Chapter 5

Regression Testing of Web-Components

With emergence of services and information over the internet and intranet, Web sites have become complex. Web components and their underlying parts are evolving rapidly and are becoming a basic part of business quality of service. As web services play an important role in providing different functionality, correctness and reliability of web components are often crucial to the success of organizations. Testing web component is an effective technique to ensure the quality of Web applications. Testing Web components raises new challenges as the diversity and complexity of web systems derived the need for test automation. Test Automation is use of software for automatic execution of test cases and their comparison with expected outcome.

In this chapter, analysis based modeling and regression testing technique for web applications has been presented. Web application is identified with the set of pages that can be accessed from a given Web server. Model based testing is a test automation approach that generates and maintains useful and flexible tests from explicit descriptions of the application. Models that describe web component are traversed to generate test cases for the initial version of component and identify those that can be re-run on the changed component. The regression test-suite thus generated for the component is optimized using “all-path” coverage criteria. The proposed approach yields substantial results with optimized test data generation.

5.1 Motivation for Testing Web Components

In today’s scenario, with the development of internet and intranet technology, the Web applications become more and more complicated with strict requirements of correctness, quality, reliability and inter-operability. Approximately 70% of websites contain bugs that reside in both static and dynamic pages (BIG-SF 1990). It reflects that either the existing web application tools are not sufficient to detect failures or developers felt that the time spent in testing is not worth the payoff. Therefore,
effective and efficient web component testing techniques have to be developed to tackle this problem.

Regression testing of web applications is a complicated task. This is because the traditional software development cycle is becoming shorter and shorter, that makes regression testing more challengeable. The development cycle of web component differs a lot from the traditional software component; as web applications consist of static as well as dynamic pages, which require a lot of interactions between users and Web browser. A novel regression testing technique for GUI based web components based on the concept of domain analysis and modeling is presented. Both the design and implementation of the testing technique are given.

5.2 Web Components

Web applications or web components currently make up one of the largest growing areas in software web. They employ a number of new languages, technologies, and programming model and are used to implement highly interactive applications with high quality requirements. In the recent years, with the rapid growth of WWW and the internet, web applications have become more prevalent around the world. They do not just give us new types of applications, but provide an entirely new way to deploy software applications to end users. Current web applications are sophisticated, interactive programs with complex Graphical User Interface.

There are a number of factors which differentiate between a web component and traditional software in terms of their testing requirements, like web applications are built on Internet and open standard technologies as compared to close standard technologies of traditional applications, and an important feature of web components is that unlike traditional software applications, they are not sold or distributed when updates are made. This makes maintainability crucial for web software. Moreover, unlike traditional businesses whose potential customers are typically limited by physical concerns such as geography and traffic, growth in web-based businesses has unlimited potential. Therefore, web components must be scalable and ready to grow in terms of servers, services, and customers.
5.2.1 Implications in Testing Web Components

A web-based client, such as a browser, communicates with a J2EE application through server-side objects called web components. Web components are of two types - Java Servlets and Java Server Pages (JSP). Servlets are Java programming language classes that dynamically process requests and construct responses. Web Components are establishing a new standard of adding functionality to a Web page. In contrast to traditional applications, web applications are built on Internet and open standard technologies. This brings new features to web applications and makes them extremely difficult to test. Following are some of implications involved in testing web components:

- **Heterogeneous nature**: Web components need to be tested not only for correctness and efficiency but for other quality factors like capability and interoperability as server components can be distributed to different operating systems, or client components may have different versions of web browsers running under a variety of operating systems.

- **Dynamic role of client and server**: Web components are mostly dynamic in nature. Hence, subsequent interactions between the client and server depend on the previous inputs that need to be considered.

- **Rapid change**: Web technologies evolve rapidly and the changes in web application requirements are also very dramatic. Therefore, web components require frequent testing and maintenance in an effective way to withstand peculiar time-to-market pressure.

- **Control of execution**: During the execution of web application, users can break the normal control flow without alerting the program controller. For example, users can press the back or refresh button in the web browser, which totally changes the execution context, causing unexpected results. Furthermore, changes in the client-side configuration may affect the behavior of web applications. For example, users can turn off cookies, which can cause subsequent operations to malfunction.
5.2.2 Criteria for Testing Web Components

Web applications are the vital components of the global information infrastructure. It is important to test them adequately for correctness and completeness to ensure quality and reliability (Kam 2008). A web component needs to be tested for the following aspects.

i. **Functionality Testing:** It incorporates testing all the links in web pages, database connection, forms used in the web pages for submitting or getting information from user and Cookie testing.
   - **Links:** All the outgoing links and all the internal links on a web page require testing.
   - **Forms:** They are the integral part of any web site. Forms require checking in terms of validation check on each field and default values of the fields.
   - **Cookies:** They are small files stored on user machine and are used to maintain the session, mainly login sessions. They need to be tested or enable and disable option by enabling or disabling the cookies in the browser option.
   - **Database:** Data consistency needs to be maintained in web application. It requires maintaining data integrity while performing any transaction on them.

ii. **Usability Testing:** It involves the following tests:
   - **Test for navigation:** Navigation means how the user surfs the web pages, different controls like buttons, boxes or how user uses the links on the pages to surf different pages.
   - **Content checking:** Content should be meaningful, check for spelling errors. All the anchor text links should be working properly.

iii. **Interface Testing:** The Interface testing includes testing of Web server and application server interface along with Application server and Database server interface.
iv. **Compatibility Testing:** A web component should support Browser compatibility, Operating System compatibility, Mobile browsing and Printing options

v. **Performance testing:** Web application should sustain to heavy load. Web performance testing should include Web Load Testing and Web Stress Testing

### 5.3 Regression Testing Framework Design

Web components are an integral part of business infrastructure. They are built on internet and open standard technologies, which contributes greatly to their popularity and success. Web components and their underlying systems grow over time. They need to be retested whenever there is a change to verify that the quality of the system has not altered. Testing consumes effort and time. A model-based approach to this problem of reusing existing test cases to test the modified component has been proposed. For testing web applications two different steps, namely Domain Analysis and Modeling, and Test case generation are followed, which are inter-related.

i. **Domain Analysis and Modeling:** Different modeling strategies such as object oriented model, finite state machine and modeling notations, such as use case diagrams, sequence diagrams etc., are used to generate the web models. Varying testing techniques employ different methods. The test model represents different parts of web application, which is relevant for testing purpose.

ii. **Test case generation:** After creation of web model, test cases are generated to test the component by traversing the model.

The proposed approach shown in Figure 5.1 consists of the following steps:

i. Domain Analysis and Modeling

ii. Model traversal and test case generation

iii. Optimizing test cases using coverage criteria

iv. Regression test suite generation
Initially, thorough analysis of the web application is carried out by accessing specifications of its every part. It includes its functionality in terms of links, button, database interactions etc, web applications static composition i.e. different pages, name of links and buttons it contains and transitions. From this information, a Graphical Web Model of the component is constructed. The model is traversed using ‘All Link Coverage’ to generate test sequences. By using this criterion it is confirmed that all the links available in the web application will be tested at least once.

**Figure 5.1: Framework for Regression Testing**

**Web Application Test Sequence:** WTS: T0→T1→T2→... →Tm is a sequence starting from root page to some leaf page that can be traversed, T0 is the initial page and ‘m’ represents total number of Test Sequences, where (m>0). WTS represents available navigation from page T0 to Tm by traversing different intermediate pages.

The last step of the framework is to generate regression test suite. The modified version of web application may incorporate or delete some pages or links. Once again, it goes through Domain Analysis and Modeling activity. Optimized set of
test sequences are generated in such a way so that all the links are covered. Test sequences following the same traversal path that differs only at the end points are merged. For example, if first test sequences is WTSi: T1→T2→T3→T5→T6 and second one WTSj: T1→T2→T3→T4 then we can merge these two test sequences as T1→T2→T3→T4→T3→T5→T6.

To reduce the number of test sequences, another coverage criterion i.e. “All Page Coverage” is followed. According to this coverage criteria, remove all the copied pages which are the leaf nodes and links corresponding to these copied pages from the web model. By doing so, the length of test sequences will automatically be reduced as it ensures that all the pages will be tested at least once.

5.4 Experimental Setup

To demonstrate the functionality of approach, login system of web application has been considered. Let it be named Web Sign Up and Login System (WSL). The respective application has been analyzed and Graphical Web Model (GWM) has been constructed, which is traversed to generate test sequences. The effectiveness is finally shown by incorporating changes and generating Regression Testing Suite.

Initially, a web application having a home page (Page id Pₐ), which is static in nature and further contains three links namely Login, Sign up and Search is considered. Login page (Page id Pₐ) is also static and contains one button namely, Submit. Logged in page (Page id Pₜₐ) contains two links Read and Log out. The Information page is dynamic in nature and contains two links – Continue and Search. Finally, Logged Out page (Page id Pₜ₉), containing three links Home, Login and Search. Error pages ((Page id’s P₉ₐ, P₉₉, P₉₆), which are invoked if there is any error and contain Return link. Sign up page (Page id P₉₉), and search pages (Page id P₉₉) are traversed based upon initial conditions.

Complete Graphical web model is represented in Figure 5.2 and the changed component web model is represented in Figure 5.3. In the changed component new pages are created, which results in more links. Now the two models differ in terms of some functional aspects that need to be tested. Therefore, test sequences for both are different.
Figure 5.2: Graphical Web Model of Component

Figure 5.3: Graphical Web Model for Changed Web Component
5.5 Results and Discussion

After creation of basic web model from the domain analysis of the requirements, the model is traversed to generate optimized set of test data and regression test suite.

Test Generation and optimization: Algorithms are written and implemented in Visual C++. The model is implemented as a queue data structure and test sequences are generated with the help of list, which is implemented with the help of ‘ArrayList’, an inbuilt class in Visual C++, which acts like list class. The output as test sequences and their statistics are shown in Figure 5.4. The process of optimization takes test sequences generated on the previous stage as its input, and the output is the optimal test sequence.

![Figure 5.4: Optimal Test Sequences of Web Component](image-url)
**Regression Test Suite Generation:** Some changes are incorporated in the web application to create newer version of the component. The regression test suite is generated for this component after its traversal, and comparison with web model of the initial web component. Figure 5.5 shows test sequences for modified web component along with the new and updated test sequences.

![Figure 5.5: Regression Testing Suite for Changed Component](image)

<table>
<thead>
<tr>
<th>TEST SEQUENCES FOR MODIFIED WEB MODEL</th>
<th>OLD TEST SEQUENCES</th>
<th>NEW and UPDATED TEST SEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1: A→B→C→D→E→D→B</td>
<td>S1: A→B→C→D→E→D→D</td>
<td>S1: A→B→C→E→K→E→F→E→A</td>
</tr>
<tr>
<td>S2: A→B→C→E→F→E→F→E→F→E→A</td>
<td>S2: A→B→C→E→D→E→B</td>
<td>S2: A→B→E→E→A</td>
</tr>
<tr>
<td>S3: A→B→G→B</td>
<td>S3: A→H→H→B</td>
<td>S3: A→B→K→B→K→A</td>
</tr>
<tr>
<td>S4: A→F→A</td>
<td>S4: A→H→H→H</td>
<td></td>
</tr>
<tr>
<td>S5: A→H→C</td>
<td>S5: A→H→H→H</td>
<td></td>
</tr>
<tr>
<td>S6: A→H→H→H</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S7: A→H→H→H</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S8: A→K→B→K→A</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**5.6 Comparison with Existing Methods**

The proposed framework offers an effective strategy for testing web-based components over the existing ones in the following aspects:

i. Test sequences are generated based upon all-link coverage criteria, which ensure thorough structural testing of the web model. Earlier techniques focused on all-page coverage criteria.
ii. Web model employs reverse traversal of links as well resulting in exploring all
the functionality.

iii. The testing criterion ensures that all the links and pages are traversed at least
once, thus achieving better test result.

iv. Regression testing is incorporated in the existing work, thereby making it
more effective for testing.

5.7 Summary

To deal with increasing diversity and complexity of web components, domain
analysis and modeling technique for testing web applications has been adopted.
Further, automation and optimization of test generation process has been taken. A
Graphical Web Model for the system has been generated and optimal test cases are
produced by applying all-path coverage criteria. Its purpose is to validate the usage of
the web application in order to guarantee its use without any failure. In the next
chapter, metamorphic testing technique has been explored. It is solely based upon
inherent properties of the system.