A Systematic Review on Regression Testing for Web-Based Applications

Anis Zarrad*
Department of Computer Science and Information Systems, Prince Sultan University, Riyadh, Saudi Arabia.

* Corresponding author. Tel.: +966114948531; email: azarrad@psu.edu.sa
Manuscript submitted December 12, 2014; accepted April 16, 2015.
doi: 10.17706/jsw.10.8.971-990

Abstract: Web-based applications and their underlying parts are growing rapidly to offer services over the internet around the world. Web applications play critical roles in various areas such as community development, business, academic, sciences etc. Therefore their correctness and reliability become important factors with each succeeding release. Regression testing is an important means to ensure such factors for every new released version. Regression testing is a verification process to discover the impact of changes in other interconnected modules. With the goal of selecting an appropriate regression testing approach to respond adequately to any changes in Web applications, we conduct a complete systematic study of regression testing techniques in Web applications. Out of 64, we identified a total of 22 papers reporting experiments and case studies. We present a qualitative analysis for used tools, an overview of test case section techniques and empirical study evaluation for every selected work. No approaches were found clearly superior to other since results depend on many varying factors and the deployment environments. We identified the need of evidences where approaches are evaluated cost effective rather than technical description.

Key words: Regressing testing, web-based application testing, empirical study, test set, software testing.

1. Introduction

In today's scenario, as the world became global and with the advent of internet technologies, Web-based applications become more effective manner for enterprises, and academic entities to produce business strategies and policies. Web applications help you to gain information, collect data and to find out anything you want. The development of Web-based application differs from the traditional software development cycle; due to the heterogeneous and dynamic nature of the such application, where their components are built on different technologies and protocols like JSP, JS, PHP, XML, ODBC, JDBC, CSS, HTML5, HTTP etc… as well as the necessity to maintained regularly by different developers around the world. As a consequence error-prone is very high when maintenance is required. Therefore, efficient regression testing is an important issue, even crucial, for organizations/enterprises to ensure correctness, and reliability of Web components in new released versions.

Regression Testing involves a re-testing approach of system components and guarantee that any modification have not caused unintended system functioning. For that there is a need to re-use test cases already created in previous release and compare expected results with previous ones. Chittimalli *et al*. [1] indicate that regression testing takes more than 80% of the total testing budget and more than 50% of software maintenance cost is employed on testing. Regression testing is a well investigated field in software

engineering community. Authors in [2]-[4] suggested more than 200 papers presenting various techniques, tools, and surveys about regression Testing (RT). A comprehensive survey study was presented in [4]. While regression testing has been received a great deal of research effort in many software domains such as test case selection based on code changes [5]-[9] and specification changes [10]-[12], regression testing for database applications [13]-[15] , and regression testing for GUI [16], [17], contrary regression testing for Web applications has received relatively limited attention from testing research community [18]-[20].

The main purpose of our systematic literature review is to identify the most influential and creative techniques for Web application regression testing; justify why those techniques are so important for RT, and how can we properly use those techniques in Web applications. Also, we investigate the tools proposed and their efficiency in RT evaluation approaches. We presented the contents and structure of this systematic review report following Kitchenham's method [21].

The paper is structured as follows: Section 2 introduces regression testing in Web applications and their importance. Section 3 overviews the research methodology and research questions including Data sources, search strategy, and selection criteria. Section 4 summarizes research works on regression testing for Web applications, the results and evaluation findings. Section 5 discusses the unresolved problem, and the area of success in regression testing for Web applications. Finally, the paper concludes in Section 6.

2. Regression Testing in Web-Based Applications

The general topic regression testing was first presented in paper conference in June 1972 at the inaugural software testing conference in North Carolina [22]. Hartmann [23] chronicled the journey of regression test selection over the past thirty years with most important milestones in its development process. Early references to regression testing can be found in [23], [24]. In 1981 Fischer *et al.* [25] suggested a general approach for regression testing using linear equations to define the relationships between test cases and basic blocks (single-entry, single-exit sequences of statements in a procedure). The proposed approach is applicable to static web application.

Testing Web application's components after integration is vital for quality assurance and maintenance purposes. Regression testing is one of important testing techniques to discover failures and defects in multiple released versions. According to [26] regression testing is defined as a verification process to ensure that previously functioning components are unaffected after rewriting and or modifying code.

2.1. Process of Regression Testing in Web Application

The process of regression testing for Web application can be defined as follow:

Version 1

1. Develop web application – WA1
2. Test WA with a required test set S1
3. Evaluate test results
4. Release WA1

Once a modification is required (Version 2 – WA2) a regression testing is needed.

Version 2 – New release

1. Modify WA1 into WA2
2. Test WA2 for new functionalities
3. Perform regression testing on WA2 to ensure that the code carried over from WP1 was not affected and run correctly

- ✓ **Test Case minimization**: Identify a test set S2 derived from S1
- ✓ Run S2 and evaluate test results

4. Release WA2

The regression testing is an iterative process and can undergo as long as we have a new release for a specific Web application.

2.2. Importance of Regression Testing in Web Applications

The reader will notice that starting from 2005; the number of publications related to RT in Web application is increased. An explanation of this phenomenon is related to the appearance of new Web technologies such as AJX, HTML5, PHP, JSP, and ASP.net; to develop fast and dynamic Web pages. Web application development becomes an easy task. Therefore, the frequency of Web maintenance process is increased and need to re-test the application in cost effective way to ensure quality for new releases. Contrary to static Web where change takes effect only when Web developer updates and publishes the file again in the server.

Regression testing for Web applications must be cost effective, efficient and low time consuming testing techniques. In [27] authors mentioned three main reasons for testing Web applications; (1) to ensure that the expected software can run smooth on different Web browsers, and various operating systems. (2) Maintaining the security and protection from unauthorized access. (3) Verification process to ensure that navigation functions based on hyper-textual links are not affected. To increase Web application quality in practice, regression testing must be applied in accordance with functionalities updates.

3. Research Methodology

This systematic review follows the protocol proposed by Kitchenham [21]. In this section we first introduced the research questions. Then we discussed libraries, search string, and the selection approach used in this study.

3.1. Research Questions

The objective of this research is to better understand regression testing approaches in Web application, characterize the approaches proposed in the literature, the tools efficiency and adopted experiments methods. The research questions are divided into two main categories:

1) General Research questions (GQ)
   a) GQ1: What is regression testing (RT)?
   b) GQ2: What is the Importance of RT in Web applications?
2) Focused Research questions (FQ)
   a) FQ1: What techniques and methods are used in RT?
   b) FQ2: How efficient are the used tools in Web applications RT?
   c) FQ3: How evaluation approaches are carried? Empirically evaluated?
   d) FQ4: What are the best practices in Web applications RT?
   e) FQ5: What are the limitations of Web applications RT?
   f) FQ6: What are the future scopes of RT techniques?

General questions (GQ1 and GQ2) are already discussed in Section 2. Focused research question 1, 2, 3 and 4 are handled in the Section4. Remaining questions are considered in the discussion and conclusion sections.

3.2. Search and Selection Standards

In order to perform a rigorous study on research papers related to our selected topic, we emphasize on widely used languages French and English. By using search terms and search strings, we scanned relevant and reliable articles in literature resources. We narrow our search using the following key words: <Regression> and (<test> or <Testing>) and (<Web> or <web application> or <web based application> or <Web system>) \ <test de régression > and <application Web>.

The start search year was set to 1994 when the Internet became available to the public. In 1995, the Internet has become a platform for a large number of users [28]. Also we may mention in this regard that in January 1983, TCP/IP became the official standard [29] and the first Internet tentative starts around the year 1960 as stated in [30]. In this systematic review, we searched related sources in six electronic resources and databases (IEEE eXplore, Elsevier, ACM Digital library, Springer, and Wiley Online Library).

We built a repository for regression testing in Web application which includes more than 60 papers from 1995 to 2014. We discard papers not written in English and French We also searched for Master and PhD that have made a significant contribution to the development of regression testing for Web applications. Results are listed in Table 1.

Table 1. A List of Master and PhD Works on Regression Testing in Web Application

Author/ Supervisor	**Title**	**Type/ Year**	**University**
D. Roest / Dr. Ir. A. Mesbah	Automated Regression Testing of Ajax Web Applications [31]	MSC/ 2010	Delft University of Technology Netherlands
Florian Haftmann	Regression Testing on Web-based Information Systems [32]	MSC/ 2012	Not mentioned
Kinga Dobolyi / Dr. Westley Weimer	An exploration of user-visible errors in Web-based applications to improve Web-based applications [33]	PhD/ 2010	University of Virginia
Tamim .A. Khan/Dr. Reiko Heckel	Model-Based Testing Using Visual Contracts [34]	PhD/ 2012	University of Leicester

As a result of searching Master's theses and PhD dissertations, we found a very limited number. This is can be explained by that regression testing for Web application is still in an early stage. Lastly, we went through a selection process to filter unrelated papers. A two-stage selection methodology is used, which aims in finding efficient and contemporary approach in regression testing for Web applications.

First stage: abstract papers are analyzed for preliminary selection. Many papers described testing approach in general for Web application. These papers are rejected an example of [35]. About 73 papers are selected initially. The following question is asked: Is the study focus specifically on regression testing for Web application?

After the first stage only 31 papers remained. The purpose on relying on abstract rather than title's article in the first stage is because some publications are not mentioning regression testing in their titles but it is covered in the content; example article [36].

Second stage: papers content are evaluated carefully to select relevant articles. The following questions are asked:

1) Are the techniques and tools described in the paper support current Web technology? For example paper could discuss regression techniques [25] that can be applied in static web application, but cannot support dynamic technology such as JAX, .Net, HTM5 etc…
2) Is the content already discussed in similar previous works or has been extended. Example [37]-[38]
3) Is proposed technique empirically evaluated?

After running the second stage only 22 articles become candidate for this study. The selection strategy was more exclusive than inclusive. Only good papers are selected, comparing and surveys papers are discarded. Selected articles were again assessed and evaluated by an external experienced researcher in the testing field. Table 3 in appendix shows all qualitative results and the inclusion/exclusion causes of all retained papers. Fig. 1 shows the articles selection procedure used in this study.

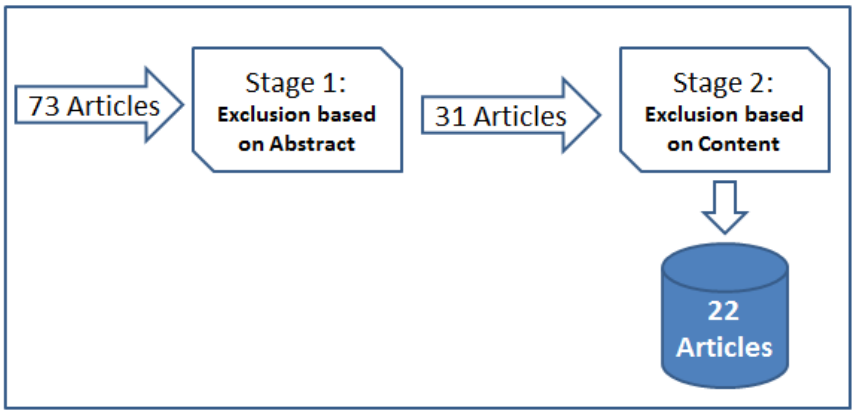

Fig. 1. Selection procedure.

Using the selection procedure described in Fig. 1, only 22 articles are selected. Some surveys and reviews are used as reference point for this study. The acceptance rate in our SLR is about 22%.

4. Regression Testing Approaches and Evaluation

One of the main sources of computational cost in regression testing is the running cost of executing large test set. In regression testing for Web application we have mainly two attitudes: reduction of generated test cases, and reduction of the execution cost. In response to our first there focused research questions (FQ1, FQ2, FQ3, and FQ4) we present respectively our finding in three subsections: Approach evaluation, empirical study evaluation, and tool evaluation.

4.1. Approach Evaluation

Authors in [39] discussed an automatic session data repairing algorithm for RT Web applications. A white box examination map is used to detect any change. A case study was applied to ten different versions of an online book store website. When modifications were made i.e. new hyperlink or dynamic pages are added or removed, some of the session data becomes invisible. Thus, a new automated session data repair algorithm which constructs a new version of session near to its original version to uncover missing paths and parameters is created. The repair algorithm is as follow:

1) **Edge Deletion:** If an edge (link) between two pages is removed, but the nodes (pages) remain, therefore there is a need to select new path navigation between both nodes.
2) **Node Deletion:** In case of node (page) deletion, links become invalid. In this situation, there is a need to found a new navigation from node's predecessor to its successor.
3) **Default:** Neither of the above cases applies: If both edges and nodes are deleted, in this situation a "damage limitation" approach is used when no path in the new structure can be found. This approach split old session into two sub sessions to which the repair algorithm is recursively applied.

Garg *et al*. [40] presented a regression testing technique called two-level prioritization approach using FDG (functionality dependency graph) and IFD (inter-procedural control graph). The proposed approach detects automatically modifications completed in system functionalities and prioritizes test cases needed.

To effectively minimize the number of test cases an automated approach is used to detect modification in functionalities and prioritizes test cases for them. Two priorities scheme are implemented FDG: Priority of test cases and Priority of test cases for Interference Constrains Graph (ICG). A Unified Modeling Language (UML) modeling was used to extract specifications of functions. A similar technique was proposed in [41] by imposing parallel execution during the prioritization process. Also Harrold *et al.* [42] used the prioritization technique to construct the FDG by identifying the functionalities in a Web application from the UML diagram of that Web application. Authors in [43] proposed a regression testing (RT) techniques based on event dependency graphs (EDG) and event test trees (ETT). Using the above techniques we can easily reduce the number of test cases and tackle the dynamic issue of Web application. EDGs are structured on cyclic dependency and to remove redundancy they introduced new step after making EDG which is ETT.

Roest *et al.* [44] and [31] suggested a common way to provide correctness for Ajax Web based application. The proposed approach automatically infers a model of the Web application and used as an input for the test suite. To overcome the error-prone behavior of Ajax technology like asynchronous nature and extensive use of Document Object Model (DOM) an oracle comparator is used.

A regression testing approach based on test selection prioritization was proposed in [45], [46]. Athira *et al*. [45] presented an UML model based on test prioritization. An activity diagram was used to model the system and capture coverage information. The priority rule is based on covering most important paths to reduce the test cases. The model executed the complete test suite and collect information related to the changes. The collected information is then used for test suite prioritization. Two models for RT are presented using activity diagrams and activity paths. In [46] authors introduced the prioritization rule of test cases using dependency based analysis approach. Initially they analyze the dependency relationship with the help of control and data flow information in WS-BPEL. Then a weighted graph is constructed to analyze the impact of modified objects. Finally test cases with high coverage are prioritized for modified-affected objects with highest weights are selected. Besides reducing test cases, the approach can eliminate fake dependencies in Business Process Execution Language (BPEL) process based on Business Process Flow Graph (BPFG). A prediction RT approach was proposed in [47], to detect specific test cases that need human inspection and attention. A model is used to harness the inherent similarities between initial and changed Web-based application in order to reduce the execution cost of regression testing. An automated tool called oracle comparator that relies on the semantics of HTML (or XML) output can decide if a pair of test case outputs indicates an error by comparing both files. Proposed method does not require manual seeding of faults and uses training data from other applications to automate this process. Lei *et al.* [19] suggested Web applications modeling using System Dependent Graph (SDG) and then introduced a slicing regression testing technique that emphasize on the indirect-dependent way. As a result, the usability of SDG will increase the workload and cost of the testing process, however, slicing technique offer to the user more effort on contents simplification, and improves the work efficacy. Authors in [48] proposed a systematic regression testing platform. It is designed to execute safe end-to-end regression test selection RTS for both intra- and inter-enterprise Web services using control-flow graphs (CFGs). The approach used an algorithm developed by Rothermel and Harrold for monolithic applications based on control flow graphs (CFG) [49]. The platform uses three phases: (1) create two CFGs for both old and new web application; (2) detect dangerous edges by comparing both CFGs; (3) and finally identify test cases that need to be rerun. For security reason a hash code technique is used to hide source code from testers. Regression testing is integrated into effective change management system to reduce test case execution time. Ruth *et al*. [50] developed a similar safe testing framework. The only reported enhancement rely on privacy-preserving techniques to enable multiple parties to complete a common task without exposing more than necessary,

and at the same time provide a secure model to protect sensitive information contained in both CFGs. The main contribution of this work is to protect sensitive information contained in CFGs, without reducing the effectiveness of the used regression testing selection (RTS) technique. A Web Service Regression Testing Model (WSRTM) framework is described in [51] for semantic Web service and test case generation. A model-based on retest process for end user is used to supports engineers in identifying WSDL-based changes.

Edwards *et al*. [52] described a faster and easier approach to create scripts for functional testing and load testing for Web application developed with AJAX Technology. The basic idea is to shorten the testing time by integrating the process of recording and creating scripts. Details of user's activities in the Graphical User Interface GUI are extracted based on reading Model signals document object (DOM) specific to the browser. Authors in [18] suggested a regression testing approach based on internal information in order to form Time Labeled Transition Systems (TLTS) and Task Precedence Graphs (TPG) for each selected service. The idea was initially proposed in [53] and then extended in [18]. Sensitive information was left unprotected which implies that the technique suffers from security issue.

Authors in [11] suggested a novel testing methodology. The regression testing method is based on the enhanced change reported to the component version in order to test the software system containing some modified components. It is a collaborative process, between component developer and system tester. Kinga *et al*. [54] proposed a regression testing approach using define invariant expressions of expected result over program variables and assert their correctness at run-time. The regression testing approach is based on dynamic analysis of JAVA SCRIPT code to infer invariants from a given Web application. According to their result analysis, the approach achieved a reduction cost execution time. More details can be found in the PhD dissertation [33]. Florian Haftmann introduced in his master work [32] an easy-to-use approach for regression testing of Web interfaces using an iterative approach. It allows interactive use and does a sophisticated reporting on changes. The applications are highly customizable and personalizable, so a lot of states which directly influence the applications' behaviour need to be tested to ensure consistency. Gagandeep *et al*. [55] suggested a model based approach for regression testing of Web applications. The proposed approach consists of four steps: 1. Domain Analysis and Modelling; 2. Model traversal and test case generation; 3.Optimizing test cases using coverage criteria; 4. Regression test suite generation. Graphical Web Model of the component is constructed. The generation of regression test-suite is optimized using "all-path" coverage criteria in order to reduce the effort and cost of rigorous cycle of software development and testing process. Also, Reiko *et al*. [56] presented a similar model-based approach for regression testing for Web services applications, where service interfaces are described using three layer: implementation, interface model, and observable behavior. The approach consider the impact of evolution for selective retesting but we also propose a coverage analysis mechanism to see if there is any requirement of new test cases to retain coverage as well.

Authors in [57] have introduced an automated tool to locate changes in PHP Web application. The idea is based on WebCrawler to shorten the path between users and their destinations. More detail about the tool is described in section C. Hussein *et al*. [38] presented an approach to detect areas affected by code changes using impact analysis. A program slice is used to generate new test cases for the crashed areas using program slices and consider both numeric and string input values. To do so, the researchers implemented a PHP analysis and regression testing engine called (PARTE [58]). The focus is on Web applications written in PHP. The approach is effective in reducing the amount of time needed to apply regression testing for frequently patched Web applications. Due to the compatibility constrain related to the web programming languages support, the proposed approach is not recommended. Authors in [36] suggested a meta-modeling approach using UML. A high level representation of the Web application is modeled using

Unified Modeling Language (UML [25]) for the assessment of static Web structure. Test cases are generated based on the computation of the path expression [59]. The main contribution of this paper is in the definition of a novel UML model used for analysis and testing techniques applied to Web applications. However, this process requires many manual tasks. An enhancement must be reported as a future work.

In response to our research focused question number 4, our study results shows that the most common best practice of regression testing in Web application is graph theory including CFG(Control flow graphs), ETT(Event test tree), and FDG (functional dependency graphs). Fig. 2 shows percentages of used approaches in RT for Web applications.

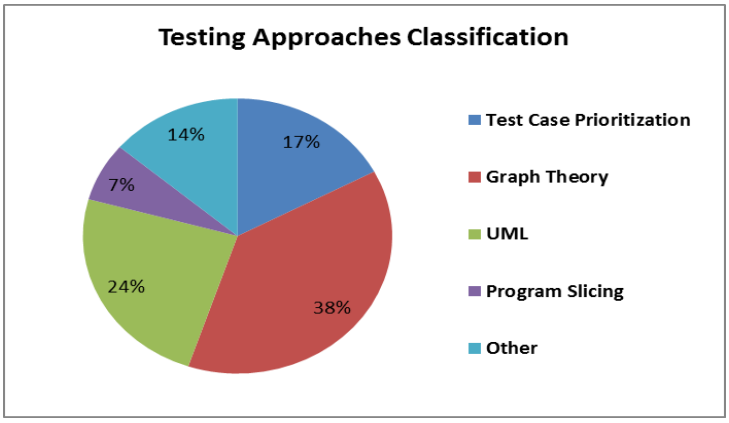

Fig. 2. Percentage of regression testing approaches applied in web applications.

As you can see the Graph theory has the heist percentage followed by test prioritization. Test case prioritization technique is the second widely technique used in regression testing for Web application. It is important to mention that some approaches [40], [41] used combined techniques such as UML and graph theory in their proposed solution. Program slicing is very important tool for incremental regression testing problem However, it is not appropriate for testing the initial copy of the Web application, because of the block redundancy. Fig. 3 shows the number of publications using different kind of regression test selection techniques for web applications.

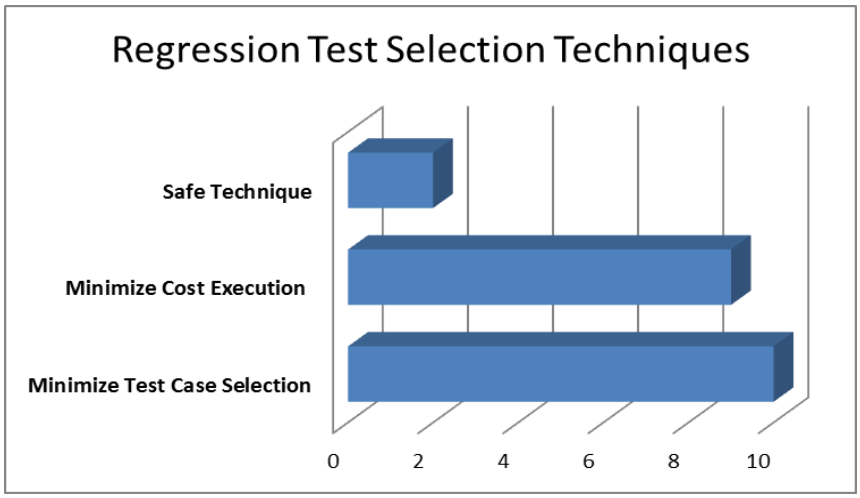

Fig. 3. Number of publication using reduction technique for web applications.

Our results highlight several differences between various regression testing techniques, and explore essential trade-offs that should be taken into consideration when choosing a technique for web application.

Two main minimization techniques are used in the literature: cost execution minimization and test case minimization. Authors in [55] suggested a combined approach where both cost and test selections are applied. Most regression test selection minimization techniques are not designed to be safe. No safe techniques can fail to select appropriate test case in the modified web application. Safe technique is an alternative solution. A set of safety conditions must be satisfied during the test case selection process. Recently, several safe regression test techniques have been presented in [18], [48].

4.2. Empirical Results Evaluation

In response to our second focused research question, we investigate and analyze empirical and case studies used in each selected publication. We found in our proposed study that nearly all papers used the number of generated test cases as a performance metric for RT in Web applications. We classify proposed approach based on the size of experiments and the number of case study. Regarding the size we classify them as follow: small, medium, and large, depending on the number of case study used, the number of test cases generated, and the complexity of the Web application. A small study has less than 50 test cases using simple Web application. A medium study has between 50 and 100 test cases. Finally large study has more than 100 test cases generated and executed in their experimental section in complex application.

In [39] an online book store application is used to test the tool performance. A series of ten modification where made to the book store application to implement new functionalities. Small size experiment, only 22 books are stored in the database. A case study is described to analyze the impact of the proposed tool in terms of effort and rate of automation in the development of a new Web application project is suggested in [1]. The experience generated 147 test cases on a help desk application Webpage and result showed that their techniques decreased size of test cases to 44-90%. Authors in [44] used total of four case studies. Two case studies were conducted one on open source contract management system and the rest is applied on a highly dynamic Google Reader application. Their approach includes analyzing the DOM trees of both original and modified versions. An air ticket reservation Web application is used in [45]. Kumar *et al.* [43] performed their case study on SWLS (Simple Web Student Login System) and found that test cases were reduced by 41.4%. In [40] an experimental evaluation was proposed for efficiency study. Authors randomly seeded 20 faults in various modified functionalities of the Online Bookstore application. Three faults categories are injected in application: Logical Faults, Form Faults and Appearance Faults. 130 test cases were e generated from UML Activity diagrams of the Online Bookstore application.

Authors in [47] experimentally tested their hypothesis that Web site similarities are very helpful to aid in the automation of various aspects of testing Web-based applications. A predictive model power at detecting faults between test case and output pairs is used. Optimal results are obtained for two Web applications. An empirical analysis for applying safe regression to Web services is proposed in [48]. Five Web services systems selected from different domains. All selected systems were developed in Java environment using the Apache Axis toolkit. A total of 461 test cases were generated. In [46] an ATM cases study was proposed to investigate the applicability of our method. A rich series of experiments were performed in [50] to evaluate the execution cost impact of the approach. In [51] authors illustrated an End User Regression Testing method for a simple Web service Bank Account. Authors in [11] presented preliminary experiments to validate the effectiveness of their proposed testing strategy. Medium scale systems are used. Experiment results show that, the regression testing method is fairly feasible and cost-effective in practice. Tarhini *et al*. [18] considered a real help desk Web application to evaluate a regression safe testing selection technique. The application is implemented using .Net 2005 environment. Empirical section shows promoting results, the test set size was reduced for about 44-90%. Authors in [54] presented a case study conducted on nine open source Web applications to evaluate the proposed approach. The results show that the approach is able to effectively generate stable assertions and detect JAVA SCRIPT regression faults with

a high degree of accuracy and minimal performance overhead. Very elementary experience is presented in [31] to test something simple (e. g. adding an entry to a low-level table). Used tool is still in an early development stage.

A very simple Sign Up and Login System (WSL) application was proposed in [55] to demonstrate the functionality of approach. No test case generation is specified. A small but real application is used to assess the correctness of proposed approach presented in [56]. The selected application is derived from an open source desktop application called BTsys is selected. The evaluation is based on an automatic bug tracking system implemented in C# to report failure data. Authors in [57] have proposed an automated tool to locate the changes in the Web application which. Therefore, regression testing effeteness is enhanced. Hyunsook *et al*. [38] designed a controlled experiment using five open source Web applications that have multiple versions, various sizes, and different characteristics to assess their approach. For evaluation purpose authors used two open sources Web applications written in PHP. Applications were downloaded from SourceForge. Different versions of osCommerce [60] and FAQForge [61] are used. This study was performed using a virtual machine on multiple hosts. In [36] Over 15 Websites were periodically downloaded and analyzed using the tools ReWeb and TeslWeb.

The qualitative assessments of empirical results were summarized in Table 2. The results were divided into six different categories according to the test set size and Web application complexity used during the testing process.

Table 2. Classification Scheme of Empirical Studies

Small	**Simple Web Application**	[31], [40], [43], [55]
	Complex Web Application	[51], [56]
Medium	**Simple Web Application**	[11], [45], [47], [50]
	Complex Web Application	[11], [18]
Large	**Simple Web Application**	[13], [39], [44], [48], [54]
	Complex Web Application	[36], [38]

Some works are not discussed in this section because of information shortage in their publications. Empirical study judgment cannot be conducted about their works.

4.3. Tools Evaluation

Tools are important for regression testing. They enable the transformation of the process from theoretical to practical area. In response to our focused research question 3, we classify available tools based on their: Usability, Scalability, and Compatibility with Web application programming languages, and operative type (manual or automatic). It important to mention that some tools cannot be evaluated due to lack of information missing in their presented work.

Table 3 summarizes tools evaluation used in RT for Web application. There have been 16 tools implemented. We can realize that tool development has received an increasing attention from research community since year 2010.

Authors in [44] and [31] presented an open source testing tool called CRAWLJAX to handle the dynamic aspect of AJAX applications. SWT-based GUI is integrated to analyze the difference between DOM trees of both original and modified versions. In [40] a C# application is described to implement the proposed prioritization approach and generated test cases. Test cases are then executed using Selenium test tool [13]. A semi-automatic too is suggested in [47]. A comparator judgment about XML/ HTML test case should be made by checking the tree-structure features, rather than using textual difference.

A comparison is made between both output documents to find any alignment between them. During the test suite execution, user interventions required at several points. Authors [62] and [48] performed RT testing using an open source tool Apache Axis for Java applications to simulate the functionality and performance in an end-to-end manner. An approach calling it Code transformation by Java Interclass Graph (JIG) to analyze the static and dynamic behavior and reduce the test cases used. The limitation of the proposed tool is that application should be deployed in Axis server. Ruth *et al.* [50] designed a manual tool to secure the information saved in each node. The proposed testing framework implements three components, namely CFGs, test cases, and coverage information. Qing *et al.* [51] developed a manual tool based on WSDL (Web service regression testing model interface) for semantic Web service. The Inputs, Outputs, Preconditions, and Effects (IOPEs) paradigm are used to define Web service semantics. Usually in Web service users are interested in the change of their WSDL and IOPEs and impacts on other components in the system, and Web service unit/system test suits. eValid [52] is a new testing tool working on events from the interaction of the end user with the Web application. eValid addresses both simple applications and modern Web applications based on JavaScript. The scripting engine is fully incorporated into the browser and behaves like a real client browser. eValid require user intervention for script editing. Authors in [11] proposed a novel testing component-based approach that uses a collaboration channel between the component provider and system tester. Developers detect the changed information from the procedure call graph and deliver it to component users through XML files. Abbas *et al.* [18] described regression testing tool to view Web application as event driven environment. The interaction is based on following: data dependence, control dependence and call dependence. The proposed tool uses two event dependency graphs. The first one is used to represent the original system. The second one and is used to represent the modified system. Authors in [54] proposed an automated technique for JAVA SCRIPT regression testing, which is based on dynamic analysis to infer invariant assertions. JSART extends and builds on top of InvarScope [63] tool. For JAVA SCRIPT code interception, we use an enhanced version of Web-Scarab's proxy [64] to automatically analyze and modify the content of HTTP responses before they reach the browser. A tool called HTTrace is implemented in [32] to record user actions within a Web browser, save it persistent to a test run file. User must replay an arbitrary test run file as it was recorded on the Web browser, then compare and report any difference between the browser pages as they were at record time and as they are at replay time. The tool is still in an early development stage. The COM technology used in HTTrace sometimes does not scale well, e. g. after playing many traces in sequence. Gagandeep *et al.* [55] proposed a tool implemented in Visual C++ to analyze and model the application, and then generate test sequences based on "all path" coverage criteria for optimization. The model is implemented as a queue data structure. The algorithm complexity needs improvement for performance evaluation. Authors in [56] evaluated their proposed approach using a bug tracker service tool by writing a client in Java using the integrated development environment (IDE). The tool is semi-automated since user is involved to generate manually the test set. Test set completeness is evaluated trough fault seeding. Scalability is not supported; also no real projects are used to evaluate the implemented tool. Shikha *et al.* [57] have introduced an automated tool for locating the changes in the Web application to increase the effectiveness of regression testing. The basic idea is based on WebCrawler to shorten the path between users and their destinations. The tool implement, 1) a Web crawler to crawls the Web application, 2) a tool for constructing the HTML DOM tree representation for a Web page, 3) a comparator that compares the new DOM tree with a previous version of the DOM tree stored in our system. Redundant Web blocks from more than two Web pages and tree storage damage the approach scalability. A Testing tool that combines PHP Analysis and Regression Testing Engine (PARTE) is implemented in [38]. The approach focuse on PHP Web application, but can easily be applied to other languages by extending the front end file conversion

functionality. The current path generator implementation does not provide Web elements, so they had to be provided manually. Therefore, not all test cases can be run. Ricca *et al.* [36] presented automatic support techniques for analysis and testing of Web applications. Two tools ReWeb and TestWeb where developed. ReWeb downloads and analyzes the pages of a Web application with the purpose of building a UML model of it, in accordance with the metamodel using spidering approach, similar to the one used in [39]. ReWeb's helps to understand the site organization, using navigation paths (history view) and of variable usage (data flow view). TestWeb generate and execute a set of test cases for a Web application whose model was computed by ReWeb. Contrary to [39], the whole process in [36] is semi-automatic, with user interventions required at several points. Authors in [41] implement a C# module to evaluate the proposed approach. Selenium test tool is used for automatic test suite execution. In [45], [46] authors described a manual approach without mentioning any tools. Researches discussed in [65], [66] a similar tool presented in [13].

Fig. 3 shows the number of tools addressing each Web application programming language. Many articles did not specify explicitly what Web programming languages can be supported by their tools example [13], [40], [57]. Several other tools are discussed in [67]-[69] to support mobile application and database applications.

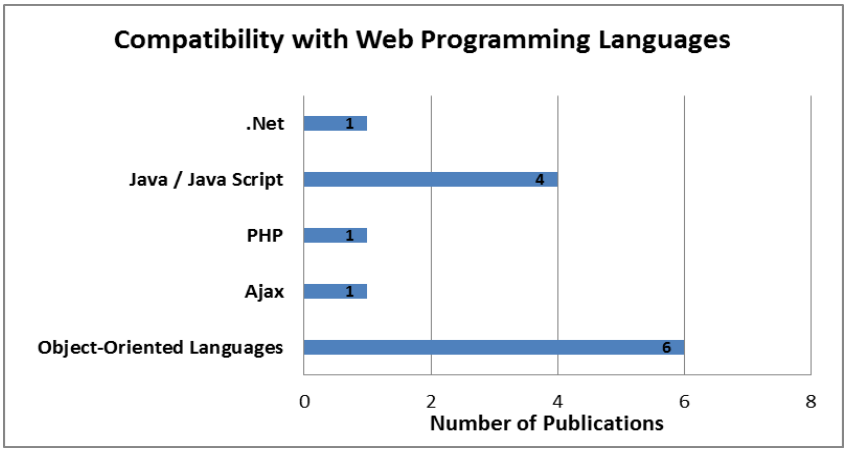

Fig. 3. Compatibility with web programming languages.

Compatibility characteristic is relatively high compared to other of studies. Scalability aspect was not strongly supported by previous work. This is mainly because of HTML DOM tree problem, COM, and path selection approaches used in regression testing. Number of paths increases exponential when Web application contains many pages, similarly redundant blocks increase when HTML DOM tree is used.

5. Discussion

Our evaluation results strongly support the conclusion that the scalability is a concern in regression testing for Web application. As we observe from tool evaluation, technologies used in tool implementation dos not support scalability. Due to rapid and frequent changes in Web application it is nearly impossible to cover all modified paths and nodes. Large and complex Web applications require time and efforts to reveal invisible paths. A challenge was presented in [39] to discover invisible path based on session data. Automated RT techniques are useful and timeless, but on the other hand any error in test cases generation could result in missing some paths due to dynamism. When a Web application contains many pages, more paths are added in HTML DOM tree generation which involved redundant blocks. Also, correcting errors in HTML can be hard. Similarly, when graph theory is used the number of nodes and edge tend to be increased. Another limitation involves test case generation. Many approaches [50]-[56] require manual action for test

set selection. As a result, not all faults were able to be detected during the regression testing. Again scalability is recurred.

Regression testing for Web application is growing rapidly. Many efforts have been made to extend tool compatibility for testing new Web programming languages. A meta-model technique was discussed for web application in [70]. Developed techniques are not costly compared to other testing techniques such as mutation testing and system testing. Based on existing achievement we can expect that the future will bring more paradigms to develop more scalable approaches without over compromising on quality and cost. More efforts must be placed on safe selection to guarantee the selected subset contains all test cases that can reveal faults modified version. Security requirement must be investigated also during the regression testing.

Table 3. Summary of Published Regression Tools for Web Applications

Tool	**Articles**	**Automatic / Manual**	**Compatibility with Web Application Language**
JSpider	[36], [39]	Academic environment /Semi-automated	Supporting many languages
UML and Selenium	[13], [40], [41]	Semi-Automated	Not mentioned
CRAWLJAX	[44]	Open Source / automated tool	Ajax
Apache Axis	[48], [62]	Open Source / Automated tool	Java/ JavaScript
RTS	[50]	Manual tool	Not mentioned
WSDL	[51]	Manual tool	Not mentioned
eValid	[52]	Commercial Semi-automated tool	JavaScript
Component-base	[11]	NA	procedural or Object-Oriented application
Event dependency graphs (EDG)	[18]	NA	.Net
JSART	[54]	Academic environment/ Automated tool	Java Script
HTTrace	[32]	Automated tool	Supporting many languages
Graphical Web Model (GWM)	[55]	Automated tool	Supporting many languages
bug tracker service	[56]	Semi-automated tool	Java
Web Crawler	[57]	Automated tool	Not mentioned
PHP analysis and PARTE	[38]	Automated tool	PHP
ReWeb and TeslWeb	[36]	Semi-automated tool	Supporting many languages

6. Conclusion and Future Work

The paper has provided a detailed systematic review and analysis of trends on regression testing for Web applications. The paper covers regression test selection methods, empirical studies, and regression testing tools. From the data we collected, there has been much effort on cost and test case selection minimization techniques. Our findings reveal that there is an increasing interest to regression testing for Web application since, it plays crucial role in maintaining Web applications after modifications and changes are made. Recent trends include new open source and industrial tools, which indicate the maturity level of this research filed. We also find some evidence that there are an increasing number of new approaches for security regression testing in Web application.

Recent work tends to focus on test set minimization rather than cost execution reduction. The desired decision has risen because of the effect of larger test set requires extra cost execution. Our study found that

only 20% of articles in the selected research pool did not discuss and present testing tool. We have identified some basic problems in regression testing for Web applications domain. Proposed tools are not evaluated sufficiently. In many studies, only one simple Web application is evaluated. Furthermore, original source code is edited for modest testing. Only few studies draw powerful conclusion based on their rich empirical studies.

Future works for the research community are scalability and safety selection issues. Security requirement must be explored in Web application when component base model and Graph theory are applied. Encourage compatibility to cover wide programming languages for future developed tools.

Appendix

Summary of Published Papers about Regression Testing for Web Applications

Ref	Title	Year/ Source	Size of the Empirical Study	Used Approach / Tool	nclusion/Exclusion
[39]	Automated Session Data Repair for Web Application Regression Testing	2008 IEEE	Regression Testing, Web Applications.	Automate tool Jspider using edge and node analysis	inclusion
[13]	Extension of Selenium RC tool to perform automated testing with databases in web applications Automation of Software Test (AST)	2013 IEEE	Selenium RC, Software Testing, Functional Testing, Test Automation, Database Testing.	extension of a very popular functional testing tool for user interface (Selenium RC) to support database testing in web based applications.	Exclusion approach focus on database testing in web application
[44]	Regression Testing Ajax Applications: Coping with Dynamism	2010 IEEE	Regression testing; Ajax; web applications	Crawljax DOM tree using inferred state flow graph. Automated tool.	included
[45]	Web Services Regression Test Case Prioritization	2010 IEEE	Web services, Test Prioritization, Activity Diagram, Activity paths.	Manual approach using activity and activity paths	inclusion
[43]	Event Driven test case selection for Regression Testing Web Applications	2012 IEEE	Regression Testing, Web Application, Event Dependency Graph, Event Test Tree	Event driven Technique Manual approach	inclusion
[40]	A Two-Level Prioritization Approach for Regression Testing of Web Applications	2012 IEEE	Regression testing, Test case prioritization, Web applications, Two level prioritization	Automatic approach using two level prioritization	inclusion
[47]	Harnessing Web-based Application Similarities to Aid in Regression Testing	2009 IEEE	Regression testing; automated solutions; number of false positives	automated oracle comparator that relies on the semantics of HTML (or XML)	Inclusion
[37]	An Effective Regression Testing Approach for PHP Web Applications	2012 IEEE	Regression testing, impact analysis, test case generation, PHP web applications	Automated approach using PHP Analysis and Regression Testing Engine (PARTE)	Exclusion similar approach proposed in [38]
[19]	Regression Testing for Web Applications Based on Slicing	2003 IEE	Web application; Regression testing; slicing method	Manual slicing approach that emphasized on the indirect-dependent among data	Inclusion
[53]	Regression Testing Web Services-based Applications	2006 IEEE	Regression testing, event dependency graph	Manual approach based on TLTS (Timed labeled transition system) for WSDL(Web services description language)	Exclusion topic covered and extended in [18]
[62]	Applying Safe Regression Test Selection Techniques to Java Web Services	2006 IEEE	Web application; Regression testing; Regression test selection (RTS)	Automated code transformation approach using Regression test selection (RTS)	Exclusion topic covered in [65]
[65]	A Safe Regression Test Selection Technique for Web Services	2007 IEEE	Web application; Regression testing; Regression test selection	Automated approach based on Regression test selection (RTS)	Exclusion topic covered in [48]

			(RTS); Control-Flow Graphs		
[48]	Towards Automatic Regression Test Selection for Web Services	2007 IEEE	Web application; Regression testing; Regression test selection (RTS); Control-Flow Graphs	Automatic framework to synchronize the RTS processes	inclusion
[46]	Test Case Prioritization for Web Service Regression Testing	2010 IEEE	Web service; regression testing; test case prioritization; impact analysis; dependence analysis	Manual approach using flow graph	Inclusion
[35]	Challenges in Audit Testing of Web Services	2011 IEEE	Web service Composition, Audit Testing, Regression Testing.	Automated tool based on change detection.	Excluded Topic covering general RT techniques. Out of scope (web application)
[50]	Employing Privacy-Preserving Techniques to Protect Control-Flow Graphs in a Decentralized, End-to-End Regression Test Selection Framework for Web Services	2011 IEEE	Test Selection; Regression Testing; Web Services; End-to-End; Decentralized; Privacy-preserving	Manual approach to share their control flow graph CFG using privacy-preserving techniques to protect sensitive information	inclusion
[51]	An Approach of End User Regression Testing for Semantic Web Services	2011 IEEE	Semantic Web Services; Regression Testing; WSDL;	Manual approach using WSRTM (web service regression testing model) for semantic web service	Inclusion
[42]	Prioritizing test cases for regression testing	2001 IEEE ACM	Test Cases, Regression testing,	Average percentage faults detected measure (APFD) using SAS statistical package and Bonferroni. Automatic Tool	Exclusion: general approach dos not specifically tackle web applications.
[52]	Le Test Automatise des Applications Web Modernes	2011	AJAX, Web 2.0, regression testing, automatic testing	eValid [71]: recording mechanism based on a clone browser Internet Explorer. Semi-automatic tool	Inclusion
[6]	Program slicing-based regression testing techniques. Software Testing, Verification and Reliability	1996 Wiley	Regression testing, slicing technique, data flow	Slicing approach. No tool mentioned	Exclusion: slicing approach used in [19].
[11]	Regression testing for component-based software systems by enhancing change information	2005 IEEE	Regression testing, component metacontents, component-based applications	Metacontent aware tool for code-based and specification-based.	inclusion
[18]	Regression testing web applications	2008 IEEE	Regression testing, event dependency graph, web application	Select test cases based on the event based dependency graph (EDG). Mentioning a tool without any detailed information	inclusion
[54]	JSART: JavaScript Assertion-Based Regression Testing	Springer	Regression Testing; invariant assertions; invariant assertions. JavaScript, dynamic analysis	Based on on-the-fly JAVA SCRIPT source code instrumentation and dynamic analysis to infer invariant assertions. Automated tool : JSART	inclusion
[31]	Master Thesis: Automated Regression Testing of Ajax Web Applications	2010	Regression testing; Ajax web application; Test failure Visualization.	Automated tool called CRAWLJAX for handling the dynamic aspect of AJAX applications using Oracle Comparator Pipelining (OCP)	Exclusion, approach discussed in [44]
[32]	Master thesis: Regression Testing on Web-based Information Systems		Regression testing; Web based	Automated tool called HTTrace that uses COM interface to Microsoft's Internet Explorer.	Inclusion
[33]	PhD thesis: An Exploration of User-Visible Errors in	2010	Testing, error detection; user-visible; web	Automated oracle comparator approach towards comparing	Exclusion similar approach in

	Web-based Applications to Improve Web-based Applications		application	test case outputs in web-based applications.	described in [54]
[55]	Automatic Generation of Regression Test Cases for Web Components using Domain Analysis and Modeling	2010	Regression testing, Web Components; Domain Analysis; Test Automation	Analysis based modeling approach. An automated Graphical Web Model tool implemented in visual C++.	inclusion
[56]	On Model-Based Regression Testing of Web-Services Using Dependency Analysis of Visual Contracts	2011 Springer	Regression testing; web services; Model based approach	Model-based approach to abstract the detail of programming language. Semi-automated tool called bug tracking service.	inclusion
[57]	An automated tool for regression testing in web applications	2013 ACM	Regression testing; DOM tree generator; code change.	An automated tool composed from web crawler; HTML DOM tree generator and a comparator	inclusion
[38]	An efficient regression testing approach for PHP web applications: a controlled experiment	2014 Wiley	Regression testing; impact analysis; test case generation; PHP web applications; empirical studies	Test case generation approach by using program slices considering both string and numeric input values An automated tool: Hypertext Preprocessor (PHP) Analysis and Regression Testing Engine (PARTE).	inclusion
[34]	PhD thesis: MODEL-BASED TESTING USING VISUAL CONTRACTS	2001 IEEE	Web applications, testing, UML modeling	A semi-automated tool : ReWeb and TestWeb	Exclusion: Similar discussed in [56]
[36]	Analysis and Testing of Web Applications	2001 IEEE	Web applications, testing, UML modeling, reverse engineering	An UML model of Web applications is proposed to exploit a white box testing criteria. A semi-automated tool : ReWeb and TestWeb	Inclusion
[41]	Parallel Execution of Prioritized Test Cases for Regression Testing of Web Applications	2013 IEEE	Regression testing, Test case prioritization, Web applications, Parallel execution	Approach based on partitioning test suite into test sets according to the functionalities and associate the test sets with each module of the FDG.	Inclusion
[25]	A methodology for retesting modified software	1981 ACM	Minimization-based regression test, linear equations	minimization techniques, to select minimal sets in regression technique	Exclusion : Approach applicable to static web application

Acknowledgment

The author would like to thank Dr. Izzat Alsmadi for his assistance in collecting research papers and valuable comments.

References

[1] Chittimalli, P. K., & Harrold, M. (2009). Recomputing coverage information to assist regression testing. *IEEE Transactions on Software Engineering*, *35(4)*, 452–469.

[2] Engstr¨om, E., Runeson, P., & Skoglund, M. (2010). A systematic review on regression test selection techniques. *Information and Software Technology*, *52(1)*, 14–30.

[3] Hartmann, J., & Robson, D. (1988). Approaches to regression testing. *Proceedings of the Conference on Software Maintenance* (pp. 368-372).

[4] Yoo, S., & Harman, M. (2010). Regression testing minimization, selection and prioritization: A survey.

Software Testing, Verification and Reliability, 145-158.

[5] Ostrand, T., & Weyuker, E. (1988). Using data flow analysis for regression testing. *Proceedings of the Conference on Annual Pacific Northwest Software Quality* (pp. 142-148).

[6] Gupta, R., Harrold, M., & Soffa M. (2006). Program slicing-based regression testing techniques. *Software Testing, Verification and Reliability*, *6(2)*, 83-111.

[7] Harrold, M., & Souffa, L. (1988). An incremental approach to unit testing during maintenance. *Proceedings of the Conference on Software Maintenance* (pp. 362-369)

[8] Hartmann, J., & Robson, D. (2014). Techniques for selective revalidation. *IEEE Software*, *7(1)*, 31-37.

[9] Pei, H., Xiaolin , L., Kung, D., Chih-Tung, H., Liang, L., Chen, C., *et al.* (1997). A technique for the selective revalidation of OO software. *Journal of Software Maintenance: Research and Practice*, *9(4)*, 217-33.

[10] Rothermel, G., & Harrold, M. J. (1994). A framework for evaluating regression test Selection techniques. *International Conference on Software Engineering*.

[11] Mao, C., & Lu, Y. (2003). Regression testing for component-based software systems by enhancing change information. *Proceedings Asia-Pacific Software Engineering Conference* (pp. 256-301).

[12] Orso, A., Harrold, M., Rosenblum, D., Soffa, M., & H. Do. (2001). Using component metacontent to support the regression testing of component-based software. *Proceedings of the IEEE International Conference on Software Maintenance* (pp. 716-725).

[13] Castro, M. D., Macedo, G., Collins, E., & Neto, A. A. C. D. (2013). Extension of selenium RC tool to perform automated testing with databases in web applications. *USA Automation of Software Test (AST)*, 125–131.

[14] Willmor, D., & Embury, S. M. (2005). A safe regression test selection technique for database-driven applications. *Proceedings of the IEEE International Conference on Software Maintenance* (pp. 421-430).

[15] Scott, W., & Ambler, W. (2007). Test-driven development of relational databases. *IEEE Software*, *24(3)*, 37-43.

[16] Zhan, W. H., Chen, R., & Huang, S. (2010).GUI regression testing based on function-diagram. *Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent Systems* (pp. 559 – 563).

[17] Chen, J., Mengxiang, L., & Shao, B. (2012). When a GUI regression test failed, what should be blamed?. *Proceedings of the IEEE International Conference on Software Testing, Verification and Validation* (pp. 467-470).

[18] Tarhini, A., Ismail, Z., & Mansour, N. (2008). Regression testing web applications. *Proceedings of the International Conference on Advanced Computer Theory and Engineering* (pp. 902–906).

[19] Xu, L., Baowen, X., Zhenqiang, C., Jixiang, J., & Huowang, C. (2003). Regression testing for Web applications based on slicing. *Proceedings of the Annual International Conference on Computer Software and Applications* (pp. 652–656).

[20] Xu, L., Xu, B., & Chen, Z. (2003). Survey of web testing. *Computer Science*, *30(3)*, 100-104

[21] Kitchenham, B., Pearl, B. O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering. *Information and Software Technology*, *51(2)*, 7–15.

[22] Hetzel, W. (1973). *Program test Methods* (pp. 201-213). Prentice Hall,

[23] Hartmann, J. (2012). 30 years of regression testing: Past, present and future.

[24] Leung, H., & White, L. (1990). Insights into testing and regression testing global variables. *Journal of Software Maintenance*, 209-222.

[25] Fisher, K., Raiji, K,. & Chrusckicki, A. (1981). A methodology for retesting modified software. *Proceedings of the Conference of the National Teleconference* (pp. 1–6).

[26] Engström, E., Skoglund, M., & Runeson, P. (2008). Empirical evaluations of regression test selection

techniques: A systematic review. *ACM ESEM*, *30(12)*, 971-978.
[27] Giuseppe, A. A., Lucca, D., & Fasolino, A. (2005). *7 Web Application Testing*.
[28] Jazayeri, M. (2007). Some trends in web application development. *Future of Software Engineering*, 199-211.
[29] Postel, J. (1981). NCP/TCP Transition Plan. Retrieved September 27, 2014, from http://www.rfceditor.org/rfc/rfc801.txt
[30] Kleinrock, L. (2010). An early history of the internet. *IEEE Communications Magazine*, 26-36.
[31] Master Thesis. Retrieved September 25, 2014, from http://swerl.tudelft.nl/twiki/pub/Main/PastAndCurrentMScProjects/Thesis_Danny_Roest.pdf
[32] Master thesis: Retrieved September 27, 2014, from http://www4.in.tum.de/~haftmann/student/mthesis/thesis.pdf
[33] PhD Thesis: Retrieved October 09, 2014, from https://www.cs.virginia.edu/~weimer/students/kinga-phd.pdf
[34] Tamin, K. A. PhD Thesis. Retrieved October 13, 2014, from https://lra.le.ac.uk/bitstream/2381/27571/1/2012KhanTAPhd.pdf
[35] Nguyen, C., Marchetto, A. & Tonella, P. (2011). Challenges in audit testing of web services. *Proceedings of the Conference on Software Testing, Verification and Validation Workshops* (pp. 103-106).
[36] Ricca, F., & Tonella, P. (2001). Analysis and testing of Web applications. *Proceedings of the International Conference on Software Engineering*.
[37] Marback, A., Hyunsook, D., & Ehresmann. K. (2012). An effective regression testing approach for PHP web applications. *Proceedings of the International Conference on Software Testing, Verification and Validation.*
[38] Hyunsook, D., & Hossain, M. (2014). An efficient regression testing approach for PHP web applications: A controlled experiment software testing. *Verification and Reliability*, *24(5)*.
[39] Harman. M., & Alshahwan, N. (2008). Automated session data repair for web application regression testing. *Proceedings of the 1st International Conference on Software Testing, Verification, and Validation* (pp. 298-307).
[40] Garg. D., Datta, A., & French, T. (2012). A two-level prioritization approach for regression testing of web applications. *Proceedings of the Conference on Software Engineering* (pp. 150-153).
[41] Garg, D. & Datta, A. (2013). Parallel execution of prioritized test cases for regression testing of web applications. *Proceedings of the Conference on Thirty-Sixth Australasian Computer Science* (pp. 23-32)
[42] Rothermel, G., Untch, R., & Harrold. M. (2001). Prioritizing test cases for regression testing. *IEEE Transactions on Software Engineering*, *27(6)*, pp.929 –948
[43] Kumar, A., & Goel R. (2012). Event driven test case selection for regression testing web applications. *Proceedings of the International Conference on Advances in Engineering, Science and Management* (pp. 121-127)
[44] Roest, D., Mesbah, A., & Van, D. A. (2010). Regression testing ajax applications: Coping with dynamism. *Proceedings of the International Conference on Software Testing, Verification and Validation (ICST)* (pp. 127-136).
[45] Athira, B., & Samuel, P. (2010). Web services regression test case prioritization. *Proceedings of the International Conference on Computer Information Systems and Industrial Management Applications* (pp. 438-443).
[46] Chen, L. Z., Wang. L., Hongmin, S., & Baowen X. (2010). Test case prioritization for web service regression testing. *Proceedings of the Symposium on Service Oriented System Engineering*.
[47] Dobolyi, K., & Weimer, W. (2009). Harnessing web-based application similarities to aid in regression

testing. *Proceedings of the International Symposium on Software Reliability Engineering* (pp. 71-80).

[48] Ruth, M., Sehun, O., Loup, A., Horton, B., Gallet, O., & Shengru, T, (2007). Towards automatic regression test selection for web services. *Proceedings of the Conference on Computer Software and Applications* (pp. 729-736).

[49] Rothermel G., & Harrold, M. (1997). A safe, efficient regression test selection technique. *ACM Trans. Software Engineering Methodology, 6(2)*, 173-210.

[50] Ruth, M. (2011). Employing privacy-preserving techniques to protect control-flow graphs in a decentralized, end-to-end regression test selection framework for web services. *Proceedings of the International Conference on Software Testing, Verification and Validation Workshops (ICSTW)* (pp. 139-148).

[51] Zhang, T., Yao, Q., Zheng, X., Chunyan, M., & Wang, H. (2001). An approach of end user regression testing for semantic web services. *Proceedings of the International Conference on Management and Service Science* (pp. 1-4).

[52] Edward, F., & Miller, A. (2011). Le test automatise des applications web modernes. *Genie Logiel,* 6-12.

[53] Tarhini. A., Fouchal, H., & Mansour, N. (2006). Regression testing web services-based applications. *Proceedings of the International Conference on Computer Systems and Applications* (pp. 163-170).

[54] Mirshokraie, S., & Mesbah, A. Java script assertion-based regression testing web. *Engineering Lecture Notes in Computer Science* , *7387*, 238-252

[55] Gagandeep, A., & Sengupta, J. (2010). Automatic generation of regression test cases for web components using domain analysis and modeling. *International Journal of Computer Applications*, *12 (4)*, 14-17.

[56] Ahmed, K. T., & Heckel, R. (2011). On model-based regression testing of web-services using dependency analysis of visual contracts. *Fundamental Approaches to Software Engineering. Springer Lecture Notes in Computer Science, 6603*, 341-355.

[57] Raina, S., & Prakash, A. A. (2013). An automated tool for regression testing in web applications. *ACM SIGSOFT Software Engineering Notes archive*, *38 (4)*, 1-4.

[58] Marback, A., & Do, H. (2010). A regression testing engine for PHP web applications: PARTE (fast abstract). *Proceedings of the International Symposium on Software Reliability Engineering* (pp. 404–405).

[59] Beizer, B. (1990). *Software Testing Techniques* (pp. 193-212). International Thomson Computer Press.

[60] OSCommerce. Retrieved September 212, 2014, from http://www.oscommerce.com/

[61] FaqForge. Retrieved September 12, 2014, from http://sourceforge.net/projects/faqforge/

[62] Feng, L., Ruth, M., & Shengru T. (2006). Applying safe regression test selection techniques to java web services. *Proceedings of the International Conference on Next Generation Web Services Practices*..

[63] Groeneveld, F., Mesbah, A. & Van, D. A. (2010). *Automatic Invariant Detection in Dynamic Web Applications*. Technical Report TUD-SERG-2010-037, TUDelft.

[64] Bezemer, C., Mesbah, A., & Van, D. A. (2009). Automated security testing of web widget interactions. *Proceedings of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering*.

[65] Ruth, M., & Shengru T. (2007). A safe regression test selection technique for web services. *Proceedings of the International Conference on Internet and Web Applications and Services*.

[66] Chen, Y., Rosenblum D., & Vo, K. (1994). Testtube: A system for selective regression testing. *Proceedings of the Conference on Software Engineering* (pp. 163-171).

[67] Singh, Y., Arvinder, K., Bharti, S., & Singhal, S. (2012). Systematic literature review on regression test prioritization techniques. *In Informatica*.

[68] Pasala, A., & Bhowmick, A. (2001). An approach for test suite selection to validate applications on deployment of COTS upgrades. *Proceedings of the Conference on Asia-Pacific Software Engineering* (pp. 401-407).

[69] Szabo, C., Samuelis, L., Ivanovic, M., & T. Fesic. (2012). Database refactoring and regression testing of Android mobile applications. *Proceedings of the IEEE International Symposium on Intelligent Systems and Informatics* (pp. 135-139).

[70] Yanelis, H., Tariq, M., Jairo, P., & Peter, J. (2009). A meta-model to support regression testing of web applications. *Proceedings of the Conference on Annual Southeast Regional* (pp. 123-129).

[71] EValid. Retrieved September 12, 2014, from http://www.e-valid.com

Anis Zarrad was born in Ksar Hellal, Tunisia, in 1961. He received his B.E degree in computer science and software engineering from University of Ottawa, Ottawa, Canada, and the master's degree in computer science from Concordia University, Montreal, Canada, and the Ph.D. degree in computer science from University of Ottawa, Ottawa, Canada. He is currently an assistant professor in Prince Sultan University, Riyadh, Saudi Arabia. His research interests include wireless network protocol testing based on state transition testing, and mobile collaborative virtual environment.