the multiple technologies that are found in e-business applications including HTML, JavaScript, ActiveX, Java, Windows 98 controls, Visual Basic and C++. SilkTest allows testing Java applets or components across multiple environments from a single script. Tests can be integrated with the Java Developers Kit (JDK) and SWING user interface components.

VII. Result and Conclusion

After our study in this subject, I feel that I have only touched the surface of this subject. There is not much theory concerning automated test tools. The information we found is usually, and then it is difficult to hear about failures and disadvantages of automated tests. Our goal was to compare different test tools and check it for future enhancement and the comparison is between automation tools. It was also difficult to get information about automated tools from these companies because you have to buy the product to fully receive all information about it.

VIII. References
