Testing of Web Applications based on Model Driven Architecture

Harsimrat Kaur[1], **Janpreet Singh**[2]

[1]Assistant Professor, Punjab Business School, Fatehgarh Sahib, Punjab, India
[2]Assistant Professor, CSE, LPU, Phagwara. Punjab, India

Abstract: ***The Conversion from desktop applications to web or remote applications makes website development as a growing area of software industry. During the analysis phase of web application development web model of the web application is created. This web model represents all the components of the web application that are to be tested. After generating the web model the very next step that is taken for testing the web application is to generate test cases. Web testing deals with this issue and during this testing we generate test cases. This testing is followed up by regression testing if we bring any feasible change in the web model. The regression testing verify whether previous model functionality will regress or fail by marking which previous test cases are valid or discarded in the updated model [12]. The updated web model is then used to generate the new test cases. Each and every time when we added a new module in the existing model then we have to check the effect of test cases generated earlier for existing modules because the new module added to the system may interact with the existing modules. In this paper we proposed an automatic way to deal with the modification part of the test cases existing for the already mentioned modules in the web model.***

Keywords: Web Model, Regression Testing, Web Applications.

1. Introduction

In the software development field the development process starts with the initial requirement gathering process then the requirements helps to make the design of the system and then in implementation part the identified design is transferred to its implementation in a specific language. After implementation phase of software engineering life cycle Testing phase appears in the list. Complete testing of a web-based system can help in addressing the issues before the system is revealed to the public [2]. Quality of the web site is very important to the user otherwise the user will leave for the different site if the web site is too complex and is of low quality. So methodologies for adequately analyzing, constructing, understanding, testing and maintaining web applications are essential. The criteria for testing websites are: Timeliness, Quality, Accuracy and Consistency, Response Time and Latency, and Performance. Issues such as the security and load and performance of the web application are the basic functionalities to be considered. But all these features are difficult to achieve because of certain constraints and challenges that make Web testing is a difficult task [6]. The problems that can occur depend on the application, technologies used, settings and environment, user software configurations etc. The current testing field focus is on techniques for unit–level testing and sub system testing [17, 18]. However web applications testing lack systematic testing methods and techniques to cope with challenges and constraints of testing. Currently the common strategies and techniques for web testing are adopted and usually driven by specific tools, metrics or immediacy rather than the objectives to achieve. The main issue of web testing is to generate test cases for all the modules identified to make a software system. So to achieve the Accuracy feature of web test case generation is a bigger challenge. In this paper we introduces a USE CASE model based system to generate automatic test cases for existing modules at the time when a new modification is done on the website or when a new module is added to the current existing website. This process is necessary because the newly added module on the website may affect the test cases of existing module if the newer one interact with the existing modules. The result part of this paper also addresses the issue that how much efforts have to spend to generate of update the existing test cases according to the increase in number of inputs from the newly added module. This paper is the extension of our previous review paper on which the current algorithm was proposed and this paper consists of the implementation results and comparisons of the different aspects of the proposed system.

2. Use Case Approach

Use Case diagrams of Unified Modeling Language (UML) provides the outer view of the system in which we have to identify the users who have to interact with the system, the different inputs that have to be provided to the system for the smooth functioning of the functions supported by the system, the different outputs generated by the system. Use Case diagrams also shows the hierarchy that the different outputs produced by the system will serve as inputs to other modules. The different aspect of the Use Case diagrams is listed in Table 1. The different users of the Use Case diagram are known as actors. Name of the Use Case is always written in the main rectangle used to represent the module name.

Table 1: Components of Use Case diagram

Use case name:	*It includes the name of the use case.*
Actor:	An actor is a person or other entity external to the software system being specified who interacts with the system and performs use cases to accomplish tasks.
Preconditions:	List any activities that must take place, or any conditions that must be true, before the use case can be started.
Post conditions:	Describe the state of the system at the conclusion of the use case execution. Post-conditions can be numbered like preconditions.
Base Flow of events:	Provide a detailed description of the user actions and system responses that will take place during execution of the use case under normal, expected conditions.
Alternate flow of events:	Describe any anticipated error conditions that could occur during execution of the use case, define how the system is to respond to those conditions and cover behavior of an exceptional scenario

3. Objectives

a) Identifying and analyzing the domain areas of Web applications.
b) Representation of Web applications by using suitable Web Architecture.
c) To identify the key attributes for Web Testing.
d) To select a specific testing strategy to be applied on the Web components.
e) Application of Web Testing techniques at different levels.
f) Verification and validation of results.

4. Proposed Methodology

Our proposed framework for testing of web application is as mentioned in figure 1. The proposed methodology is a combination of these seven steps. In the very beginning the web application is modeled into the high level use cases diagrams called UCTMs (Use Case Transition Models). In the second step use case diagram is designed into use case templates that show the functionality of Web application. Here the different inputs and outputs of the modules are identified from use case diagrams and putted to the automated test case generation system as input. Here those outputs are also identified from different modules which may serve as input to existing modules to check the functionality of existing modules. Testing method is implemented by transferring the textual description of use cases into WC- CDDs (Constraint Directed Diagrams). Thereafter the test case sequences are generated by traversing the WC-CDD. In the next phase we have worked to generate the regression Testing Suite by inserting the changes in the use case template and converting WC-CDD into MWC-CDD. The test sequences so obtained from it are then validated by giving the statistics like which old test sequences are reused, how many are updated or modified and newly added. This will lead to reduce the time while regression testing, as we need not to test sequences again which are used without any modifications.

a) Design of use case Template
b) Conversion of Template into WC-CDD (Web component Constraint directed Diagrams)
c) Generation of Test sequences
d) Insertion of changes in Use case Templates

In order to get the test scenarios conveniently, we transform the textual description of use cases to the WC-CDD. Because the flow of events are the most important part for generating test cases from use cases, we only transform the flow of events of use cases to the WC-CDD.

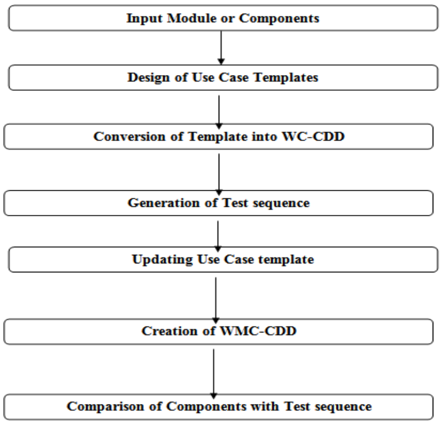

Input Module or Components
Design of Use Case Templates
Conversion of Template into WC-CDD
Generation of Test sequence
Updating Use Case template
Creation of WMC-CDD
Comparison of Components with Test sequence

Figure 1: Proposed Methodology

5. Results

In this results section we discussed two different scenarios where we tested our proposed system. First is the case of College Automation System (CAS), where we have a module Fee Collection. In this module some inputs and outputs are mentioned. When we update two new modules as FINE and SHOLARSHIP to the website, both the modules provide some inputs to the initial module Fee Collection. This interaction affects the initial test cases of Fee Collection module. So we have to update the test cases of Fee Collection module. Table2 shows the increase in the test cases of module under modules. In the results section we also take another example of module named "abc", the test cases of this module also affected by the introduction of other modules "mod_1" and "mod_2" as mentioned in table3. In the result section we also mention a new Table4 where we show the results as if we increase a new input to the existing module as 1 then the existing test cases are the half of the final test cases and if we increases the inputs to 2,3,4 and 5 then the composition of initial test cases in the number of final test cases is 25, 12.5, 6.25, 3.13 percentage respectively. Fig2 shows the scholarship module and its input and outputs, Fig3 shows Fee Collection module and its inputs and outputs, Fig4 shows the inputs and outputs of Fine module. Fig5 shows the interaction of Fee Collection module with Fine module and Scholarship module.

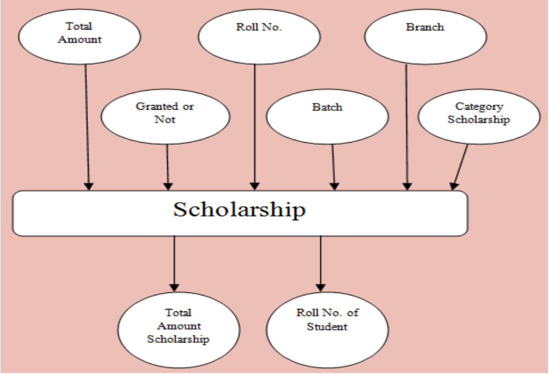


Figure 2: Inputs and Outputs of Scholarship module

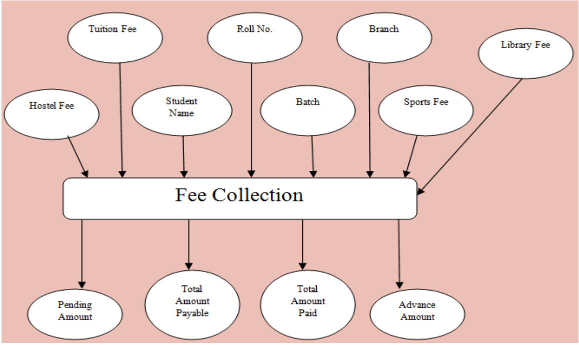


Figure 3: Inputs and Outputs of Fee Collection module

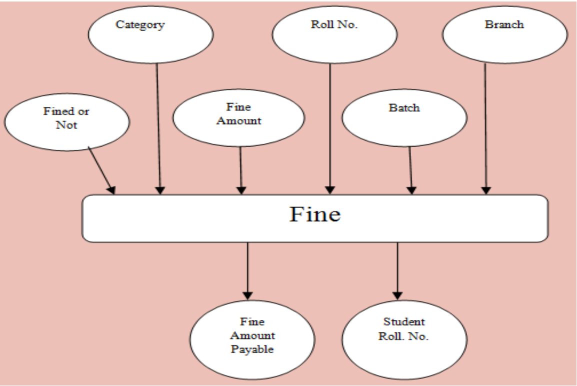


Figure 4: Inputs and Outputs of Fine module

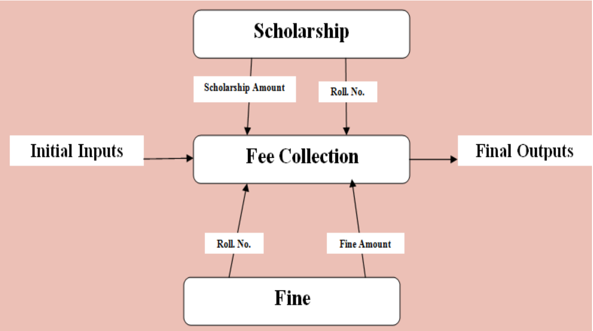


Figure 5: Interaction of different modules

Table 2: Results from Fee Collection Module

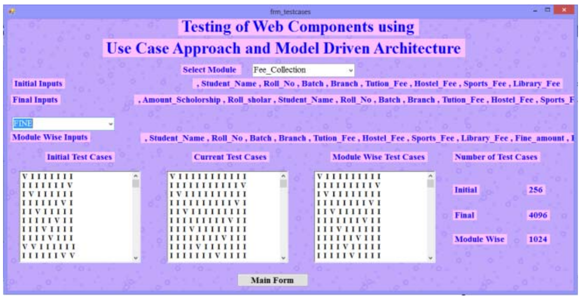
Module Name: **Student_Section**	Initial Inputs: **08**	Final Test Cases: **4096**	
Outside Modules	*Inputs supplied by other modules*	*Test Cases by other modules*	*% Composition of initial Test Cases*
Fine	2	1024	25.00
Scholarship	2	1024	25.00
Final Solution	**12**	**4096**	**06.25**

Table 3: Results from "abc" Module

Module Name: **abc**	Initial Inputs: **01**	Final Test Cases: **64**	
Outside Modules	*Inputs supplied by other modules*	*Test Cases by other modules*	*% Composition of initial Test Cases*
Mod_1	3	16	12.50
Mod_2	2	08	25.00
Final Solution	**06**	**64**	**03.13**

Table 4: Effects of Inputs to the test case Composition

Inputs Supplied	***% Composition of initial Test Cases***
1	*50.00*
2	*25.00*
3	*12.50*
4	*06.25*
5	*03.13*
6	*01.56*



Testing of Web Components using
Use Case Approach and Model Driven Architecture

Select Module: Fee_Collection

Initial Inputs: , Student_Name , Roll_No , Batch , Branch , Tution_Fee , Hostel_Fee , Sports_Fee , Library_Fee

Final Inputs: , Amount_Scholorship , Roll_sholar , Student_Name , Roll_No , Batch , Branch , Tution_Fee , Hostel_Fee , Sports_F

FINE

Module Wise Inputs: , Student_Name , Roll_No , Batch , Branch , Tution_Fee , Hostel_Fee , Sports_Fee , Library_Fee , Fine_amount , I

Initial Test Cases

```
V I I I I I I I
I I I I I I I V
I V I I I I I I
I I I I I I V I
I I V I I I I I
I I I I I V I I
I I I V I I I I
I I I I V I I I
V V I I I I I I
I I I I I I V V
```

Current Test Cases

```
V I I I I I I I I I I I
I I I I I I I I I I I V
I V I I I I I I I I I I
I I I I I I I I I I V I
I I V I I I I I I I I I
I I I I I I I I I V I I
I I I V I I I I I I I I
I I I I I I I I V I I I
I I I I V I I I I I I I
I I I I I I I V I I I I
```

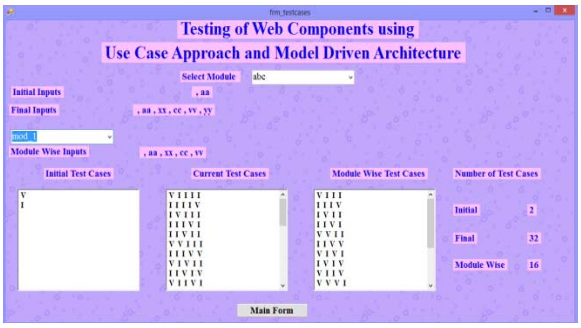
Module Wise Test Cases

```
V I I I I I I I I I
I I I I I I I I I V
I V I I I I I I I I
I I I I I I I I V I
I I V I I I I I I I
I I I I I I I V I I
I I I V I I I I I I
I I I I I I V I I I
I I I I V I I I I I
I I I I I V I I I I
```

Number of Test Cases

Initial	256
Final	4096
Module Wise	1024

Figure 6: Implementation Screenshot-1



Testing of Web Components using
Use Case Approach and Model Driven Architecture

Select Module: abc

Initial Inputs: , aa

Final Inputs: , aa , xx , cc , vv , yy

mod_1

Module Wise Inputs: , aa , xx , cc , vv

Initial Test Cases

```
V
I
```

Current Test Cases

```
V I I I I
I I I I V
I V I I I
I I I V I
I I V I I
V V I I I
I I I V V
V I V I I
I I V I V
V I I V I
```

Module Wise Test Cases

```
V I I I
I I I V
I V I I
I I V I
V V I I
I I V V
V I V I
I V I V
V I I V
V V V I
```

Number of Test Cases

Initial	2
Final	32
Module Wise	16

Figure 7: Implementation Screenshot-2

6. Conclusion

As we observed from the literature survey that the dynamic nature of websites requires updates and modifications time to time. During the development of websites the different modules are tested after development of those modules by creating their use cases and then by making test case for each use case. And when some modifications have to be done on websites during their modification process then we make test cases for only the modified aspects of the websites to test the performance of new inserted portions. But we ignore the fact that the new insertions may disturb the test case already generated for the existing modules as the modified portion may interact with the existing modules. Also this may lead to software disaster or unacceptable conditions by website users. we proposed a solution for the above problem that whenever there is a requirement to modify a website by inserting some new modules then during the testing of new modules when we create use cases for those new modules and generate test cases for each use case then we have to consider the previous use cases of the existing modules which may be affected with the introduction of new modules and we have to updated the test cases of those existing modules on the basis of their new use cases then check the overall performance of website. This technique definitely will help to reduce the unwanted effects produced during the insertion of new modules. Result section of the paper also discussed about the %age of initial test cases in the final test cases.

References

[1] E.Hieatt, R.Mee, "Going Faster: Testing the web applications", IEEE Software, Mar.2002, pp.60-65.

[2] Andrew, J. Offutt, R. Alexander, "Testing Web Applications by Modelling with FSMs", Software and Systems Modelling. 2004, pp. 326-345.

[3] Jeff Offutt, Ye Wu, Xiaochen Du and Hong Huang, " Web Application Bypass Testing" , In Proceeding of the 28th Annual International Computer Software and Applications Conference (COMPSAC 2004), Oct. 2004, pp. 106-109.

[4] Y. Qi, D. Kung, E. Wong, "An Agent-Based testing Approach for Web Applications", In Proceedings of the 29th Annual International Computer Software and Applications Conference (COMPSAC'05), 26-28 July.2005, pp. 45-50.

[5] Adriana Carniello, Mario Jino, Marcos Lordello Chaim , "Structural Testing with use cases", JCS&T Vol. 5 (No. 2), Aug. 2005, pp. 100-106.

[6] Ghulam Mustafa, Abad Ali Shah, Khadim Hussain Asif and Amjad Ali, "A Strategy for Testing Web Based Software", Information Technology Journal 6(1): 74-81, 2007.

[7] Shengbo Chen, Huaikou Miao, Zhongsheng Qian , Shanghai Univ,Shanghai, "Automatic Generating Test Cases for Testing Web Applications", International Conference on Computational Intelligence and Security Workshops, Dec. 2007, pp.881-885.

[8] Atifah Ali, Aamer Nadeem, Muhammad Zohaib Z. Iqbal and Mohammad Usman, "Regression Testing based on UML Design Models", In 13th IEEE International Symposium on Pacific Rim Dependable Computing, Dec.2007, p.85-88.

[9] Hassan Reza, Kirk Ogaard, Amarnath Malge, "A Model Based Testing Technique to Test Web Applications Using Statecharts ", Fifth International Conference on Information Technology, Apr. 2008, pp. 183 – 188.

[10] N. Alshahwan, M. Harman, "Automated Session Data Repair for Web application Regression Testing", International Conference on Software Testing, Verification and Validation, Apr. 2008, pp. 298 -307.

[11] Huaikou Miao, Zhongsheng Qian and Bo Song, "Towards Automatically Generating Test Paths for Web Application Testing", International Symposium on Theoretical Aspects of Software Engineering, June. 2008, pp. 211 – 218.

[12] Javier J. Gutierrez, Maria J. Escalona, Manuel Mejias, Jesus Torres, Arturo H. Centeno " A Case study for generating Test cases from Use cases", In second International Conference on Research Challenges in Information Science, June. 2008, pp. 209-214.

[13] Liping Li, Huaikou Miao, Zhongsheng Qian, "UML based approach to testing web applications" International Symposium on Computer Science and Computational Technology, Dec. 2008, pp. 397-401.

[14] Liping Li, Huaikou Miao "An approach to modelling and testing applications based on use case", International Symposium on Information Science and Engineering, Dec.2008, pp. 506-510.

[15] Ananya kanjilal swapan bhattacharya and sabnam sengupta, "Analysis of Object- Oriented Design: A Metrics Based Approach", In TENCON 2009 - 2009 IEEE Region 10 Conference, Jan. 2009, pp. 1-6.

[16] B. Kam, T. R. Dean, "Lessons Learned from the Survey of Web Applications Testing", Sixth International Conference on Information Technology: New generation, Apr.2009, pp. 125-130.

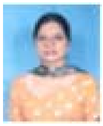
[17] Z. S. Qian, "An Approach to Testing Web Applications Based on Probable FSM", International Forum on Information Technology and Applications, May.2009, pp. 519-522.

[18] Gabriela Robiolo, Cristina Badano and Ricardo Orosco, "Transactions and Paths: two use case based metrics which improve the early effort estimation", In Third International Symposium on Empirical Software Engineering and Measurement, Oct.2009, pp.422-425.

[19] Ashalatha Nayak and Debasis Samanta, " Automatic Test Data Synthesis using UML Sequence Diagrams", In journal of Object Technology, vol.09 (No.2), March-April 2010, pp. 75-104.

[20] Hesham A. Hassan and Zahraa. E. Yousif, "Generating test cases for platform independent model by using use case model" In International Journal of Engineering Science and Technology, Vol. 2(12), 2010, pp. 7665-7675.

Author Profile



Harsimrat Kaur received the B.A. from Punjab University, Chandigarh (India) and MCA degrees in Computer Applications from Guru Jambheshwar University, Hisar (India) in 2000 and 2003, respectively. During 2010-2013, she is pursuing M. Tech (CSE) from Lovely University, Phagwara (India) in Computer Science and Engineering.