Analysis and Identification of Cross Browser Inconsistency Issues in Web Application using Automation Testing

Nepal Barskar[1]
PG Student , CSE Department
IET, DAVV,Indore(M.P.) India
nepal.rgtu3@gmail.com[1]

C. P. Patidar[2]
Assistant Professor, IT Department
IET-DAVV, Indore(M.P.) India
cpatidar@ietdavv.edu.in[2]

Dr. Meena Sharma[3]
Professor, CSE Department
IET-DAVV, Indore(M.P.) India
msharma@ietdavv.edu.in[3]

***Abstract*— Web technologies of inconsistency and Web standards it to be consistent with different web technology evolution, Web application developers has been broadly rang of face on the different certain problem . Web applications become difficult part for them to keep track problem of their web correctly rendered across broad range of browsers and platforms. It kind of cross-browser inconsistency (XBI), developers are keep checking that document and platform produced by the web application is appropriate pass the across every useful browser-platform so many combinations. It take the requires more than execution time and error results in web application. This web cross-browser inconsistency existing testing tools for speed up the process of automating and the rendering of a document in cross browsers and different platforms, and using either image analysis, and Document Object Model (DOM) analysis to the feature of cross-browser inconsistency. This dissertation are comparisons of the problem of cross-browser inconsistency testing with the modern web applications and their functionality check the accuracy of web application's behavior with different different web browsers and their present a solution. In the reasons of cross-browser inconsistency issues and their so many solutions to them are been presented feature different cross-browser inconsistency. Proposed solution has been used the concept of Automation Testing using Behavioral methodology of the web application on different browsers such as Firefox, Internet Explorer and Google Chrome. For the test bed selenium and TestNG variation has been integrated onto the eclipse java environment. Obtained results has proven that the proposed XBI validation method outperform and provides the cost effective solution instead of paying high cost to other testing service provider lie "Browser-stack".**

Keywords- Automation testing, Web Driver, Cross-browser inconsistency (XBI), IE tester, Selenium, TestNG, web application, web server, software testing, web testing Introduction

I. INTRODUCTION

Advancement in cloud and Internet Technologies and also advances the role of the browsers. Browsers are the main interfaces to deliver/access the information in one click. Web is been considered as a platform for various web applications & websites to communicate with the users by sharing resources such as data, audio, video, applications. These web applications are used for interacting over wide network to cover many purposes [1-6].

More and more things are shifting to the Internet. Applications that are built for offline use are now available online [7, 8]. People have the need to be online [9]. Some web applications are publicly available while others are private and only accessible through login credentials [10]. Web applications are built with multiple programming languages and can make use of different design patterns. One way of building web applications is by using the .NET framework combined with the MVC design pattern. The MVC (Model View-Controller) pattern is used to divide a complex system into three components, each component with their own responsibilities. The model contains data, the view the presentation, and the controller the logic of the system. The web applications we are currently interested in are based on the .NET framework supported with this MVC design pattern [11]. For many years, when applications, for offline purposes, were developed less effort was spent in testing the software.

Nowadays more effort is spent in testing and measuring quality of software. One of the reasons is that customers demand a higher quality of software. Testing is now integrating in development methodologies, like agile development [12]. With the latter, testing is a part of the development cycle. The development of software contains at least several cycles and this amount is increasing depending on the size of the software. In each cycle, testing is performed on the developed code. By testing during these development phases more faults can be detected, resulting in more sustainable and higher quality software. The gain in sustainability results from the fact that solving faults during development time or even preventing them on the drawing board is cheaper than fixing the same faults afterwards [13]. The software is of better quality, because we know the system is test and possible faults that could occur are prevented. However, this does not guarantee that the software is free from

bugs. It is possible that faults in the software still exist but were not detected with testing.

While more effort is nowadays being spent on the testing of normal software applications, there is still a lack in testing effort during the development of web applications. Simple web applications, that only have the purpose of serving informational content, are often either insufficiently tested or not tested at all [14, 15].

Hence, these web applications still contain faults. A specification is often used during the testing of software. Otherwise, we do not know what we have to test. However, in most of the cases there is not any specification available, when developing web applications. Most of the time there is no documentation available. This makes knowing what to test hard.

A great example is the lack of proper documentation at the company Bluenotion for small web application. From experience, we know that the actual design of the web application is the only available documentation. Furthermore, the small web applications are badly tested and still contain faults after the application is going live.

Hence, these web applications still contain faults. A specification is often used during the testing of software. Otherwise, we do not know what we have to test. However, in most of the cases there is not any specification available, when developing web applications. Most of the time there is no documentation available. This makes knowing what to test hard.

A great example is the lack of proper documentation at the company Bluenotion for small web application. From experience, we know that the actual design of the web application is the only available documentation. Furthermore, the small web applications are badly tested and still contain faults after the application is going live.

This paper comprises the problem of cross-browser inconsistent testing of the modern web applications as a functional accuracy check of web application's behaviour with various web browsers and present a solution for it. Proposed method's adopted the principal of Agile methodology to test the BDD (Behaviour Driven Development) and ATDD (Acceptance Test Driven Development) of the web application. Also, reasons of cross-browser inconsistency issues and solutions to them are been presented inconsistency issues and solutions to them are been presented. Proposed solution has been used the concept of Automation Testing using Behavioral methodology of the web application on different browsers such as Firefox, Internet Explorer and google chorme. For the testbed selenium and TestNG variation has been integrated onto the eclipse java environment. Obtained results has proven that the proposed XBI validation method outperform and provides the cost effective solution instead of paying high cost to other testing service provider lie "Browser-stack".

Rest of the article is organized as follow, Section II presents the brief introduction about the Automation Testing using Selenium TestNG, whereas Section III presents proposed idea to enrich the XBI (Cross Browser Inconsistency) detection using BDD automation Testing, Section IV discusses the outcome of the proposed methodology and it's performance comparasion with the existing technique. And finally Section V concludes the papers with the future directions of this work.

II. AUTOMATION TESTING USING TESNG

A. TestNG

TestNG is an open-source testing framework written in Java and inspired by JUnit and NUnit. The framework was created as a response to problems in the framework JUnit 3.x. It was created at a time when JUnit 4 which addresses those deficiencies was not released yet. TestNG contains improvements [16] over the JUnit as:

- Wider set of annotations (e.g. @BeforeClass)
- Support for parametric and data-driven testing (@DataProvider annotation).
- Powerful execution model (no more TestSuite).
- The ability to run tests in parallel and in multiple threads
- Greater possibility of the configuration during the execution of tests (e.g. the ability to configure different test groups and run them under different conditions, test dependencies) Creating TestNG tests consist typically from three parts:
- Create business logic to be tested and annotate test methods.
- Add configuration information (e.g. names of classes or groups to be executed) in to the testng.xml or the testng.xml configuration file.
- Execute Tests.

As shown in the figure 1-

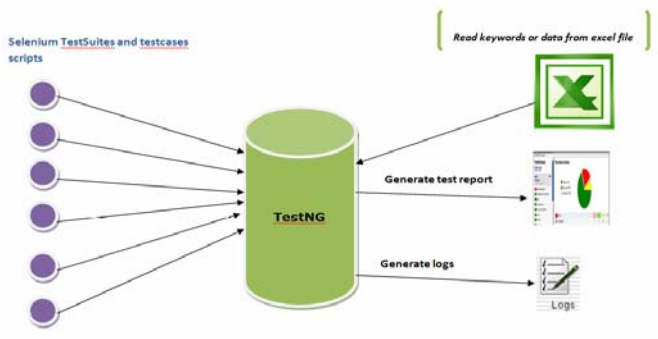

Figure. 1 TestNG Architecture

TestNG is available as a plugin to all of the most widely used software development environments from which tests can be easily configured and executed. Very important is also functionality to produce structured reports using TestNG Reporter API.

B. ReportNG

RerportNG is a simple plugin for TestNG framework, that has been developed as a replacement for proprietary reporting tool supplied with TestNG.

It provides a simple and well arranged view of testing results. Compared with basic TestNG tool test reports are more transparent and easier to read which is probably the biggest advantage. In addition to conventional HTML report ReportNG is able to generate XML output, which allows it to be integrated with tools for CI such as Hudsonx.

C. Selenium

Selenium is a set of different software tools designed to automate testing of web applications. The suite is developed by the company ThoughtWorks since year 2004. All of these tools together provide a rich set of testing functionality, compatible with all the most widespread software platforms and web browsers. In addition to standard browsers Selenium also supports virtual browser HtmlUnit, implemented in Java. This browser does not have a graphical interface and servers solely as a JavaScript interpreter to simulate the real web page. All components of the project are released under opensource license Apache 2.0. Under this license the source code of Selenium is freely available to be used for commercial purposes. With regard to the fact that Selenium is open-source project is the technical support provided by a large community of volunteers and it is available in form of user discussion groups [17]. Another way of the Selenium support is the Selenium Bug Report[17] system, that offers simple way to report found issues. There is also possibility of the commercial support, which is currently provided by more than 20 companies worldwide.

The Selenium IDE is a prototyping tool for creating automated tests, which is implemented as a plugin for the web browser Mozilla Firefox from the version 3. It provides simple interface that contains the record functionality.

By using this functionality the system can directly record steps performed by a user when using the web application. User's actions are recorded as triplets of values:

- Command - representing performed action
- Target - identifying element for which the command is performed.
- Element can be identified by the name, the attribute or using the XPath expression
- Value - set by the command to target. It may also be empty. Each such triplet is one command for the Selenium IDE. These commands are called Selenese and they are used to perform tests. Selenium

offers a wide range of these commands that create whole test language.

Commands can be divided into three groups:

- **Actions** - commands used to manipulate the state of the application (e.g. Click on the button)
- **Accessors** - used to detect application's state and to store results in a variable (e.g. storeTitle). Accessors are also used to automatically generate assertions.
- **Assertions** - used to compare current state of the application to the expected state (e.g. assertText)

With the basic knowledge of the Selenese HTML syntax and the Selenium IDE it is possible to create automated tests very fast and easily, without closer programming skills. For completeness, specific example of a simple test recorded in the Selenium IDE.

Example is showing the simple test of logging into the web application. Uploaded test is for proper functioning supplemented by an assertion. At the bottom part of the figure is notation using the Selenese syntax. Vertical lines in the notation represent particular table cells.

From tests created by this method the Selenium IDE can also generate scripts in programming languages C#, Java, Python, Perl. The Selenium IDE is also able to generate scripts for other unit test frameworks. Supported frameworks are JUnit, NUnit(.Net), RSpec(Ruby), unittest(Python) or TestNG.

The disadvantage is that precisely because of the simplicity and associated small set of useful functions the Selenium IDE does not allow to use of conditions and cycles required to create complex test scenarios. At the same time it is quite difficult to reduce duplicate code and this disadvantage is causing great difficulty in tests maintenance. The tool is therefore characterized by a good level of productivity for a small number of tests, but productivity and usability decreases with their widening number. Because of these drawbacks Selenium IDE is intended solely as a quick prototyping tool to support the development of automated tests by using other tools from the Selenium family.

1) Selenium Remote Control (RC)

It is the older version of the Selenium API, that has the server as the main Selenium project until it was merged with the Selenium WebDriver. It is still supported for backward compatibility, but is no longer actively developed. However the Selenium RC still provides some implemented in Selenium WebDriver, such as the support for older versions of web browsers. It consist of two basic components:

- **Selenium Server** - The server side takes care of the administration of browser windows in which tests are performed. It is also responsible for sending and executing the commands for the web application and also acts as a proxy between the tested application and the web browser. Server is written in Java and can by run just like any other Java application, therefore requires no installation.
- **Client libraries** - The client part contains libraries that are used to create tests in Java, C#, Perl, PHP, Python and Ruby.

The primary object used to create tests is the browser, represented by an instance of a class which implements interface Selenium. Via the browser object various features (e.g. type or click) can be called and also the current state of the application (e.g. getText) can be queried. For the determination of the attribute with which the method should work text identifier is used. The identifier is determined by name in the DOM model or by using XPath expressions.

2) Selenium 2 (WebDriver)

Selenium 2 is the latest framework from the Selenium family, which has been created as a combination of the Selenium RC and the project WebDriver. It is also called Selenium WebDriver[17]. WebDriver has been designed as a simple and accurate API that repaired some of the shortcomings and limitations of the Selenium RC. It has a better support for dynamically generated web pages, in which the previous version lacked support. The main difference is in the WebDriver approach to the browser. Each of supported browsers is using a special controller, using that WebDriver communicates directly with the web browser. It does not send commands using the JavaScript as the Selenium RC does but instead of this it communicates directly by using browser native bindings for the test automation support.

Another difference with the Selenium RC is that there is no need to start the Selenium Server in case that tests are going to be executed locally on the same computer and they will only used WebDriver API. There are drivers available for all most common operating systems and related browsers starting with versions:

Google Chrome 12+, Internet Explorer 6+ (both 32 and 64-bit version), Opera 11.5+, Mozilla Firefox 3.6 - 18, Safari 5.1+, HtmlUnit 2.9. Support for two most prevalent mobile platforms has also been recently added and WebDriver now supports: Android 2.3 + (the physical device and emulator), iOS 3+ for smartphones and iOS 3.2+ for tablets (the physical device and emulator).

D. Selenium Grid

Selenium Grid is a tool enabling parallel execution of various automated tests using multiple remote test environments with different web browsers. The tool is suitable for faster and more effective testing of large and time consuming test suites that must be executed on a variety of test environment configurations. Grid consists of so-called Hub and one or more Nodes.

The Selenium Server is running independently on each of these nodes. The Hub maintains configuration information for each authorized node and it is responsible for routing of test requirements to the Selenium RC node with the correct configuration (e.g. Widows 7 with Firefox web browser). If we want to take advantage of the Selenium Grid all automated tests must be able to run in parallel by using features for the parallel test execution delivered by for example the TestNG framework. Using the Selenium Grid and one of the frameworks for unit testing can significantly reduce time for executing the entire automated test suite because there is no need to wait until the single test is completed and tests can be distributed on several test environments.

The selenium architecture as shown in the figure 2-

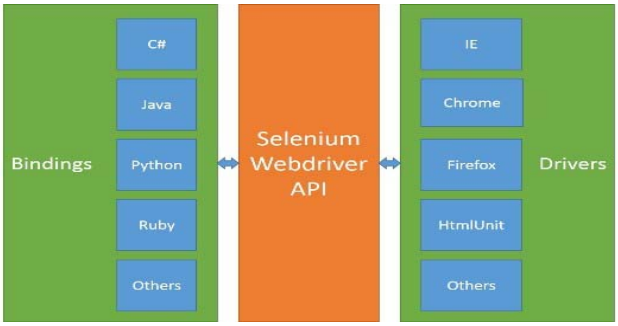

Figure. 2 Selenium Web Driver Architecture

E. Tellurium

It is a tool for automated testing of web applications based on the Selenium RC framework. The tool has been developing since 2008 and it is regularly updated. Similar to the Selenium Tellurium is open-source project licensed

under Apache 2.0 license and freely available for commercial use. The motivation to develop this tool was to remove the lack of robustness emerged by using other frameworks, especially the Selenium. It uses different philosophy in the approach to each element on the website. While the Selenium accesses each DOM element like buttons or text fields as a separate object. Tellurium treats all user interface elements as one unit grouped into

so-called UI models. Each element is defined by the unique identifier (uid). By using this uid elements are referred when called by a function. To define the UI modules and functions for different types of elements (e.g. button and corresponding action click) scripting language Groovy is used. By the separation of elements definition and the test code Tellurium increases the robustness of developed tests. The framework is thus able to adapt on minor changes of elements and their attributes during the development phase of the application, because element locators are not fixed, but dynamically generated at the test run time. These locators can have different formats including JQuery or XPath expressions.

III. PROPOSED XBI TESTER

Web browsers are the best medium to access /serve to the customer or information locally and remotely. Now browsers are playing very important role now a days, traditionally they just content static to dynamic pages now they integrate with highly configured interfaces to deliver the banking needs to the cloud services. Browsers are the combination of so many UI (User Interface) back-end (java, php, servlet and databases). Platform independency (thick to thin device i.e. Desktop/Laptop to mobile/tab/kindle) is main SQA (Software Quality Assurance) parameters for the browsers.

Software Testing: Software testing is a wide engineering field, which has an irreplaceable position in the process of software development. According to the definition proposed by IEEE Std 610.12[18] we can define software testing as [19]:

- The process of operating a system or component under specified conditions, observing or recording the

results, and making an evaluation of some aspect of the system or component.

- The process of analyzing a software item to detect the differences between existing and required conditions (that is, a bug) and to evaluate the features of the software items

A. *Problem Identification*

Automated testing has potential to track bugs in software and their performance at the end user. This section will present an advanced methodology for evaluating the XBI (Cross Browser inconsistency) for the web application.

Following problems has been identified as a dissertation objective, and extract following issues which needs to be addresses as a proposed solution in context of the XBI:

1. WebPages viewing content (UI), inconsistency and different viewing experience in all different browsers.
2. performance (loading, response, error)
3. Duplicate reports
4. Cross-Platform Error

B. *Proposed Method*

Proposed work presents an cost effective and faster solution to check the cross browser inconsistency issues such a:-

- Viewing content (UI), performance (loading, response, error) of the WebPages in the browsers.
- Execution Time on Different Browsers
- Automated Test Case's

IV. PERFORMANCE AND RESULTS ANALYSIS

Advancement in the Internet/internet technologies has gives wings to information sharing resulting the faster delivery/access of the information around the globe. Development of WWW and browsers technologies enhancement gives ease to the users to do the same.
Proposed idea is to overcome the cost in terms time and money to do the testing of the web application.

A. *Experimental Setup*

Following environment has been setup for the evaluation of the proposed light weight automated XBI testing performance of various browsers such as Internet explorer, Mozilla Firefox, Google chrome etc.

1. Web application (which has to be tested)
2. Selenium web driver and server (standalone)
3. Eclipse and TestNG plugin
4. Jdk and jre 1.7 or above

B. *Results*

1) Execution Time of Webapplication under various browsers-

A proposed solution has outperformed as compared to the existing solutions and can test the application more faster. As shown in figure 3-

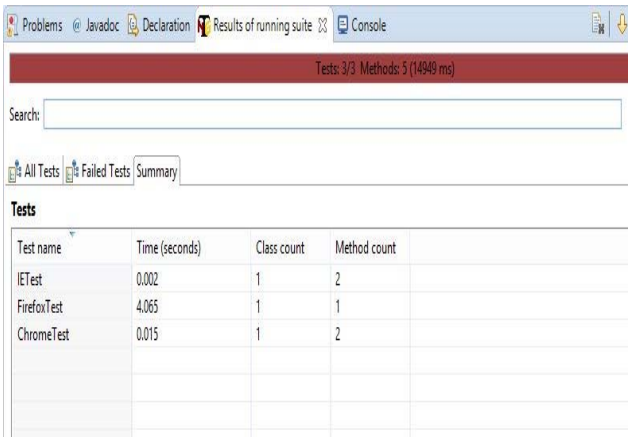

Figure 3. Execution Time on Different Browsers

2) Test Results-

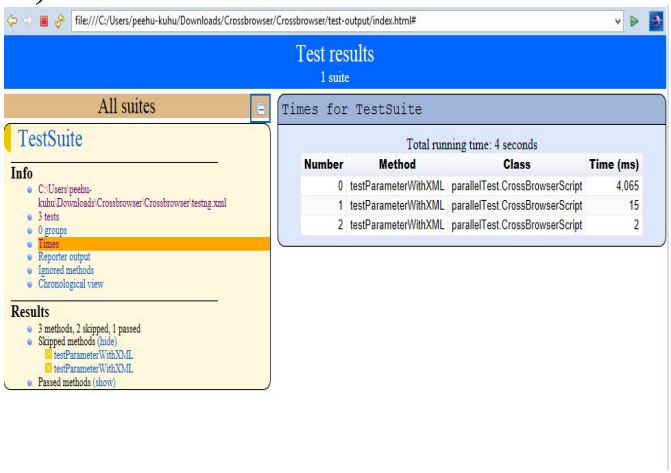

Figure 4. (a)Test Results

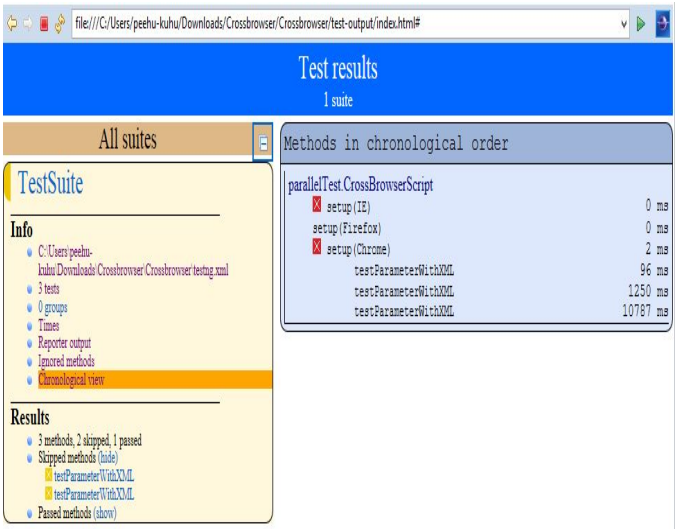

Figure 5. (b)Test Results

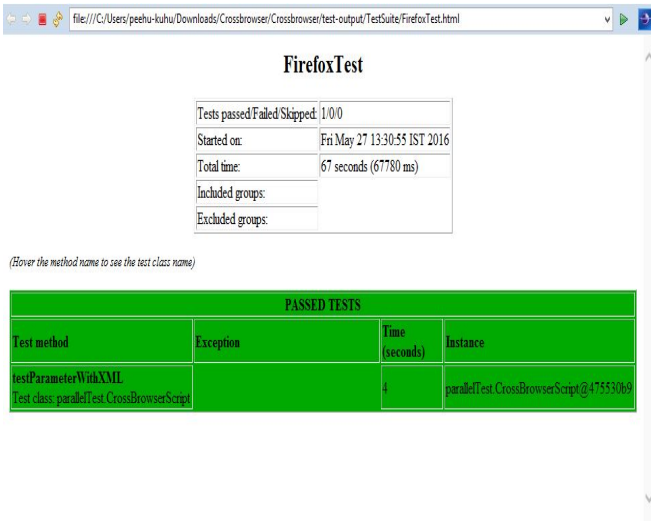

Figure 6. (c) Browser Specific Test Results

V. CONCLUSION

In this paper, the proposed methods five the idea to resolve the problem of cross-browser inconsistent testing of the modern web applications as a functional accuracy check of web application's behaviour with various web browsers and present a solution for it. Also, reasons of cross-browser inconsistency issues and solutions to them are been presented feature out cross-browser inconsistency. This paper comprises the problem of cross-browser inconsistent testing of the modern web applications as a functional accuracy check of web application's behaviour with various web browsers and present a solution for it. Also, reasons of cross-browser inconsistency issues and solutions to them are been presented inconsistency issues and solutions to them are been presented. Proposed solution has been used the concept of Automation Testing using Behavioral methodology of the web application on different browsers such as Firefox, Internet Explorer and google chorme. For the testbed selenium and TestNG variation has been integrated onto the eclipse java environment. Obtained results has proven that the proposed XBI validation method outperform and provides the cost effective solution instead of paying high cost to other testing service provider lie "Browser-stack".

REFERENCES

[1] S. Roy Choudhary and A. Orso, "Webdiff: Automated identification of cross-browser issues in web applications," in ICSM '10: Proceedings of the International Conference on Software Maintenance. IEEE, September 2010.

[2] Choudhary, S. R., Prasad, M. R., & Orso, A., "Crosscheck: Combining crawling and differencing to better detect cross-browser incompatibilities in web applications", In Software Testing, Verification and Validation (ICST), 2012 IEEE Fifth International Conference on, pp. 171-180, 2012.

[3] Choudhary, S. R., Prasad, M. R., & Orso, A., "X-PERT: A Web Application Testing Tool for Cross-Browser Inconsistency Detection", 2014.

[4] S. Roy Choudhary, M. R. Prasad, and A. Orso. X-PERT: Accurate Identification of Cross-browser Issues in Web Applications. In Proceedings of the 2013 International Conference on Software Engineering, ICSE '13, pages 702-711. IEEE Press, 2013.

[5] Patidar, C., "Cross Browser Testing: A Challenge for Web Testing", International Journal of Scientific Research in Computer Science and Engineering, Vol-1, Issue-3, pp.-62-64, 2013.

[6] Nepal Barskar and C.P. Patidar, "A Survey on Cross Browser Inconsistencies in Web Application", International Journal of Computer Applications (0975 – 8887) Volume 137 – No.4, March 2016.

[7] Newell, D. 6 Reasons You Should Buy An Online Versus Offline Business. 2014 [cited 2015 12-05-2015]; Available from: http://feinternational.com/blog/6-reasonsbuy-online-versus-offline-business/.

[8] Naseer, N., 4 reasons to shift your offline CRM to the cloud now. 2014.

[9] Post, H. A Shift from Online to Offline: Adolescence, the Internet and Social Participation. 2014 [cited 2015 12-05-2015]; Available from: http://www.huffingtonpost.com/undergraduate-awards/a-shift-from-online-toof_b_4431523.html.

[10] Catone, J. Adobe Preparing Full Shift to Web Apps. 2007 [cited 2015 12-5-2015]; Available from: http://readwrite.com/2007/10/18/adobe_preparing_full_shift_to_online_apps.

[11] Microsoft. ASP.NET MVC. [cited 2015 6-4-2015]; Available from: http://www.asp.net/mvc.

[12] Highsmith, J. and A. Cockburn, Agile software development: the business of innovation. Computer, 2001. 34(9): p. 120-127.

[13] The Incredible Rate of Diminishing Returns of Fixing Software Bugs. 2009 [cited 2015 14-5-2015]; Available from: http://superwebdeveloper.com/2009/11/25/the-incredible-rate-ofdiminishing-returns-of-fixing-software-bugs/.

[14] magazine, I.s. Three-quarters of US small firms have not tested website cybersecurity. 2011 [cited 2015 12-05-2015]; Available from: http://www.infosecurity-magazine.com/news/three-quarters-of-us-small-firmshave-not-tested/.

[15] Pugh, T. Contractors say late changes, lack of testing doomed health care website launch. 2013 [cited 2015 12-05-2015]; Available from: http://www.mcclatchydc.com/2013/10/24/206431/contractors-say-latechanges-lack.html.

[16] BEUST, C.: TestNG - Welcome. 2004, [online], [cit. 2013-03-13]. Available at: http://testng.org/doc/index.html.

[17] Selenium Projects. [online], [cit. 2013-03-13]. Available at: http://docs.seleniumhq.org/projects.

[18] IEEE Standard Glossary of Software Engineering Terminology. 1990, [online], [cit. 2013-03-26]. Available at:

http://web.ecs.baylor.edu/faculty/grabow/Fall2011/csi3374/secure/Standards/IEEE610.12.pdf.

[19] ODVARKO, J.: Firebug - Web Development Evolved. 2013, [online], [cit. 2013-03-20]. Available at: http://getfirebug.com.