1234 Main Street

Fredericton, NB

ABC 123

August 31, 2010

Patricia Meng

Co-op Coordinator

Co-op Program, Faculty of Computer Science

University of New Brunswick

P.O. Box 4400

Fredericton, NB

E3B 5A3

Dear Patricia Meng:

During the summer term, I was employed with XXXXXX for my third work term. I was under the supervision of XXXX in the Quality Assurance department. My duties and responsibilities involved creating, executing and modifying test cases.

My work term report is titled “An evaluation of Automated Testing Tools for Web Applications that Support Flash”. The report relates to Quality Assurance, where I worked.

Although I was not involved in any automation testing, I asked my supervisor if there was any topic in this area that I could research. She provided me with the research topic, however all work done in the report is my own work.

Sincerely,

XXXXX

Encl: Work Term Report

An Evaluation of Automated Testing Tools for Web Applications that Support Flash

XXXXX
3333333

August 31, 2010

Executive Summary

This report looks at several different areas that need to be considered when evaluating Automation Testing tools, and applies them to five randomly selected tools. An overview is given of automation testing - touching on what it is, how it is different from manual testing, and how it applies to Flash technology.

The areas that are considered when evaluating the tool are: Flash testing capability, Browser support, Operating system support, User friendliness, Scripting support, Results, Technical Support, Trial Version Availability and Cost. The reasons for looking into these requirements are examined, as well as what should be expected when evaluating the chosen testing tools.

The five randomly chosen automated testing tools that are evaluated are: iMacros, TestComplete, HP QuickTest Professional, Selenium and Watir. Each is analyzed based on the 9 requirements.

The recommendations reached are:

Both Watir and HP QuickTest Professional should not be used. Watir is not yet developed enough, and HP QuickTest Professional is too developed - too many features, with too high a cost.

Selenium, iMacros and TestComplete are recommended for use, under certain conditions. Selenium should be used when automation is desperately needed, but no budget exists to purchase a commercial tool. iMacros or TestComplete are recommended tools for use in any organization that has the budget to allow for them.

Table of Contents

1 Introduction

There are always new ways of developing software being introduced into the world. A popular new trend for web applications is to use Flash technology. [1] As a result, automated testing tools need to adapt to this new design. Here, we explore why Flash cannot be tested using normal automated testing tools, and compare and contrast several applications which do support this feature. Based on the general needs for good automation software, several considerations that need to be taken into account are; [2]

1. Flash testing capability
2. Browser Support
3. Operating System support
4. User friendliness
5. Scripting support
6. Results
7. Technical Support
8. Trial Version Availability
9. Cost

Using these requirements as a guide, an evaluation will be performed of several automated testing tools available which support flash testing. Five randomly selected testing applications will be analyzed;

1. iMacros
2. TestComplete
3. HP QuickTest Professional
4. Selenium
5. Watir

2 Overview of Automated Testing

As the name suggests, automated testing involves testing done automatically, rather than manually by a person. In order to automate a test case, you must enter the instructions and expected results into the testing program. Depending on the program, this could involve detailed programming or be as simple as performing the instructions yourself while the program records the actions. Once the test case has been entered into the system, you can execute it to get a result. There are different ways for the testing program to execute the tests. The two most common methods are injecting JavaScript code into the browser and using macros.

Injecting JavaScript into the browser involves the testing program generating code using the instructions you gave it, and sending requests to the browser. The response will generally be new html code to display in the browser; letting the program check and return a result (Pass or Fail).

When using a testing program that employs macros, you are telling it to simulate mouse clicks and keystrokes. This better simulates user-actions but does not take into account the possibility of an object appearing somewhere different from where you originally clicked it on a webpage.[2]

Either way, the result is given after the test has been run, and can be evaluated.

2.1 Differences between manual and automated testing

Some people mistakenly believe that all testing should be automated. Automation testing is good for repeated actions and regression testing, but should not replace manual testing. Humans will always be able to detect patterns and anomalies, whereas in automation the program only looks for bugs and errors exactly where it is told to. [3][4]

Automated testing can complete a given set of test cases much faster than manually running them, but requires a lot of maintenance. Every time a new release is done, the scripts must be checked and updated to reflect any expected code changes that occurred. [5]

Ultimately automated testing should be considered an asset to manual testing; not a replacement for it.

2.2 Automated testing of Flash

Since Flash is a fairly new technology, it is taking the vendors of testing tools time to adapt their products to allow for the testing of it.

Flash is typically embedded in a web page as an object file (See Appendix B for an example). When a person clicks on an element of the flash object, the html source does not change - so normal methods of determining the results of an action cannot be used.[1] Automation software must turn to other means to test Flash.

One option is to instead use the API provided by Macromedia Flash to access the elements of the Flash object.[6] From these elements, it can be determined what has changed and what has not.

Another way to test Flash objects in a webpage is to record an image of what they should look like (expected result). The testing tool will then execute the test and compare the given image to the one on the webpage. [7]

It is the development involved with adapting testing tools to new methods that slows down the vendors from supporting the testing of Flash.

2.3 Requirements

Each of these requirements needs to be considered when choosing a tool to use for automation testing. What will be considered for each is explained below.

2.3.1 Flash testing capability

The testing tool must support the ability to test Flash. As mentioned earlier, this is fairly new technology and many are starting to use it in their web applications. While it can easily be tested manually, the idea here is to evaluate automation tools that can test web applications using Flash.

Many testing tools that support Flash have a plug-in or new feature that allows better integration and testing of Flash objects.[7][8] These will be investigated and evaluated.

2.3.1 Browser support

Browser support is a crucial concern. The testing tool must work on any browsers that are supported by the web application. Typically this will include Internet Explorer and Mozilla Firefox. Sometimes less used alternatives are included, such as Google Chrome, Opera, Netscape Navigator and Safari. The browser's version can also be important – it will take time for vendors to support the latest versions, especially if there are major changes in the new release.

We will focus on the number of browsers that are supported - the more the better.

2.3.3 Operating System support

Another key requirement is that the testing tool will work on the same operating systems as the web application. The support of new operating systems should be taken into account as well. This means that if your web application does not work on a different operating system but will be expected to someday in the future, then the testing tool must support that operating system as well.

The testing tools will be evaluated based on the number of operating systems supported, as well as the different versions (for example, XP, Vista).

2.3.4 User friendliness

When evaluating these programs, the user friendliness will be considered. Each tool will be analyzed on user-friendliness via trial versions, if available, as well as user feedback. This includes ease of navigating the program, creating test cases and running the tests.

Ease of creating test cases will involve whether or not the process allows for quick and easy 'click and record' or if more effort is required. In addition, the difficulty required to modify and run the test cases will also be considered.

2.3.5 Scripting support

Many testing tools support scripting capabilities. This gives the tool a powerful interface that allows the user to develop more detailed test cases than is typically done with the built-in features of the tool. This also allows for features that are not supported by the testing tool to be developed and integrated into the testing process.

A key thing to look out for here is a tool that provides its own scripting language. This will slow down the testing process since the test automation engineer will need to learn the language before coding in it.

The testing tools will be evaluated based on how many languages are available as well as if they are standard scripting languages. [2]

2.3.6 Results

Something to be considered is the results aspect of testing programs. The ability to log and report the results in a clear, concise manner is important. Whether it is to create a report or examine the results of the last test run, it is essential to be able to determine what passed and what failed.

Further functionality, such as the ability to determine where a test case failed, or the ability to export the results for use in another program, will also be examined. [2]

2.3.7 Technical support

The availability of technical support is an important consideration. A commonality among almost all vendors is an online forum to discuss issues and solutions that arise. The evaluation in this particular area will focus on that, as well as other options that are available. This includes many alternatives such as;

1. Telephone support
2. E-mail
3. Chat
4. FAQ
5. Bug-Tracking system
6. How-to (Articles, Videos, Training)

2.3.8 Trial Version availability

The availability of a trial version is an important aspect of choosing an automation testing tool. If you cannot test the tool on your product(s) before purchasing it, there is no guarantee that it will work. It also good to ensure that the product is easy to use, with understandable features and processes.

Evaluation of this area will focus on the period of time allotted for trial evaluations. Consideration will also be placed on what version is given for evaluations as well as what features are limited for the trial.

2.3.9 Cost

The cost for a testing tool can be a big factor in choosing. Some testing tools are designed to test any application, not just ones on the web. This means that if there is a testing tool with extra functionality that is not needed, we need to decide if the extra cost is worth any added benefit. Some testing programs are open-source, so no payment is necessary.

Evaluation for this will be based on open-source versus commercial, focusing on what really needs to be paid for.

3 Applications

3.1 iMacros

iMacros is a web automation testing tool developed by iOpus Inc.[7] It involves the use of macros in the recording and execution of test cases.

iMacros uses a feature called Image Recognition Plug-in to test Flash components. This is an add-on to the testing tool that allows you to search for a specific image pattern on a webpage. This is the key technology used to test non-html technologies (Flash, Java, ActiveX, and more). Whenever the tool executes a test case involving Flash, it searches the webpage for a match to the image, and is able to determine if the result is correct. This way of approaching Flash testing is beneficial, since dynamic Flash components that do not always show in the same place can still be tested using automation.[7]

Unfortunately, iMacros only supports Internet Explorer and Mozilla Firefox on the Windows operating system(XP, Vista, 7, Server 2000/2003/2008). It does, however, also support Firefox on MAC OSX and Linux.[7] While it would be nice to have support for more browsers, these two are the most widely used.

While testing the trial version of the tool, it was found that the product was not overly user-friendly. Some tutorials and how-to videos were available, but the layout of the application was found to be difficult to use. The results were often cryptic and hard to understand - and the log file was no different. Part of this was also due to the lack of a proper reporting system. The modification of a test case required knowledge of a scripting language.

However, the program supports all Windows scripting languages.[7] This is a huge advantage over many other testing tools, which only typically support a few or just their own scripting language. This includes VBScript, Java, .Net, Perl, C# and more.

A trial version of the software is available for 30 days, giving you full access to all features of the program. A Licence for this version can be purchased at a cost of US $499, which is a good price for an automated testing tool. With this, you get several technical support options including: Access to the forums, email support and 'Silver Level Support Plan' which guarantees a response within two days.[7]

This product is good for developers and test automation engineers who don't mind get their hands dirty with coding. For testers who do not know programming well, this will be a difficult program to use.

3.2 TestComplete

TestComplete is an automation test tool developed by AutomatedQA Corp.[8] This product supports the testing of many types of programs, not just web applications.

TestComplete does support the testing of Flash components in web pages. Through the built-in flash support for actual Flash applications, this testing tool has everything it needs to successfully automate testing web Flash components. Unlike some automation tools, TestComplete does not require any changes to the code of the Flash component in order to test it.[8]

This tool, like iMacros, supports Internet Explorer and Firefox, and in addition supports Netscape Navigator. Unfortunately, only the Windows operating system is supported (XP, Vista, 7, Server 2000/2003/2008).[8] This puts the tool at a disadvantage, since web applications should be tested on multiple operating systems.

The product was found to have an intuitive and user-friendly interface during testing of the trial version. Guides, tutorials and demonstration videos were easily accessible. Creating a test case was as simple as clicking 'record', performing the actions and stating the expected result, then clicking stop. Modification of a test case did not require knowledge of a scripting language - everything could be done through a visual representation of the code. After running a test case, the results were easy to understand and interpret. Overall, the program had a very simplistic look, but contained many features to use.

TestComplete supports five common scripting languages: VBScript, JScript, DelphiScript, C++Script, and C#Script.[8] Although this collection does not include all languages, it is better than having just one.

A 30-day trial version with limited features is available for download. While there are several limitations, all features are still there. They do not deter from the ability to test the tool. Purchasing a full licence for this version costs US $1999, putting the tool in the medium price range for automation test tools. Technical support provided with this version includes: comprehensive online FAQ, access to Forums, Screencasts and videos, Articles, Email, Live Chat and a troubleshooting application online. For US $399, TestExecute can be purchased - which is a program capable of executing the tests created by TestComplete.[8]

Although a bit expensive, and supporting more than just web application testing, Test Complete is rich with features. The web testing section of the tool works well and is easy to use. This product would be good for any organization with enough money in the budget to purchase it.

3.3 HP QuickTest Professional

HP QuickTest Professional is an automated software testing program developed by Hewlett-Packard Development Company, L.P. It supports testing for a huge collection of applications.

Similar to TestComplete, QuickTest Professional is able to test Flash without needing to have specialized code. By using the technology already implemented for full-scale Flash applications, the testing for Flash components on a webpage can be automated. [9]

Unfortunately, HP QuickTest Professional only supports the Internet Explorer and Mozilla Firefox browsers. As well, its operating system support is limited to only Windows.[9][10]

This is regrettable, since the user interface was extraordinarily feature-rich and easy to use. With the assistance of a help screen, creating a test case was simple. Running the test case and getting a result was even easier. The reporting section of the tool is very detailed and metrics are provided showing how many test cases were run, how many passed and how many did not. Editing and removing a test case was straightforward to do. Similar to TestComplete, knowledge of a scripting language was not required to modify the test case.

HP QuickTest Professional supports scripting for several languages including DelphiScript, Java, and .Net.[10] While this collection is not as extensive as iMacros' support of all Windows scripting languages, there is enough variety for most people to know at least one of them.

Version 10 is available to download online as a 14-day trial. This evaluation version gives you full access to all the features. Although not listed, a licence is estimated to be worth upwards of US $5000. This price range places the product in the high-end and very expensive section of automated software testing tools. Technical support offered with this product includes: Forums, FAQ, User Guides, Email support, Case management(log/track cases), Enhancement requests, access to knowledge base and telephone support. [9]

This testing tool is not for small-medium sized businesses. If your QA department does not have much of a budget, and especially if you only need an automation tool for web applications, this is not for you. The web application testing functionality is only a small part of this program. It is not worth all the extra money for the other features that will not be used.

3.4 Selenium

Selenium is an automated web application testing system. Originally developed at ThoughtWorks in Chicago, the testing tool is now open-source.[11]

Flash testing for Selenium has been developed by another person from ThoughtWorks and open-sourced as an add-on to Selenium, called FlashSelenium. FlashSelenium is designed to adapt Selenium to the API of Macromedia Flash. The only downside is that when developing the Flash components, specialized code must be added. Methods must be created in the Flash code to return values so as to determine if those values are what they should be. These methods must then be externalized using 'Flash ExternalInterface' in order to access them through JavaScript.[11] This can involve a lot of development time, but considering that it is a free test tool, it is not so big of a deal.

Selenium supports a wide range of browsers and operating systems. It covers Internet Explorer, Mozilla Firefox, Safari, and Opera with partial support for other browsers. For operating systems, Selenium works on Windows, OS X(Mac), Linux and Solaris. [11]This is quite an advantage over most other testing tools - which typically only support Windows, and even then only Internet Explorer and Firefox.

In order to create test cases for Selenium, you must have Firefox, and have downloaded the Selenium IDE plug-in. This is not really a big deal, since Firefox is considered a major browser and most people have it. However, having the program run inside the browser is a bit disconcerting. Eventually, using the IDE begins to have an intuitive feel to it and the interface is simplistic. Even though using Selenium to test Flash requires programming, it is a free open-source tool and offers many high-quality features.

Selenium's API allows for programming to be done in a multitude of languages: Java, PHP, .Net, Ruby, Python, C# and Perl.[11] Even though these are not your typical scripting languages, they are well known and easy to get used to.

Selenium does not offer any advanced reporting features, but since the system is open source you can create your own. Exporting the results into xml and importing them into another program to evaluate the results is another option. Obviously, since it is a free open-source project there are no costs associated with it. Free technical support is available only in the form of forums and bug-tracking systems.[11]

Selenium, although requiring extra programming for use with Flash and for reporting, is a free open-source project. Not only is there no cost involved, but the program can be adapted to your organizations own needs. This is a good program for small businesses or QA departments with a very small budget.

3.5 Watir

Watir is another open-source project for automated web application testing. Using a family of Ruby libraries, it can simulate the clicking links, filling in forms and other tasks.

Flash testing is supported for Watir through an add-on called FlashWatir. Similarly to FlashSelenium for Selenium, FlashWatir requires the Flash component on the webpage to have the methods that need to be accessed externalized using 'Flash ExternalInterface'. At the moment, FlashWatir only supports Firefox.[12] This gives Watir quite a disadvantage, since Internet Explorer is the most widely-used browser.

Watir itself supports several browsers and operating systems. Among these are: Internet Explorer, Firefox, Safari and Google Chrome. It can be run on Windows, Linux and Mac so long as Ruby is also installed. Instead of supporting many scripting languages or creating its own, this product uses only Ruby as its scripting language.[12]

Using Watir requires a great deal of programming knowledge and patience. The test cases must be written using the Ruby script. There is no support for creating test cases using the record method. Due to the lack of instructions and tutorials, this tool is difficult to use.

Just like Selenium, Watir is open-source and thus does not cost anything. The only technical support available is the forum and bug-tracking system.[12]

Watir is still in the developing stages - both for a user interface and being able to support testing Flash properly. Due to this, Watir should not be used. If Flash is only tested on Firefox browsers, there are many possibilities for security vulnerabilities and bugs on Internet Explorer.

4 Conclusion

Automated testing tools for web applications using Flash are starting to become more available. The well known products on the market such as HP QuickTest Professional and TestComplete already have full support for the technology. Based on the considerations I used to evaluate the products (Flash testing capability, Browser support, Operating system support, User friendliness, Scripting support, Results, Technical Support, Trial Version availability and Cost), I have made the following recommendations.

Watir should not be used unless you are willing to put in the long development required to get Flash component working for Internet Explorer, and the extra coding needed for testing Flash. The lack of user interface also strongly suggests that this project needs more time to be developed.

HP QuickTest Professional should not be purchased and used for only automating web application testing, unless you plan to test other applications as well. There is no need to spend all the extra money if the other features will not be used.

If your organization is a small-sized business, with little to nothing for a QA budget - I recommend going with Selenium. Even with the extra coding needed to test Flash components, the availability of many features and no cost make this the best fit.

If your organization is medium-sized to large-sized, with a reasonable QA budget, I recommend purchasing and using either iMacros or TestComplete. The low cost and mass of features available with both products make them promising solutions.

5 Appendix A - Definitions and Abbreviations

API - Application Programmable Interface

HTML - Hyper text markup language (web code)

IDE - Integrated Development Environment

Macros - A series of keyboard and mouse actions recorded to a script. Used to automate repetitive actions or for automated testing.

Test case - A test with step by step instructions and an expected result.

Vendorscript - Scripting languages developed by automated test tool companies to support their product.

6 Appendix B – Embedding Flash Example

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Clicking Colors</title>
</head>
<body>
<object classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000"
codebase="http://fpdownload.macromedia.com/pub/shockwave/cabs/
flash/swflash.cab#version=8,0,0,0" width="100" height="100" align="middle">
    <param name="allowScriptAccess" value="always" />
    <param name="movie" value="Talker.swf" />
    <param name="quality" value="high" />
    <param name="bgcolor" value="#ffffff" />

    <embed src="ColoredSquare.swf" quality="high" bgcolor="#ffffff" width="500"
height="500" name="coloredSquare" id="coloredSquare" align="middle" allowscriptaccess="*"
type="application/x-shockwave-flash"
pluginspage="http://www.macromedia.com/go/getflashplayer" />

</object>
</body>
</html>
```

From: http://code.google.com/p/flash-selenium/source/browse/trunk/flash/changingcolors/colors.html

7 Bibliography

[1] "The Trend Toward Flash as a Tool for Web Application Development." *WritersUA - Training and Information for User Assistance Professionals*. WinWriters, Inc., 17 June 2009. Web. 6 Jan. 2010. <http://www.writersua.com/flash.htm>

[2] "Selecting Automated Testing Tools." *Tools for Software Testing and Quality Assurance*. AutomatedQA, 23 Nov. 2009. Web. 3 Jan. 2010. <http://www.automatedqa.com/techpapers/testcomplete/selecting-testing-tools-requirements/>

[3] Bach, James. *Test Automation Snake Oil*. Rep. 13 June 1999. Web. 3 Jan. 2010. <http://www.satisfice.com/articles/test_automation_snake_oil.pdf>

[4] Mitchell, Jamie L. "Testability Engineering." BenchmarkQA, 10 Nov. 2007. Web. 6 Jan. 2010. <http://www.benchmarkqa.com/pdf/papers_testability_engineering.pdf>

[5] "Software Testing - Test Automation." *Software Testing Geek*. TestingGeek. Web. 4 Jan. 2010. <http://www.testinggeek.com/index.php/testing-articles/136-software-test-automation>

[6] Caroli, Paulo, and Henrik Lindahl. "Writing and running functional tests for Flash with Selenium RC."*Adobe*. 23 June 2008. Web. 6 Jan. 2010. <http://www.adobe.com/devnet/flash/articles/flash_selenium.html>

[7] "IMacros | Web Testing." *IOpus | Web Browser Scripting, Web Testing, Web Scraping and Website Monitoring*. IOpus Inc. Web. 6 Jan. 2010. <http://www.iopus.com/imacros/web-testing.htm>

[8] *AutomatedQA - Tools for Software Testing and Quality Assurance*. AutomatedQA Corp. Web. 7 Jan. 2010. <http://www.automatedqa.com/products/testcomplete/>

[9] *HP QuickTest Professional software*. Web. 6 Jan. 2010. <https://h10078.www1.hp.com/cda/hpms/display/main/hpms_content.jsp?zn=bto&cp=1-11-127-24^1352_4000_100__>

[10] Mercury Interactive Ltd. "HP QuickTest Professional 10.00 Product Availability Matrix." HP, 1 Jan. 2009. Web. 6 Jan. 2010. <http://support.openview.hp.com/selfsolve/document/KM543960/binary/QTP10_ProdAvailMatrix.pdf?searchIdentifier=55217e5b:11f28414fee:-7285&resultType=document>

[11] *Selenium web application testing system*. Web. 7 Jan. 2010. <http://seleniumhq.org/>

[12] *Watir*. Web. 7 Jan. 2010. <http://watir.com/>