ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 3, pp: (233-238), Month: July - September 2015, Available at: www.researchpublish.com

Web Agent Automation Development for Web Access Management

Charan Kumar H L[1], A Thyagaraja Murthy[2]

[1]Post Graduate Student NIE SJCE Mysore, [2] Associate Professor, EC Department, SJCE, Mysore, India

***Abstract*: It is a challenging task for IT industry is to deliver software products that meet the business needs. It is a necessity to deliver software that is free of bugs. The bugs in software can cause major loss for IT organization in case they are not removed before delivery. To deliver good quality software, an efficient testing is to be conducted. Statistics say that 50% of the total cost of software development is devoted to software testing; it is even more in case of critical software. Depending on time, scale and performing methods we can classify testing. Test data can be designed either manually or automatically. Software engineering research puts large emphasis on automating the software development process that produce large more complex quantities of code with less effort. For testing this software, there is need of finding advanced innovative support procedures to automate the testing process.**

The motivation for automating test cases is to have non-programmers not lose the comfort level that they would have with the degree of informality, ambiguity (without exactness) that is inherent in natural-language expressions, and that would be closer to their thought processes. Thus, the goal is to improve the efficiency of automating the manual tests by automating the automation task. Automated software testing is automating the manual process of testing software. Automated Testing is gaining interest these days as it reduces the time and increases the efficiency.

The paper includes the automated testing development of Web access management (WAM).Web access management (WAM) is a form of identity management that controls access to web resources, providing authentication management, policy-based authorizations, audit and reporting services (optional) and single sign-on convenience. Authentication management is the process of determining a user's (or application's) identity. This is normally done by prompting for a user name and a password. Additional methods of authentication can also include access tokens (which generate one-time passwords) and digital certificates.

Once a user's (or process') identity is confirmed, policy-based authorization comes into play. A web resource can have one or more policies attached to it that say e.g. "only allow internal employees to access this resource" and/or "only allow members of the Admin Group to access this resource." The requested resource is used to look up the policy, and then the policy is evaluated against the user's identity. If the user passes the policy evaluation, she/he is granted access to the resource. If the user fails the evaluation, access is denied.

***Keywords:* Authentication, Access Management (AxM), Agent, selenium, web server, Authorization.**

I. INTRODUCTION

Access Management provides security and protects the customer resource that is stored in web sites by providing facilities like basic protection mode, challenge protection mode, mail protection mode, and phone protection mode. RSA Access Management Agent for web servers replaces or augments native webserver security mechanisms. An agent runs in the same process as the web server and is invoked whenever the web server needs to determine access rights for a particular uniform resource locator (URL). The Agent calls the Access Management Authorization Server to determine what HTTP action must be taken for the incoming request.

Access Management uses an external data store such as LDAP or SQL to store user information. Before we can grant access to a user, information about the user must be added to the data store. To make granting and denying of access to resources easier, and to help organize our Access Management system, we can create user groups. A user group can contain users called member users, and other user groups called member groups. In addition to In addition to the required information we store in our user accounts, we can assign properties to users. Properties are customizable fields that let us store organization specific data with our user records. Properties also serve as evaluation criteria for Smart Rules.Objects that we protect with Access Management are called resources. A resource can be an individual file, such as an image file, an entire directory on your web or application server, or an entire application. In Access Management, an application is a grouping of one or more associated resources. When we want to add a resource to Access Management, we add it to an application. Security policies are the link between users and resources.

AxM (Access Management) ensures a smooth implementation that suits the specific needs of customers

In planning of Access Management Implementation, we need to decide:

• How we want to protect our resources.

• How we want to administer Access Management.

• How we want to organize our users.

• Which resources we want to protect.

• Who should have access to protected resources?

II. WEB SERVER AGENT

Access Management Web Server Agents supplement the native security mechanisms of a web server. RSA Access Management Web Server Agents run in the same process as the web server itself and are invoked whenever the web server needs to determine access rights for a particular Uniform Resource Locator (URL). Access Management Web Server Agents forward access requests to an Authorization Server, which passes the answers it receives back to the web server.

RSA Access Management Agent supports the following features

1. Multiple authentication chaining: We can combine or chain together different authentication types for a particular resource.
2. Connection Types: We can secure the connections between the Agent and RSA Access Management Servers using either anonymous SSL or mutually-authenticated SSL.
3. Uniform resource locator (URL) retention: We can enable the web server to retain original page request URLs instead of sending users to a default location, such as a home page, after log on.
4. Custom user properties: We can publish user information that you have defined as custom user properties in the RSA Access Management Administrative Console. You can choose to publish the user information on authentication or on authorization to a particular resource, among other options.
5. Custom error handling: We can customize the error messages displayed to end users
6. On-demand reconfiguration: We can apply new configuration changes immediately without restarting your web server, using the command-line utility of the Agent.
7. On-demand configurable caches flush: We can flush the Agent cache based on the cache type, using the Agent's command-line utility. Passing selected arguments allows you to flush combinations of cached protected resources, unprotected resources, user properties, session tokens, “allow” authorizations, and “deny” authorizations
8. Performance variables: We can define the number of times the Agent checks the Access Management Authorization pool, define a list of URLs to be excluded from authorization checks, and define web server caching properties. These are just a few of the performance settings available in the Agent configuration file, webagent. Conf.

9. Proxy and firewall environment support: We can define how cookies are handled by the Agent when proxy servers and firewalls are in use.
10. Virtual host support: We can define virtual hosts and configure Agent behaviour separately for each virtual host.
11. Enhanced logging: We can control the number of error and configuration message logs to be recorded. You can define the amount of log header information provided and logging behaviour, including rotation of log files.
12. Agent utilities: We can encrypt parameter values, test configuration prior to implementation, perform dynamic reconfiguration, and perform cache flushing.
13. Support for non-forms-based authentication with forms-based authentication: We can enable form-based and non-form-based authentication for different virtual hosts on the same web server.
14. Authentication Only Setting: Access Management Agent extends Authentication only setting to all supported web servers.
15. Adaptive Authentication: Access Management Agent and Access Management work with the Adaptive Authentication engine to give customers a first level of authentication, before a second level of authentication with Access Management.

III. WEB SERVER AGENT AUTOMATION DEVELOPMENT TOOLS

There are various tools that help software teams build and execute automated tests. Many teams are actively using unit tests as part of their development efforts to verify critical parts of their projects such as libraries, models and methods.

The Automation developments tools that are used are:

1. Selenium
2. Log4J
3. Apache Maven
4. TestNG
5. Apache Subversion (SVN)
6. AutoIt

IV. WEB SERVER AGENT AUTOMATION DESIGN AND IMPLEMENTATION

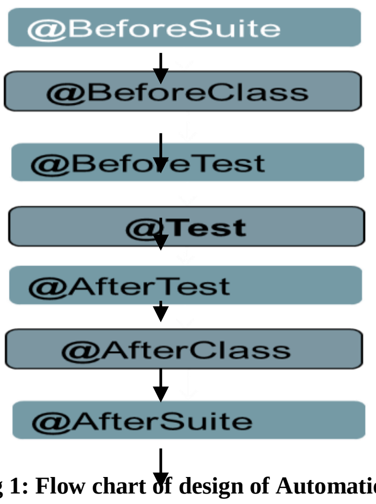


Fig 1: Flow chart of design of Automation

The Design contains following Annotations

- BeforeSuite
- BeforeClass

- BeforeTest
- Test
- AfterTest
- AfterClass
- AfterSuite

BeforeTestSuite:

1. Environment is being fetched, including webagent and server side data
2. Agent and Server Machines are checked for reachability
3. Connections to Remote Agent and Server Machine are made
4. Asp/Jsp Extension handling in Webagent

BeforeClass:

1. Initializes all the local Environment Variables
2. Edit Files Before Class, if the files include Server Side changes, remember to restart AxM Servers

BeforeMethod:

1. Initialize Page Factories.
2. Open Driver (Firefox, Chrome or IE, depending on the configurations chosen

Test Methods:

1. Clear Logs
2. Edit the Files(Usually only webagent)
3. Use the opened driver for UI Testing
4. Assert Conditions

After Method:

1. Restore the Edited Files
2. Remove the Backup Folders

After Class:

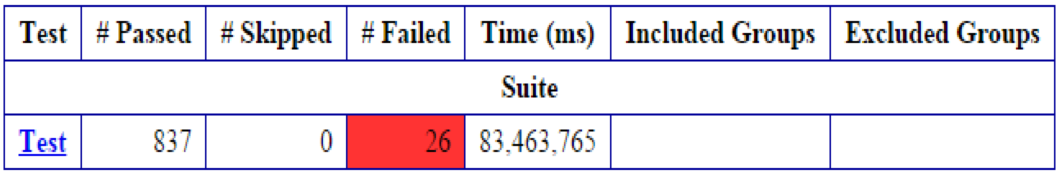
1. Driver close and quit.
2. Remove the Backup Folder.

After Suite:

1. Remove Remote Connections

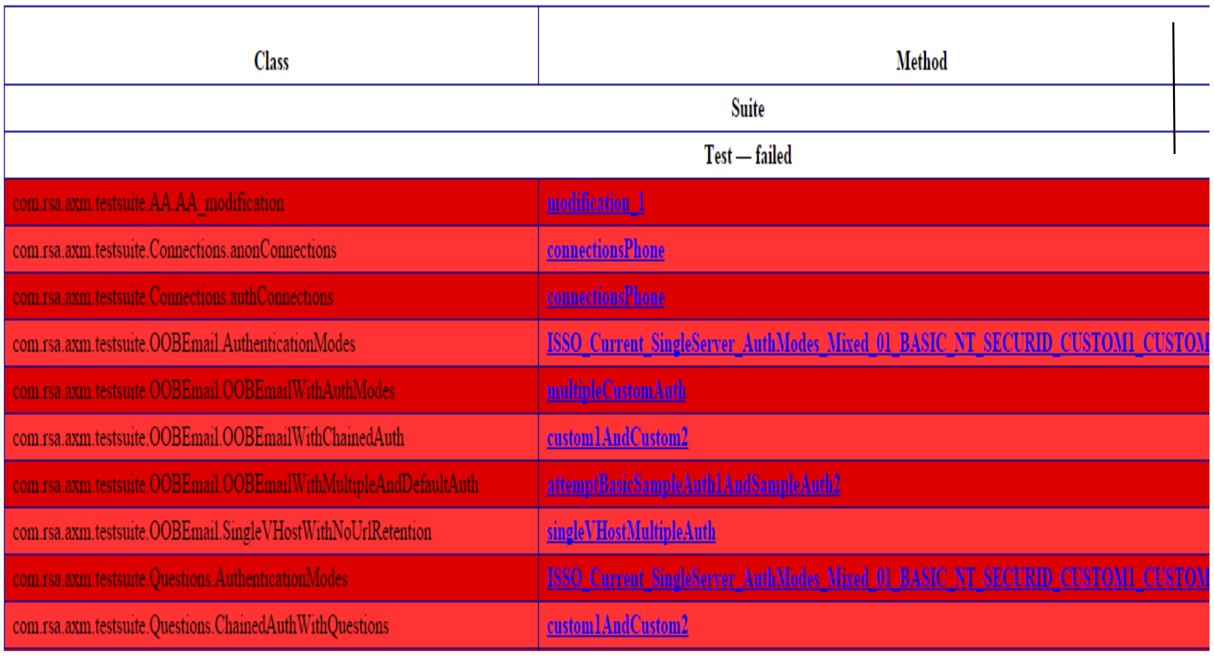
V. AUTOMATION RESULTS AND ANALYSIS

The following TestNG Emailable Report depicts the result of the test suite and the Time Taken for the Run.



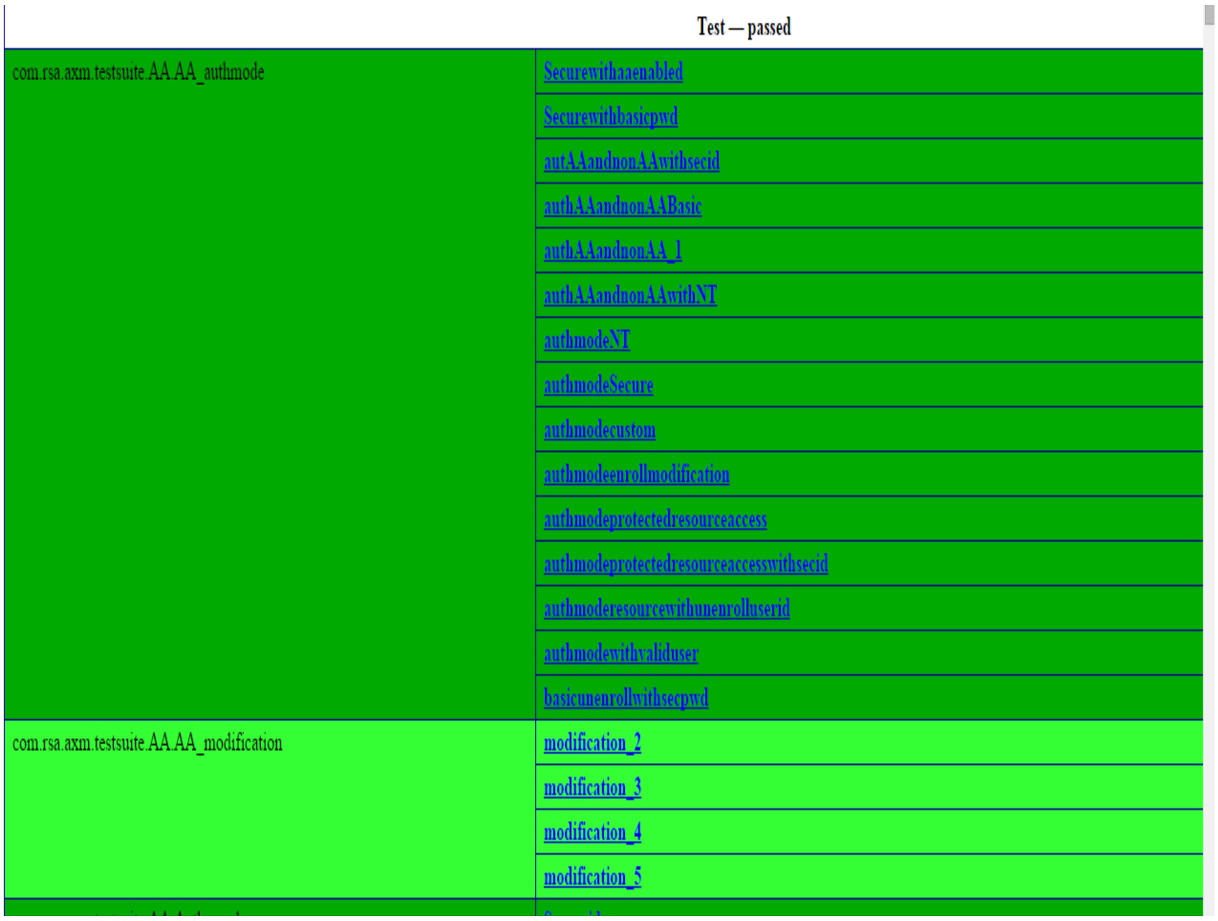
Test	# Passed	# Skipped	# Failed	Time (ms)	Included Groups	Excluded Groups
Suite						
Test	837	0	26	83,463,765		

Fig 2: Automation Results



Class	Method
Suite	
Test — failed	
com.rsa.axm.testsuite.AA.AA_modification	modification_1
com.rsa.axm.testsuite.Connections.anonConnections	connectionsPhone
com.rsa.axm.testsuite.Connections.authConnections	connectionsPhone
com.rsa.axm.testsuite.OOBEmail.AuthenticationModes	ISSO_Current_SingleServer_AuthModes_Mixed_01_BASIC_NT_SECURID_CUSTOM1_CUSTOM
com.rsa.axm.testsuite.OOBEmail.OOBEmailWithAuthModes	multipleCustomAuth
com.rsa.axm.testsuite.OOBEmail.OOBEmailWithChainedAuth	custom1AndCustom2
com.rsa.axm.testsuite.OOBEmail.OOBEmailWithMultipleAndDefaultAuth	attemptBasicSampleAuth1AndSampleAuth2
com.rsa.axm.testsuite.OOBEmail.SingleVHostWithNoUrlRetention	singleVHostMultipleAuth
com.rsa.axm.testsuite.Questions.AuthenticationModes	ISSO_Current_SingleServer_AuthModes_Mixed_01_BASIC_NT_SECURID_CUSTOM1_CUSTOM
com.rsa.axm.testsuite.Questions.ChainedAuthWithQuestions	custom1AndCustom2

Fig 3: Failed Tests from the Test suite: Failed Tests from the Test suite



Test — passed	
com.rsa.axm.testsuite.AA.AA_authmode	Securewithaaenabled
	Securewithbasicpwd
	autAAandnonAAwithsecid
	authAAandnonAABasic
	authAAandnonAA_1
	authAAandnonAAwithNT
	authmodeNT
	authmodeSecure
	authmodecustom
	authmodeenrollmodification
	authmodeprotectedresourceaccess
	authmodeprotectedresourceaccesswithsecid
	authmoderesourcewithunenrolluserid
	authmodewithvaliduser
	basicunenrollwithsecpwd
com.rsa.axm.testsuite.AA.AA_modification	modification_2
	modification_3
	modification_4
	modification_5

Fig 4: Passed Tests from the Test Suite

VI. CONCLUSION AND FUTURE SCOPE OF IMPLEMENTATION

1. Conclusion:

"Automated Testing" is very important because it uses strategies, tools and artifacts that augment or reduce the need of manual or human involvement or interaction in unskilled, repetitive or redundant tasks. The project provides complete Automation Framework for Access Manager Web Agent. Eight hundred and fiftytest cases pertaining towebagent have been automated providing Reliability, Reusability, and Cost reduction. The Implementation can be run on windows or Linux, using web servers Apache, IHS, IIS, and Sun Java.

2. Future scope of implementation:

The complexity of Automation increases rapidly as the Software/technology demands additional features. This happens when Software or Technology itself becomes complex. This calls for the future implementations of Automation as it need to sustain the growth of complexity. The Future Implementations may include...

- Detection of sounds and flashes
- Self-Healing Features
- Automation software as a plug in to the product or the software
- Detection of Images.

REFERENCES

[1] Kolawa, Adam; Huizinga, Dorota (2007). Automated Defect Prevention: Best Practices in Software Management. Wiley-IEEE Computer Society Press. p. 74. ISBN 0-470-04212-5.

[2] Elfriede Dustin et al. (1999). Automated Software Testing. Addison Wesley. ISBN 0-201-43287-0.

[3] R. Benato and A. Paolucci, EHV AC Undergrounding Electrical Power. Performance and Planning. New York: Springer, 2010.

[4] Mark Fewster & Dorothy Graham (1999). Software Test Automation. ACM Press/Addison-Wesley. ISBN 978-0-201-33140-0. [5]. Roman Savenkov: How to Become a Software Tester. Roman Savenkov Consulting, 2008, ISBN 978-0-615-23372-7

[5] “Selenium Commands” Selenium Documentation. Retrieved September 9, 2011.

[6] "Running JUnit tests from Maven2". JUnit in Action (2nd ed.). Manning Publications. 2011. pp. 152–168.

[7] Maven Build Customization. Packt. 2013. pp. 1–250. ISBN 9781783987221.

[8] Mastering Apache Maven 3. Packt. 2014. p. 298. ISBN 9781783983865.