Model based Test Cases Generation for Web Applications

Vikas Suhag
Department of Computer Science & Engineering,Deenbandhu Chhotu Ram University of Science and Technology, Murthal(Sonepat),Haryana,India

Rajesh Bhatia
Department of Computer Science & Engineering, PEC University of Technology, Chandigarh, India

ABSTRACT
With the advent of web 2.0, web application architecture has changed a lot with increasing complexity. Modeling of the web applications has a key role in development; if modeling is inappropriate then the developed application will be poor. Testing is an important activity to improve the quality of web applications. So a technique for Model based Test case generation for Web applications have been proposed. Web Diagrams and Sequence Diagrams are used to model the behavior of web application under Test (AUT). Web diagram provides the functional requirement and sequence diagram provided the most important part of the web application, the Navigation between web pages. Navigation is the only thing that makes difference between stand alone applications and Web application. The proposed technique is validated with various case studies.

General Terms
Model based Testing, UML diagrams, Web Diagram, Sequence Diagram, Web Testing,

Keywords
Web Testing Technique, UML Modeling, Web Diagram, Sequence Diagram, UML based Web Engineering, Software Testing.

1. INTRODUCTION
Testing is the process of executing any program with the intent of finding yet undiscovered errors [19]. Testing is the most time consuming phase in Software Development Life Cycle (SDLC). So it must be started as early as possible to save the cost to fix errors and manage the schedule. Model based testing is a type of black box testing approach in which test cases are generated from UML models. UML models can be used to represent desired behavior of the web applications. Standard UML 2.0 provides us with Class Diagram, Sequence Diagram, Activity Diagram Use case Diagram, Collaboration diagram.

With the advancement of technology and large number of IDEs available to programmer, it has made development easier. Similarly user expectation are also increasing, user expect a well defined, easy to use and very large system for their business role. With demand of more functionality, quality, usability of software artifacts the complexity of system has increased multi fold, so to cope up with these challenges the development teams need to make some extra effort over testing of application to guarantee quality policy as well as proper functioning. So testing of an application need to be started as soon as its requirements are known. This gives rise to new form of testing, called Model Based Testing (MBT). At present MBT is widely used in development of large business application like Banking, health care, E-commerce.

In MBT as soon as the user requirements are available, design team will make a design model of the web application to be developed. Designing is usually done with UML (Unified Modeling Language) Diagrams that consist of Class Diagram, Sequence Diagram, Interaction Diagram, Use case Diagram, and State Transition Diagram.

But UML 2.0 does not provide a way to model some specific aspects of every web application. UML provides a mechanism to extend any of its Diagram using the Stereotypes, Tagged Values and constraints. So different extension are created to model the behavior:

- UML based Web Engineering [20].
- Web Modeling Language (WebML).
- Navigational Development Techniques (NDT).
- Web Diagram (WD).

In our test case generation scheme, we have used **Web Diagrams** and **Sequence diagram** to represent the behavior of the web application to be tested.

The remaining paper is organized as follows: section 2 describes the related work. Section 3 will define the proposed scheme and present algorithm for the same. Section 4 and 5 present the experimental setup and the results and discussion. Section 6 will present a case study on which tool is applied to generate test cases. In last describe the conclusions and future scope.

2. RELATED WORK
UML is the standard language that is most widely used to visualize the behavior, functioning, security aspects and documentation of software systems. But it is not suited to all, so it has a provision for the extension by which designer may specify own Stereotypes, Tagged Values and Constraints to design new UML diagrams according to need. Salim and Mustpha[1] has focused on modeling security requirements in web application design process to enforce various security parameters like Availability, Authentication, Integrity, Secrecy and Non- repudiation. They have also proposed a New UML design extension for the Web application design to include information about all these security concerns. Inclusion of security requirement in the design itself will help reduce cost involved in security implementation during later phases of SDLC. Salim and Mustapha [1] discuss the *UMLsec*, an UML extension for security proposed by J. Jürjens which define various UML profiles used for modeling

the secure applications. They have modeled the UML security extension using *secure context model*, *security cases* and *critical scenarios* to present security view of application, secure interaction of objects and data security constraints for logical view and protected hardware configuration for technical view of web application. They have also used Collaboration diagram to develop secure Context model, use cases diagram for Security Cases model, sequence diagrams to represent Critical scenarios and data security is represented by defining security constraints on the class diagram of web application [1]. Azrul [2] presented an extension to UML modeling for web hypermedia designing of web application. It describe the principles of design processes with two design issues – complex processes and interactions. It provides information on a UML-based hypermedia design method, called Com+HDM. It also demonstrates the design efforts taken to model a case study, including conceptual design, navigation design, and user interface design. Web hypermedia application demands a number of additional design aspects including on its complex domain structures, information contents, various interactive operations and functions, navigation links, access mechanisms, and presentation layouts. It also lists the modeling elements in the form of stereotypes for its Com+HDM (Comprehensive Hypermedia Design Method for Complex Process Modeling): a systematic design method for web application using UML as the standard language and stereotypes to extend UML functionality [2]. Com+HDM divide the modeling approach into multiple layers to provide modularity. So it has conceptual design, navigation design and user interface design for modeling hypermedia applications. Conceptual is used to present complex structure of application , navigation design gives how elements interacts and represent flow between user and application through user interface [2]. Fujiwara [3] has also presented an approach for test data generation for web application by modeling the web application using UML Class Diagram and OCL (Object Constraints Language). UML class diagrams are used to represent data structure or components and events while OCL are used to present behavior of those components and data constraints. Each test case generated contains pre-state, post state and triggering of event. It uses Satisfiability Modulo Theories (SMT) solver to treat more complex data types. Web application requires more testing and maintenance than other software system. Web application are too complex and uses n-tier architecture, database servers, application servers, web servers, different types of user platform with ranging from Operating System to Browsers. The cost of rerunning the test case is extremely high so number of test cases must be reduced using some test case prioritization strategies [4]. Boni [5] and Juan proposed a test automation framework for automated functional testing based on the navigation model of web applications. Test automation is divided into 4 levels:

1. Test case generation

2. Test data derivation

3. Test case execution

4. Test case reporting

This test automation framework accepts 1. UML models 2. Selenium scripts and 3. XML files as input. Test cases are generated for each path in the navigation model and test data is derived from the application under test for each path and this test data is fed to selenium for the execution, finally giving a report for each test case [5].

Navigation model used here comprise of different UML diagrams namely, 1) Use case Diagram to model functional requirement of the Application under Test , 2) Activity Diagram for flow of application under test, 3) Presentation diagram for representing data that is handled by application under test [5]. Maras and Petricic [6] developed a reverse engineering tool called phpModeler. phpModeler generates static UML diagram that show the content of the web page like functions and dependency between web pages. It allows the web application to be modeled at higher level of abstraction. Homma et al. [7] proposes a method that model the web application's page transition and internal state transition using the two- finite state automata. Now web application properties are presented using LTL (Linear Temporal Logic) Formulae for the verification. From the model developed using the Finite state automata test cases are generated. Each state or action can be treated as test target. Test case generated covers the pages, internal state and combination of pages and internal state. Offutt et al. [8] focus on the problem of loose coupling, user's ability to change the flow of execution and dynamic generation of HTML page which causes potential change in the control flow of execution of any web application. All these problem leads to the inability to determine the control flow. So they present a way to model web application to eliminate those entire shortcomings which lead to poorly developed web sites. Ernits et al. [9] show how the model based testing can be applied to web application under test using NModel. NModel is a C# based lightweight toolkit designed to establish domain specific testing of web application. Models are developed using C# and test cases are generated from the models. They also presented a case study on commercial system WorkForce management (WFM). Main purpose of their testing is to verify correctness of WFM during the startup and shutdown. Koopman et al. [10] described a system built over a model based testing tool GAST. Expanding software system to web application makes it platform independent and user is familiar to look and feel. Web applications are accessed through Web Browser which has a set of navigation control like Back and Forward Button which changes the state of web application. But this change in state is not propagated to server so it may cause inconsistency with application. Koopman et al. [14] also presented a novel tool for model based 'on the fly testing' of the web application. Web applications are modeled using the Extended State Machine (ESM).

Souza et al. [11] developed a tool for framework based development of web information system called FrameWeb. It was developed to ease the development of design and help the developer during the coding of application. It defines the basic architecture for developing the components of a web application. It further eases the designing and development by dividing the application into different components. Frameweb consists of 4 models to define the complete design for the application to be developed using the Frameweb: Domain Model, Persistence model, Navigation model, Application model. All the UML profiles so created are extension to standard UML class diagrams. Castelluccia et al. [12] introduces WAVer, a tool for verification of Web application design by Symbolic Model Checking Techniques. It combines the ability to perform designing of the web application and the verification of the design document or diagram by means of an automated tool. Feldstein [15] emphasize that automation of testing has lesser chances of detecting the bugs as compared to manual testing. Since automated testing lacks random behavior in test automation software. Sokenou [16] uses UML protocol state machine for specification of objects

life cycle but it does not define the actions. Post condition in test oracles are obtained from state machine diagrams and sequence diagrams are the main source of information like pre-condition and initialization of objects. Generated test cases are used for Unit testing and Integration testing. Existing work in model based testing is summarized in table 1.

Table 1: Diagrams used in model based testing.

Paper Ref.	Diagrams Used for Modeling
[1]	Class, Sequence, Use case, Collaboration diagrams
[3]	Class Diagrams
[4]	Sequence Diagram
[5]	Activity, Use case diagram
[7]	FSM and Internal State Automaton
[8]	Control Flow graph
[9]	NModel
[10],[14]	Extended State Machine
[11]	UML extension based on Class Diagrams
[12]	Finite State Machine(FSM)
[13]	Class, Sequence and State Chart Diagrams
[16]	Sequence and State Diagrams
[17]	Class, Sequence, Activity and Navigation map
[18]	Class, State and Object Diagrams
[21]	Class, Sequence and State Diagram

3. PROPOSED METHODOLOGY

Design is blueprint of the application to be developed. Design can also be used by testing team to generate test cases for behavioral testing. Model Based Testing (MBT) is a widely used approach to generate test cases from design documents. In this paper, we have proposed a MBT technique based on following UML diagrams

1. Web Diagram
2. Sequence Diagram

Web diagram provides following elements like client page, server page, form, frameset, JavaScript, HTML pages and links (Redirection, Builds, and Submit) for designing web application architecture. But we are using the model based testing (a Black Box Testing), so omitted the usage of server pages, that contains server side scripts or business logic. Web diagram have only Client Page, Form, Frameset and Links to model the application with the help of association, generalization and dependency between web pages.

In most modern web applications, you will find a navigation menu over the top or on either side of the web application which remains constant throughout the web application and allow user to move to any part of the application easily and allow user's understanding about the features of the web application. Now this is presented using the concept of master page/template which hosts the collection of content pages to allow the Navigation menu and other information like address of communication,visitor counter and other information related to header and footer of the web application that is constant throughout. Now master page will include the *Framset* to maintain this behavior. *Frameset* will contains a collection of client page that will be available in the navigation menu for easy acces. Now each client page may further contains *Frameset* to host part of *client page* from other web site or within the application. Each client page may also contains *Form. Form* attributes will now form the attribute of the client page to which it is connected so attributes in any client page will have a collection of its own attributes and attributes of the related *form.* Similarly operation of the *Form* will corresponds to the operation of the client page.

Compnent of the web diagram for the application include:

1. Client Page
2. Frameset
3. Form
4. Links

All these components when used to design a web application will form a tree like structure. Tree like structure will allow us to utilize the parent child relationship to get the required details from the web diagram. For example any *form* that is related to some client page will act as the child of that page. Similarly master page will act as the parent of the complete web application i.e. root element. Frameset hosting of all the content pages for the master will be parent to all those content pages. Relationship is used to get attributes with their data type and operation from child page to parent page.

Client pages in web diagram depicting web application architecture contain attributes corresponding to input elements of web page. Operation correspond to events (dynamic entity) that will allow user to navigate through web application using hyperlinks, button clicks, hovering and other events that would result in state change.

Web application uses templates / master page as reusable components. Templates are mainly used to hold static content, which remains unchanged throughout the application. A template contains header and footer information, which contains duplicate links and most highlighted content. A template contains duplicate links and form elements. We ignore these templates to avoid redundant test cases. Frameset in conjunction with the templates make the web applications more modular. It allows only changing main content of web page while retaining all static content thus, reducing the bandwidth requirement for surfing.

Two scenarios have been used to collect user input in Client Pages. In first scenario, all the input tags represent as Attribute are embedded in web page. It is most used in small application. Second scenario involves putting all the input elements in separate form to allow code reusability across multiple domains. This is used when application is complex.

Web diagram defines the static behavior of the application under test. Web application also involves dynamic behavior. This dynamic behavior involves change of state during navigation through application. Dynamic behavior is represented by sequence diagram. Sequence Diagram demonstrates the change of state of application in response to each operation specified in web diagram during web page design. Operations involve submitting an input form; clicking any button or hyperlink, hover etc. These operations may lead to change in state of web application; change in state is denoted by sequence diagram by showing an arrow from source page to target page. State diagram can also be used to model the change is state but in web application there could be some states that involves single order of operations to reach target page. So when sequence of operation becomes important, sequence diagram wins over the state diagram. Sequence diagram also define the values that may be sent via query string in URL with various events (Operation). These values are termed as Arguments.

Design data of web application is available in XML files from Web Diagram and Sequence Diagram. Parsing the XML files to generate test cases is tedious and resource intensive task. So we mapped the XML files into Database. Database tables are much more organized and efficient to work using queries, joins, stored procedure, Transactions, views, partitioning. But while mapping XML into Database some critical details are lost, so some direct parsing of XML file has also been carried out. Only a smaller portion of details are mined from XML, hence does not affect much to efficiency of system.
Tool designed to generate test cases from the database in combination with XML file uses the algorithm 1, algorithm 2 and Algorithm 3. Algorithm 3 merely joins the result from algorithm 1 and algorithm 2 to give final test cases.

Algorithm 1 will extract the information about all the input fields (Attribute) and their data type with respect to each web page and events (Operation) that can be triggered by users.

Steps:

1. Find all Client pages from database containing web diagram details.
2. Select one Client Page Name
3. For (Client Page Name)

   If (has form)

   Then

   Find all the Attributes: Data Type pair and operation corresponding to Current Form element. Add to Table 2.

   Else if (has Frameset)

   Repeat Step 2 and 3 with Client Page Name=Frameset->Client Page Name.

   Else

   Find all Attributes: Data Type pair and operation from Current Client Page. Add to Table 2.
4. Repeat Step 2 and 3 till each Client Page is processed.

Table 2: Output from Algorithm 1

Source Page	Attribute: Data Type	Operation

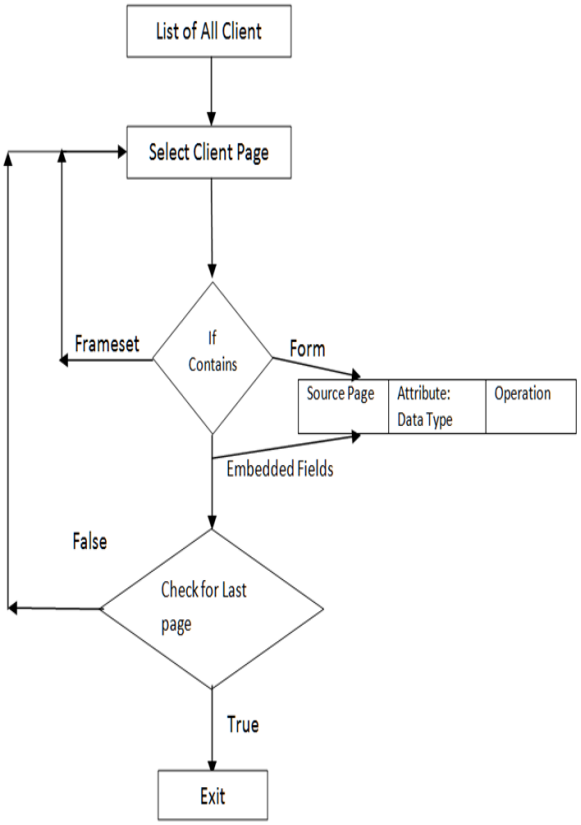

Fig. 1: Data mining from Web Diagram

Algorithm 2: will list all the web page changes due to events defined in web pages (Client Page).

Steps:

1. Find all Client Pages from Database containing Sequence Diagram details.
2. For each Client Page,
   Find all the Events and their corresponding landing page (Target Page) and argument: Data Type pair. Add Source Page (Current Page name), Event (Operation) and Target Page, Argument: Data Type to Table 3.

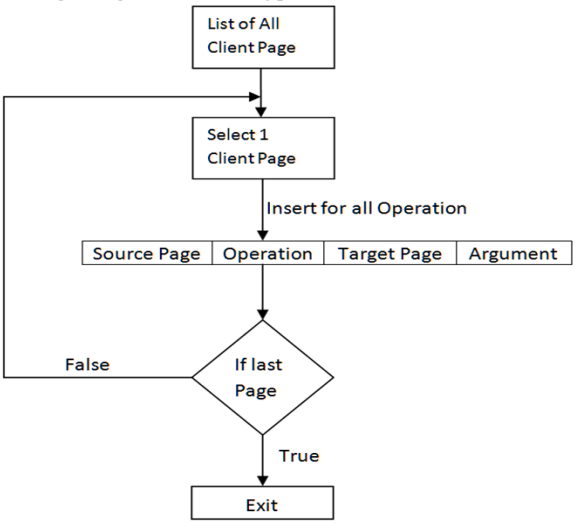

Fig. 2: Data mining from Sequence Diagram

Table 3: Output from Algorithm 2

Source Page	Operation	Target Page	Argument

Algorithm 3 will combines the results from Algorithm 1 and Algorithm 2 to generate final test cases as by applying right outer join between the Table 2 and Table 3 with Operation as Foreign key.

Resultant test case Schema is shown in Table 4.

Table 4 Resultant Test Cases Schema

Source Page	Attribute:DataType	Operation	Target page	Argument

Let us illustrate these test cases with an example. We have a web page for searching content. It comprises One Input Text Field to get user input for searching and Two Buttons labeled Search and Reset. Input Text field will act as the Attribute and Button will correspond to Events (Operations) for the Search page. Values that will be accepted by search field are denoted by the Data Type like Number for Telephone Number and String for name, address etc. Event may change the state of application so are related with source page as current page where event is declared and Target page as the page, where the search result will be shown when event is fired, it may be same page or some other page. Argument will contains the search input if GET method is used to process the data otherwise in POST method argument will be empty.

4. RESULTS & DISCUSSION

Test cases are generated using the data mining of databases of web diagram and sequence diagram. Data mining apply SQL methods to extract pattern of data. Tool has been applied to a complete fully functional real world application, which uses all of the real artifacts that a tester will encounter while dealing with the actual web application under development.

Test cases generated assess the complete functionality of the web application. Test cases provide the details like how the information flow will take place across the web application. Test cases allow web application to be tested against all the anomalies due to poor validation of input controls in web application. Anomalies that may occur due to poor handling of query string values and data type allowed. Test the navigation scheme, whether all redirects, events and hyperlinks are working as intended or not.

Hyperlinks results in navigation from host page to target page. Attribute values will be null in case of operation is a Hyperlink. Hyperlinks may use query string, then argument filed will have the argument name and its data type to generate test value against those to use while testing. If attribute column will contains Name: Type pair to let, value to be generated for form fields to check against input validations for XSS(Cross Site Scripting) and SQL Injection attacks. When attributes are not null then operation is form button click, hover, mouse out, focus out or some other event that causes validation checking. Generating values for the Attributes will check whether application is safe against input based attacks similar for arguments. Violation of navigation will be detected if operation result in different target page then specified in test cases from same source page during multiple execution sequences. Source Page is regarded as pre-condition and target page as post-condition for testing. Test cases generated are abstract test cases. Abstract test cases are not directly executable so test values need to be generated for execution.

5. CASE STUDY

Test case generation technique has been applied to a web application which allows sharing of previous year papers, notes and other academic notices among each other. It provide functionality to upload their paper and notes as well as allow downloading the same with job details, admission notices, conferences, sample resumes and information regarding the various exams. It also facilitates comments for healthy discussion among students. There are some problems like comment system is used on multiple pages so duplicating it at multiple pages is not appropriate. So it is put into a form and attached to every page that uses it. Application also contains a navigation pane to move around within the application. Navigation pane is contained in the master page that allows links in pane to be constant throughout the application. It contains a number of links, so those links are modeled as the attributes with data type as *Link* in master page. But considering those links in the sequence diagram will make the diagram too complex, because these are accessible from every state of application and causes redundancy in test cases. Redundant test cases will result in considerable wastage of time and resources in their execution. So such test cases are omitted from our test case generation scheme. Fig. 3 and Fig. 4 show the Sequence diagram and Web diagram for the case study respectively. Table 5 shows the resultant test cases generated by our technique.

Table 5 Test cases Generated from case study

Source Page	Attribute	Operation	Target Page	Argument
Index	Null	Previousyearpaper	previousyearpaper	
Index	Null	previousyearpaper_download	previousyearpaper_download	id=Integer
Index	Null	previousyearpaper_download	previousyearpaper_download	branch=String
Index	Null	govt-jobs	govt-jobs	
Index	Null	govt-job-detail	govt-job-detail	id=Integer
previousyearpaper	Null	previousyearpaper_download	previousyearpaper_download	id=Integer
previousyearpaper	Null	govt-job-detail	govt-job-detail	id=Integer
previousyearpaper	Null	Login	login	
previousyearpaper	commentTxt:String,email:Email,name:String,	Submit	previousyearpaper	
previousyearpaper	Null	Index	Index	

paper-uploader	department:String,file:File,month:String,semester:String,subject:String,university:String,year:Integer,	Submit	paper-uploader	
reference-books	Null	computer-science-engineering-books	computer-science-engineering-books	
reference-books	Null	electronics-communication-reference-books	electronics-communication-reference-books	
reference-books	Null	mechanical-engineering-books	mechanical-engineering-books	
reference-books	Null	electrical-engineering-reference-books	electrical-engineering-reference-books	
admission-notice-list	DropDownList1:String,	DropDownList1_SelectedIndexChanged	admission-notice-list	
govt-jobs	Null	govt-job-detail	govt-job-detail	id=Integer
govt-jobs	Null	Login	login	
govt-jobs	Null	Submit	govt-jobs	
register	Null	Login	login	
register	Null	forgot-password	forgot-password	
register	Null	Termscondition	termscondition	
register	captcha:Integer,checkBox1:Boolean,email:Email,name:String,	Button1	register	
login	Null	Register	register	
login	Null	forgot-password	forgot-password	
login	Null	button4	Index	
forgot-password	Null	button4	forgot-password	
forgot-password	Null	Contactus	contactus	
changePassword	TxtCNPassword:String,TxtEmail:Email,TxtNPassword:String,TxtOPassword:String,	BtnChangePassword	changePassword	
contactus	editor1:String,emailid:Email,name:String,	sendMsg	contactus	
advertise-with-us	Null	Contactus	contactus	
notes	Null	Login	login	
notes	Null	Submit	notes	
add-notes	attachment:File,course:String,editor1:String,reference:String,subject:String,topic:String,	submitNotes	add-notes	

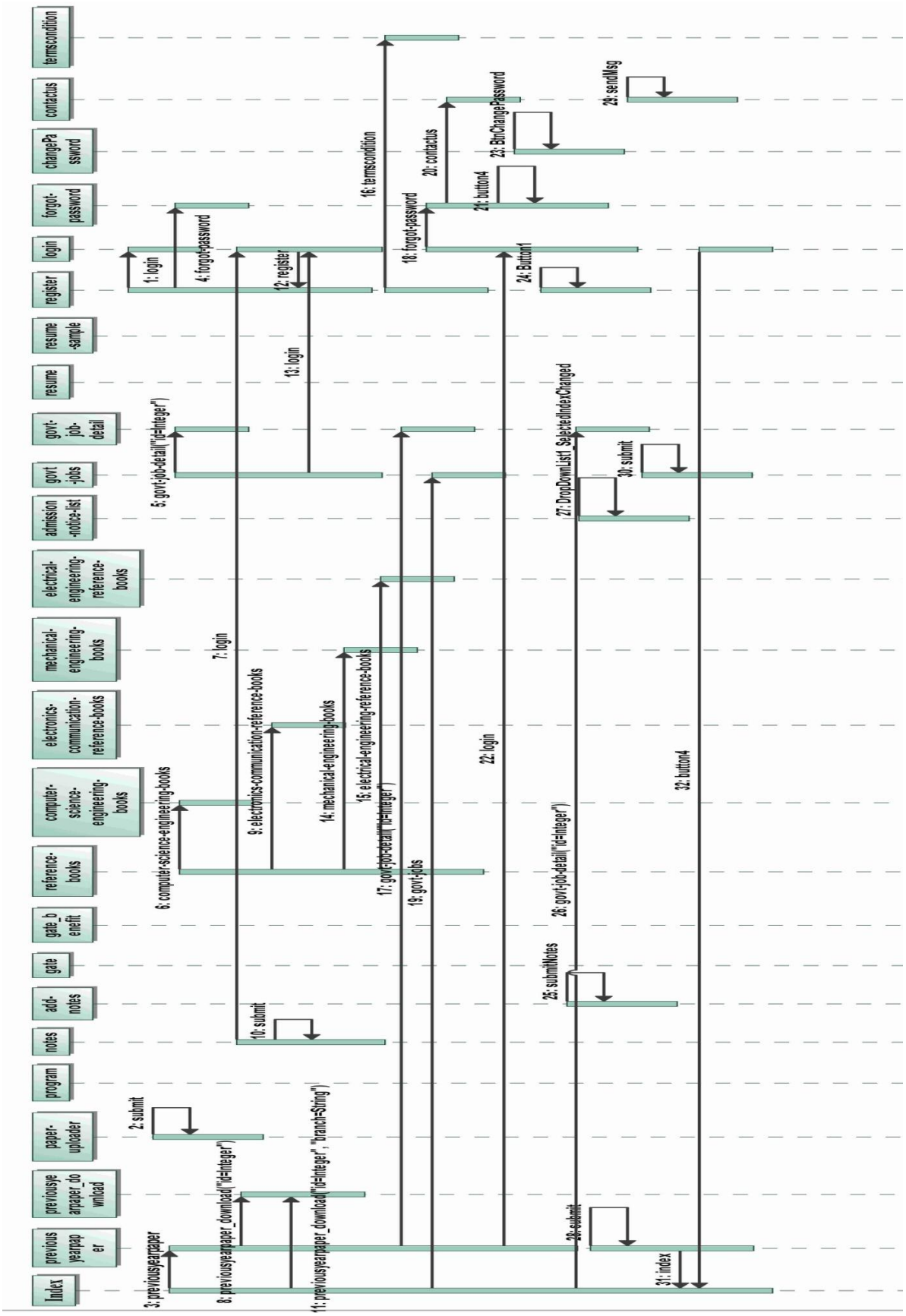

Fig. 3 Sequence Diagram of the case study

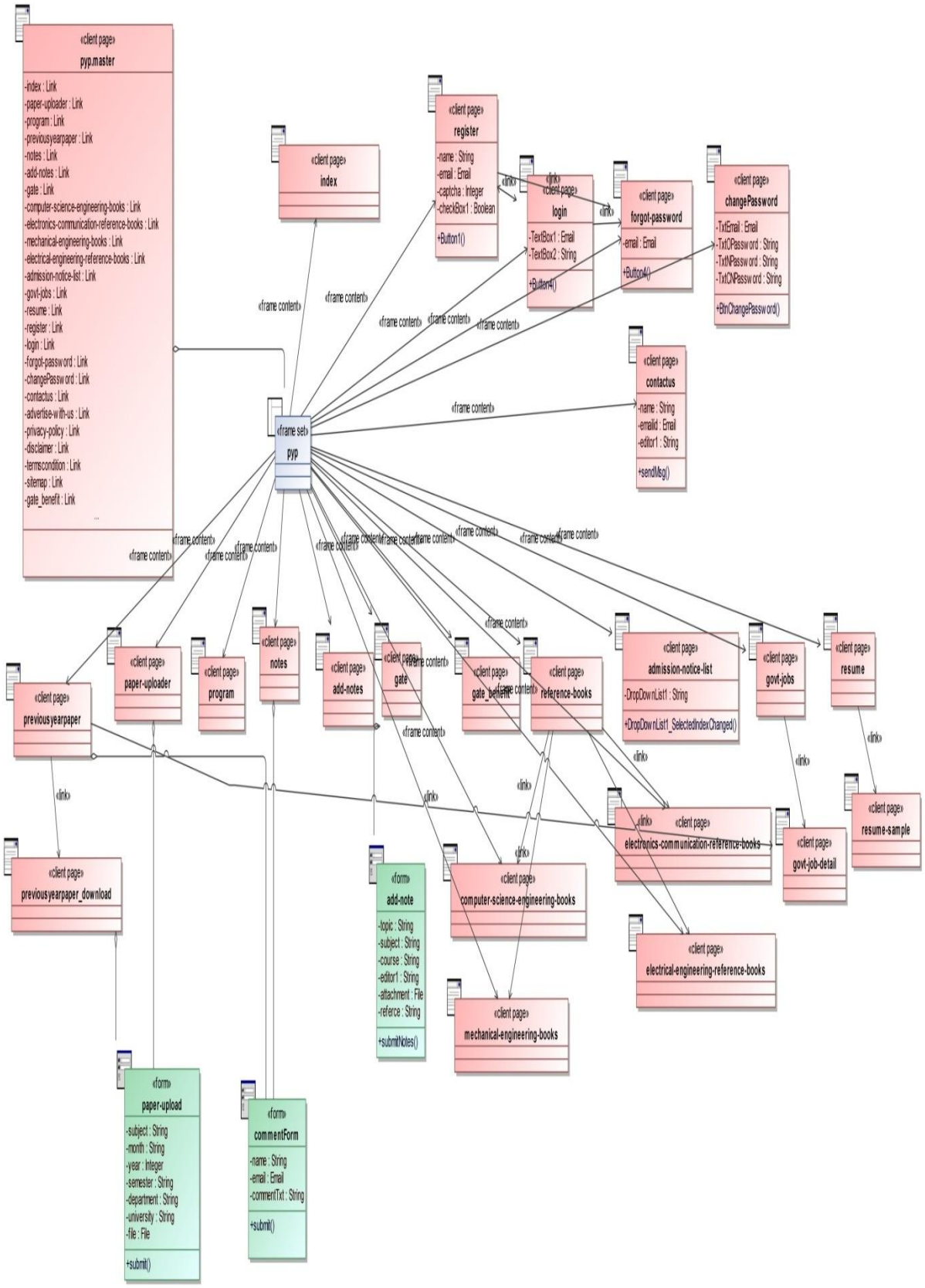

Fig. 4 Web Diagram of the Case Study

6. CONCLUSION AND FUTURE SCOPE

A model based test cases generation technique for web applications has been proposed. Proposed technique makes use of UML diagrams namely Sequence and Web diagrams for test cases generation for web applications. Web diagrams give the functional requirements like attributes, operation, form and frameset and sequence diagram present the navigation (dynamic behavior) of the Application under Test. Now Web diagrams and Sequence diagrams are mapped to Database. Test cases are generated by applying data mining principles of SQL. The technique has been verified and validated with case studies ranging from simple to complex considering the real situations that will arise during the testing process. Test cases generated provide pre-condition and post-condition for the test execution.

The major contributions are concluded as:

- As Test cases are automatically generated from Sequence and Web diagrams, problems related to omitted data have been overcome.
- Use of Sequence diagrams helped to capture the dynamic behavior of web applications.
- The extraction of data from combination of Web and Sequence diagrams has improved the test cases information.

Currently our approach is only applicable to the web application which uses single master page. It may be extended to support multiple master pages per application.

7. REFERENCES

[1] Salim Chehida, Mustapha Kamel Rahmouni: Security Requirements Analysis of Web Applications Using UML. ICWIT 2012: pp. 232-239, 2012.

[2] Azrul Hazri Jantan: An Extension of UML Modeling for Web Hypermedia Design: A Case Study, International Journal of Information and Communication Technology Research, Volume 2 No.1, pp. 69-78, January 2012.

[3] Shoichiro Fujiwara,Kazuki Munakata, Yoshiharu Maeda, Asako Katayama and Tadahiro Uehara : Test data generation for web application using a UML class diagram with OCL constraints. Innovations in System and Software Engineering, pp. 275-282, 2011.

[4] Deepak Garg, Amitava Datta and Tim French: New test case prioritization strategies for regression testing of web applications. International journal of System Assurance Engineering and Management (Oct-Dec 2012), pp. 300-309, 2012.

[5] Boni García and Juan C. Duenas: Automated Functional Testing based on the Navigation of Web Applications. Workshop on Automated Specification and Verification of Web Systems (WWV 2011), pp. 49-65, 2011.

[6] Josip Maras and Ana Petricic: Reverse engineering legacy Web applications with phpModeler. Malardalen University Software EngineeringWorkshop, 2009.

[7] Kie Homma, Satoru Izumi, Kaoru Takahashi and Atsushi Togashi:Modeling,Verification and Testing of Web Applications using Model Checker.The Institute of Electronics,Information and Communication Engineers(IEICE) Transactions on Information and Systems, Volume E94, pp. 989-999, May 2011.

[8] Jeff Offutt and Ye Wu: Modeling Presentation Layers of Web Applications for Testing. Springer's Software and Systems Modeling, Volume 9, Issue 2, pp. 257-280, April 2010.

[9] Juhan Ernits, Rivo Roo, Jonathan Jacky and Margus Veanes: Model based Testing of web applications using NModel. Springer's Testing of Software and Communications Systems. Volume 5826, pp. 211-216, 2009.

[10] Pieter Koopman, Peter Achten and Rinus Plasmeijer: Model Based Testing of Thin Client Web Applications and Navigation Input. Springer's Practical Aspects of Declarative Languages, Volume 4902, pp. 299-315, 2008.

[11] Vitor Estevao Silva Souza, Ricardo de Almeida Falbo and Giancarlo Guizzardi: A UML profile for Modeling Framework-based Web Information Systems. Proceeding of Euro American Conference of Telematics and Information Systems, 2007.

[12] D. Castelluccia, M. Mongiello, M.Ruta and R. Totaro: WAVer : A Model Checking Based Tool to Verify Web Application Design. Proceeding of the Third International Workshop on Software Verification and Validation, Electronic Notes in Theoretical Computer Science, Volume 157, Issue 1, pp. 61-76, 2006.

[13] Rohin Verma and Rajesh Bhatia: Behavior based Automated Test Case Generation for Object oriented Systems. International Journal of Computer Application, Volume 54, Issue 13, pp. 49-60, September 2012.

[14] Pieter Koopman, Rinus Plasmeijer and Peter Achten : Model-based Testing of Thin -Client Web Applications. Proceedings Formal Approaches to Testing and Runtime Verification, Volume 4262 of LNCS Springer, 2006.

[15] Jeffrey Feldstein: Model Based Testing for Java and Web Applications.http://www.sherpas.com. 2006.

[16] Dehla Sokenou: Generating Test Sequences from UML Sequence Diagrams and State Diagrams. Informatik 2006: Informatik für Menschen, pp. 236-240, 2006.

[17] Ana Cavalli, Stephane Maag,Sofia Papagiannaki and Georgios Verigakis : From UML models to automatic generated tests fir the dotLRN e-learning platform. Journal Electronic Notes in Theoretical Computer Science, Volume 116, pp. 133-144, January 2005.

[18] Hartman and K. Nagin: The AGEDIS tools for model based Testing. ISSTA '2004' Proceedings of the 2004 ACM SIGSOFT International symposium on software testing and analysis. Volume 29, issue 4,pp. 129-132, July 2004.

[19] Myers, Glenford J. The art of software testing / Glenford J. Myers; Revised and updated by Tom Badgett and Todd Thomas, with Corey Sandler.2nd ed.p.cm. ISBN 0-471-46912-2 pp 6

[20] Conallen, J.: Building Web Applications with UML. 2nd edn. AddisonWesley(2002).

[21] Rohit Kumar and Rajesh Bhatia, Interaction Diagram Based Test Cases Generations, in ObCom 2011, Part II, CCIS 270, pp. 202–211, Springer-Verlag Berlin Heidelberg 2012.