http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 10 (7): 1305-1314, 2011
ISSN 1812-5638 / DOI: 10.3923/itj.2011.1305.1314

Testing Web Applications with Web Frameset and Browser Interactions

[1,2]Bo Song and [3]Shengbo Chen
[1]College of Information Science and Technology,
Qingdao University of Science and Technology, 266061 Qingdao, China
[2]Shanghai Key Laboratory of Computer Software Evaluating and Testing, 201112 Shanghai, China
[3]School of Computer Engineering and Science, Shanghai University, 200072 Shanghai, China

Abstract: Software testing is a difficult task and testing Web applications may be more difficult, due to their characteristics. Web application is an interactive one since it came into being. How to model and test Web interactions is a challenge in software engineering. Web navigation models are useful to clarify requirements and specify implementation behaviors. Web frameset and Web browser's interactions are wildly used in today's Web applications. The behaviors of Web frameset and Web browser interactions changed the traditional Web navigation and influence the functionalities of Web applications. In this study, we took Web frameset and Web browser's interactions into account and proposed an approach to modeling the Web application's navigation and generating tests with Web framesets and browser's interactions. The construction formal semantics of structure composition of Web frameset is give out. Based on the formal semantics, the Web navigation with Web framesets is modeled. The Extended Browser Loading Model with Web frameset (EBLM) and the extended FSM is employed to formalize the navigation models with Web framesets and browser's interactions and a FSM Test-Tree (FSM-TT) is constructed using present construction algorithm. At last, test generation is given out which satisfies the corresponding coverage criteria. This study resolves the following research problems: (1) propose an approach to modeling Web applications with Web framesets and Web browser's interactions, (2) give the formal construction semantics of Web frameset and (3) formalize the navigation model and generating tests.

Key words: FSM, software modeling, test generation, test tree, navigation model

INTRODUCTION

Web applications are usually composed of front-end user interfaces in Web browser, back-end servers including web server, application server and database server etc. With prevalence of Internet and rapid development of some technologies such as distributed computing, component-based developing and Web services, Web applications have been integrated into many business critical systems and public transaction processing systems. As more and more services and information are made available over the Internet and intranets, for example, Web-based Geographical Information System (GIS) for tourism (Fajuyigbe *et al*., 2007). Web applications have become extraordinarily complex, while their correctness is often crucial to the success of businesses and organizations. Although traditional software testing is already a notoriously hard, time-consuming and expensive process, Web application testing presents even greater challenges. Therefore, a new methodology for Web application testing is required imminently to generate tests (Di Lucca and Fasolino, 2006).

The wide diffusion of Internet has produced a significant growth of the demand of Web-based applications with stricter requirements of reliability, usability, inter-operability and security. Due to market pressure and very short time-to-market, the part of modeling work is as far as possible to be simplified which causes the models of Web applications to be modeled by halves and more follow-up work is dependent on the developers. Additionally, the testing of Web-based applications is often neglected by developers, as it is considered too time-consuming and lacking a significant payoff (Hieatt and Mee, 2002). This depreciable habit affects negatively the quality of the applications and, therefore triggers the need for adequate, efficient and cost effective testing approaches for testing Web applications.

The analysis and modeling of Web applications are very difficult due to its complexity, dynamicity, heterogeneity and the diversity of its links and compositions. These new features make the modeling

Corresponding Author: Bo Song, College of Information Science and Technology, Qingdao University of Science and Technology, 266061 Qingdao, China

and testing of web-based software a challenging task in software engineering community.

Additionally, the basic Web browser features provide an adequate set of navigational facilities for Web users. In the presence of Web browser cache, for example, the users can interact not only with the Web pages but also with the Web browser itself via the use of the special buttons such as Back, Forward, Refresh or via URL rewriting. Such actions of the users may affect the overall navigations of the Web pages which can be quite sensitive to the security of the information they carry. Thus, the behavior of the Web browsers may have impact on the correctness of the Web applications: a Web application providing all correct functionality by itself may however malfunction when it is put into its supporting environment (Chen and Zhao, 2004).

Furthermore, as Web applications evolve, more and more Web framesets are wildly used to organize multiple frames and nested framesets to make the layout of some Web pages more identical and bring the development of Web applications easier. On the other hand, these make the structure of Web applications become more and more complex and also influence the Web functionalities and Web navigation. Modeling and testing Web applications should take them into account. In this study, when modeling and testing of Web applications, we consider Web framesets and Web browser's interactions.

FORMAL NOTATIONS FOR WEB FRAMESET

The FRAMESET element is a container for the FRAME element. An HTML document can contain either the FRAMESET element or the BODY element, but not both (W3C, 1999). Framesets are created on the client, loading separate pages in for each frame. In order to simplicity, in this study, we look the layout of Web frameset as two primitive types: 1. Top-bottom structure (Fig. 1a); 2. Left-right structure (Fig. 1b). Each part of Web frameset is a Web frame, i.e., Top, Bottom, Left and Right are all Web frames. All the other complicated layout structures can be constructed by combined these two primitive types using iteration. For example, the layout of Fig. 1c can be constructed by substituting Fig. 1b for the Bottom frame of Fig. 1a and the layout of Fig. 1d can be constructed by substituting Fig. 1a for the Bottom frame of Fig. 1a.

So, according to above mentioned, we can give the definition of composite structure of frameset. First of all, we give some primary definitions.

Definition 1: Composite structure of Web frameset has the following syntax given in Buckus Naur form:

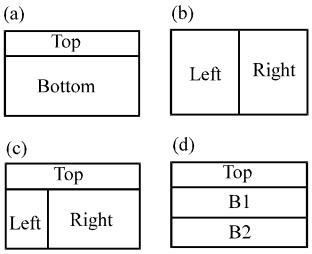

Fig. 1: The structures of frameset

$$F ::= f \mid (f \downarrow f) \mid (f \rightarrow f) \mid F \downarrow F \mid F \rightarrow F \mid$$

Where f stands for any single frame, that is to say, f is looked as a construction atom and each occurrence of F to the right of ::= stands for any already constructed structure of frameset. This constructed structure of frameset is our composite structure of Web frameset. On the other hand, from the standpoint of Mathematical Logic, F is a well-formed formula.

Definition 2: The well-formed formulas of construction logic are those which we obtain by using the construction rules below and only those, finitely many times:

Atom: Every construction atom f, g, h, … and f_1, f_2, f_3,… is a well-formed formula:

- If ϕ and φ are well-formed formulas, then so is $(\phi \downarrow \varphi)$
- If ϕ and φ are well-formed formulas, then so is $(\phi \rightarrow \varphi)$

Definition 3 (Well-formed formulas): We call formulas well-formed iff they can be deduced to be so using the definition above.

From the definitions above mentioned, the symbols "↓" and "→" are operators, we give the real meanings to them, so as they could be used to construct the composite structures of Web frameset. In this study, the notation "$\phi \downarrow \varphi$" means it is a top-bottom structure, where ϕ is on the top and φ on the bottom. And "$\phi \rightarrow \varphi$" means it is left-right structure, where ϕ is on the left and φ on the right. Consequently, we can use the formula $F ::= f \mid (f \downarrow f) \mid (f \rightarrow f) \mid F \downarrow F \mid F \rightarrow F \mid$ to formally construct the composite structures of Web frameset which we need. For example:

F = f means that the frameset F contains only one frame f.

F = f ↓ g means that the frameset F consists of two frames f and g, which is top-bottom structure where f is on the top and g on the bottom.

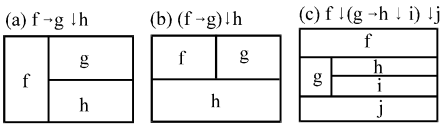

Fig. 2: Composition of frames

F = f → g means that the frameset F consists of two frames f and g, which is left-right structure where f is on the left and g on the right.

From view of structure layout, F = f → g → h is same to F = (f → g) → h and F = f → (g → h). In like manner, F = f ↓ g ↓ h is same to F = (f ↓ g) ↓ h and F = f ↓ (g ↓ h). But F = f → g ↓ h is equal to F = (f → g) ↓ h or F = f → (g ↓ h)? This confused us. In order to disambiguate this assertion, we get annoyed by a proliferation of such brackets which is why we adopt certain conventions about the binding priorities of these symbols. So in this study, we give a convention.

Convention 1: ↓ binds more tightly than →. That means the binding priority of ↓ is higher than that of →. And they are right-associative: expressions of the form f → g → h denote f → (g → h) and f ↓ g ↓ h denote f ↓ (g ↓ h).

According to the convention above, f → g ↓ h denote f → (g ↓ h), not (f → g) ↓ h, as shown in Fig. 2a and b. The composition structure as describing in Fig. 2c can be expressed as f ↓ (g → h ↓ i) ↓ j.

MOTIVATING EXAMPLE

Usually a user, while navigating through a Web application, besides following the hyperlinks to reach a target page, uses the Web browser buttons, i.e., the back, forward or refresh buttons, to redisplay some of the pages already visited along the navigation. The usage of those Web browser facilities may negatively affect the navigation (Di Lucca and Di Penta, 2003), because they might introduce some inconsistencies or violate some functional/ non-functional Web application requirements. Now-a-days, a great number of Web framesets are used in Web applications which facilitate the development of Web applications but make the Web navigations more intricate. In order to model and test a Web application with Web frameset, we must take Web frameset and Web browser interactions into account.

Consider a small Web application that we have developed to illustrate our approach, the Student Grade Retrieving System (SGRS) (Fig. 3). A user type the Web application's URL in Address field of Web browser. After the Enter key is pressed, Web server will return the p-MainPage (P0) to the user. If the user presses the link

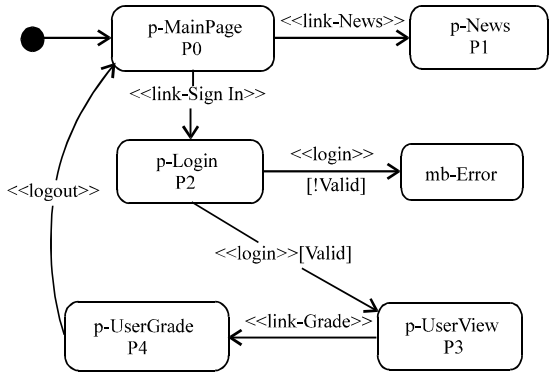

Fig. 3: Page flows of SGRS

News (Fig. 3), he/she will enter the p-News page (P1). And after pressing Sign in hyperlink (Fig. 3) in p-MainPage, the user will enter the p-Login page (P2). The user enters the username and password and presses the login button. Upon this pressing, the username and password are sent to the Web server for authentication. If the username and password are correct, a private page p-User view (P3) is loaded and displayed in Web browser to the user. After the user presses the link Grade (link-Grade) in page P3 and the p-User Grade page (P4) will be displayed. If the user presses the logout link, the p-MainPage (P0) is loaded and displayed again in the Web browser.

In practice, in order to keep the appearance of a Web application identical and attractive, some layout techniques are employed and the most popular one is Web frameset. The pure page flows without any layout technique is seldom seen in today's Web applications. For the sake of modeling and testing of Web applications, we add the Web framesets into our SGRS. In fact, the usual Web application is indeed with framesets as shown in Fig. 4, where the prefix link denotes hyperlink, p denotes Web page, mb denotes message box, public means that the frameset is public and private means the frameset is private which contains private page which is related to a certain user, not to public users. In Fig. 4, all the framesets have a self-transition, e.g., the first frameset when click the link mainpage, it will return to itself. Here have two structures of Web frameset in this example, using formal notation: top ↓ bottom and top ↓ (left → right).

And for the time being, we don't take the Web browser's interactions into account. We will consider the browser's interactions in next section. Figure 4 is the model of SGRS with Web Framesets. In order to facilitate test generation, we employ Finite State Machine (FSM) to illustrate our model as shown in Fig. 5. We give some items for short, for example, mb-Message box; 1-link-Sign

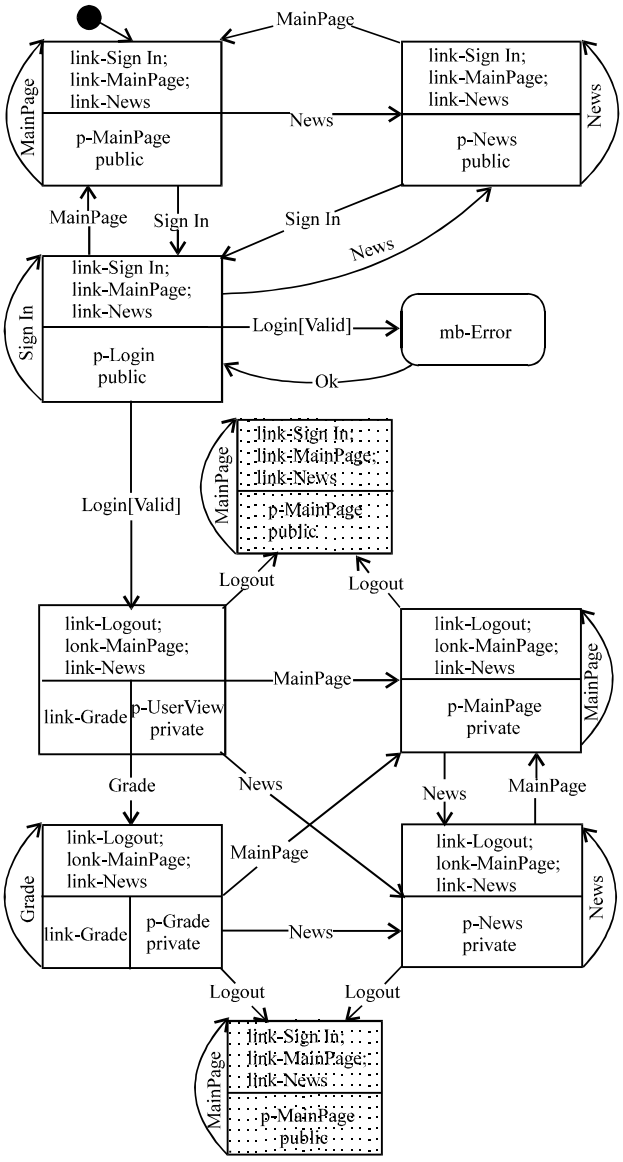

Fig. 4: Model of SGRS with Web Framesets

In; 2-link-mainpage; 3-link-News; 4-link-logout. F0 stands for a frameset which is a top-bottom structure where in its top frame there have three links: Sign In, mainpage, News and on its bottom frame is loaded a Web page MainPage. This frameset is public. So, in a formal method, we described it as:

$$A \downarrow B$$
$$A = (1, 2, 3)$$
$$B = (\text{p-MainPage})$$
$$\text{Public}$$

F3 represents a combination Frameset same to Fig. 1c where three links: Logout, MainPage, News exist on the top frame, one link: Grade on the its left frame and on its right frame is loaded a UserView page:

$$A \downarrow B \rightarrow C$$
$$A = (4, 2, 3)$$
$$B = (\text{link-Grade})$$
$$C = (\text{p-UserView})$$
$$\text{Private}$$

Additionally, the description of other framesets can follow that of F0 and F3. Note, in order to reduce the number of interweaved lines and clean the graph, we add the two dotted frameset which is the initial frameset of SGRS.

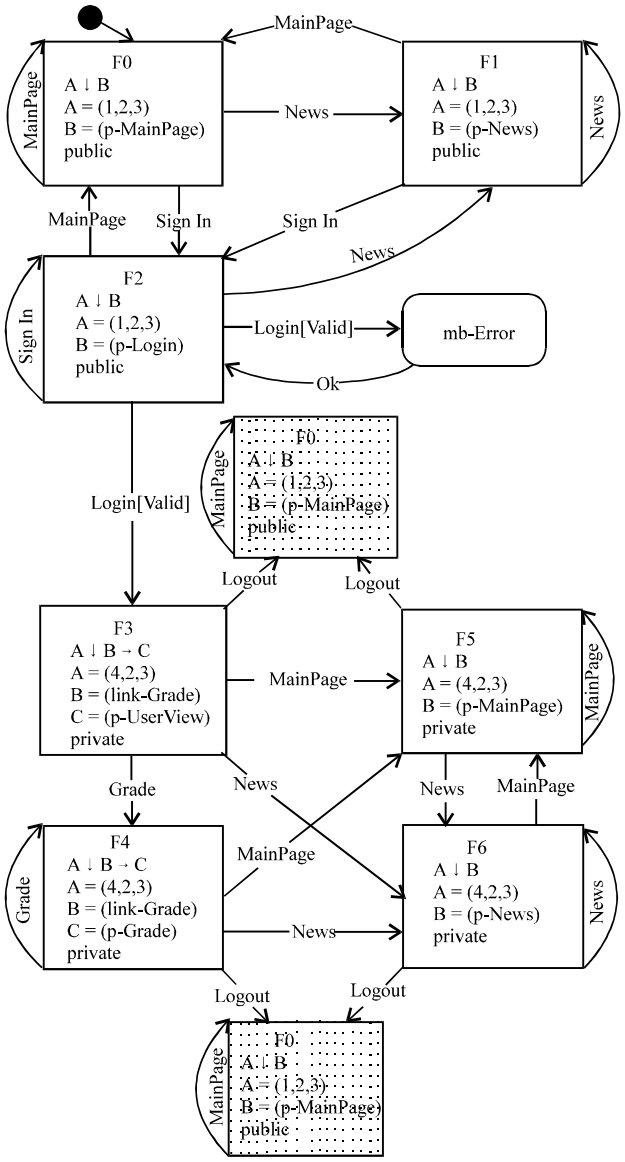

Fig. 5: FSMs of SGRS with Web Framesets TF: TopFrame; BF: BottomFame; LF: LeftFrame; RF: RightFrame; mb: Message box; 1: Link-Sign In; 2: Link-mainPage; 3: Link-News; 4: Link-logout

CHARACTERIZING FRAMESETS WITH BROWSER'S INTERACTIONS

Regarding the Web, page hyperlinks behave in different ways. The most frequent ones are replacement page hyperlinks where the destination page replaces the source one (Nielsen, 1995) and the Back button of Web browser is enabled. The second ones are that when users click the link, the destination page will display in a new window of Web browser, not replace the source one and the Back button is disabled. The third ones are that when users place the mouse icon on the link and right click and choose "Open in new window", whether the link belongs to the first ones or second ones, the destination page will display in a new window and the Back button is disabled. In this paper, because there exists Web frameset which consists of Web frames, page hyperlinks behave maybe in another ways except for the three ways above mentioned. For example, the destination page will display in another frame and the Back button of Web browser is enabled. If the Back button is enabled, we can click the Back button to revisit the previous pages. And if the Forward button of browser is enabled, we also can press it to revisit the visited pages.

Consequently, we can model the Web application with Web frameset and browser interactions by a transition system, where each state is defined by the frame which consists of Web page displayed or some links and the state of the buttons or a message box, while the user actions on clicking links or Web browser buttons determine the state transitions. As regards the Back and Forward buttons, the following states of the browser buttons can be identified (Di Lucca and Di Penta, 2003):

BDFD: Back Disabled and Forward Disabled
BDFE: Back Disabled and Forward Enabled
BEFD: Back Enabled and Forward Disabled
BEFE: Back Enabled and Forward Enabled

The features of Web browsers and Web frameset influence the navigation in Web applications negatively. Consequently, when modeling a Web application to facilitate to generating test, we must take them into account.

MODELING WEB NAVIGATIONS

It assumes that all interactions with the Web browser take place before the server session expires. Existing literatures about Web navigation models with Web browser interactions addressed only Web page as a navigation entity and didn't take the Web frameset into account. As we all know, Web frameset is used very wildly in modern Web applications, which also influences the Web navigation. Therefore, attention should be given to Web frameset when we model and test Web applications.

Definition 4 (Extended Browser Loading Model with web frameset, EBLM): EBLM is a quadruple BLM = (Pre, Br, Cur, Post), to modeling each Frameset and the behaviors of the Web browsers which loaded the frameset, namely to modeling each screen of client sides, where:

- Pre is a state of precursor frameset, {-}indicates that its precursor frameset is absent
- Br is the state of the browser behaviors, Br∈(00, 10, 01, 11), where, 00 stands for BDFD, 10 stands for BEFD, 01 stands for BDFE and 11 stands for BEFE
- Cur is a state of current frameset
- Post is a state of successor frameset, {-} indicates that the successor frameset is absent

Firstly, make use of the EBLM, we can model the states of each page at different stage. If the current frameset Cur of one EBLM is Fa (Cur = Fa) and its successor frameset Post is Fb, while the precursor page Pre of the other EBLM is Fb (Pre = Fb) and its current frameset Cur of one EBLM is Fc (Cur = Fc), well then there exits a navigation association between the EBLMs of two framesets Fa and Fc: Fa→Fc. In a like manner, in the end, by combining the exiting navigation associations, we can obtain the Web navigation models:

Table 1: The EBLM (s) of each frameset

Frameset	EBLM (s)
F0	(-, 00, F0, -), (-, 10, F0, F1), (-, 10, F0, F2), (F2, 10, F0, -), (F1, 10, F0, -), (F2, 11, F0, F1), (F2, 11, F0, F2), (F1, 11, F0, F1), (F1, 11, F0, F2);
F1	(F0, 10, F1, -), (F2, 10, F1, -), (F0, 11, F1, F0), (F2, 11, F1, F0), (F2, 11, F1, F2), (F0, 11, F1, F2);
F2	(F0, 10, F2, -), (F1, 10, F2, -), (mb, 10, F2, F0), (mb, 10, F2, F1), (F1, 10, F2, mb), (F0, 10, F2, mb), (F3, 11, F2, F0), (F3, 11, F2, F1), (F0, 11, F2, F0), (F0, 11, F2, F1), (F1, 11, F2, F1), (F1, 11, F2, F0);
F3	(F2, 00, F3, -), (F2, 01, F3, F4), (F2, 00, F3, F5), (F2, 00, F3, F6);
F4	(F3, 10, F4, -), (F3, 10, F4, F6), (F3, 10, F4, F0), (F3, 10, F4, F5);
F5	(F3, 00, F5, -), (F4, 00, F5, -), (F6, 00, F5, -), (F3, 00, F5, F0), (F4, 00, F5, F0), (F6, 00, F5, F0), (F6,11, F5, F6), (F3, 01, F5, F6), (F4, 01, F5, F6);
F6	(F3, 00, F6, -), (F4, 00, F6, -) , (F5, 10, F6, -), (F3, 00, F6, F0), (F4, 00, F6, F0), (F5, 00, F6, F0), (F3, 00, F6, F5), (F4, 01, F6, F5), (F5, 11, F6, F5);

- (-, 00, F0, -) indicates that the user enters the frameset F0, without any link operations. So, in this case, Pre and Post are all empty. 00 means BDFD, namely, Back Disabled and Forward Disabled
- (F2, 10, F3, F1) indicates that current frameset is F3 and the successor frameset is F1 and the precursor frameset is F2, 10 means BEFD, namely, Back Enabled and Forward Disabled and the user can press the Back button of Web Browser to go forward to the frameset F2

By the same logic, we can get the EBLMs of all framesets, as shown in Table 1.

Based on Table 1 above, by elaborating the EBLM (s) of each page, we can gain the navigation of the Web application with Web framesets and the browser's interactions (Fig. 6). In order to simplicity, all link actions, such as Sign In, MainPage, News, are described as "link" in Fig. 6.

FORMALIZING NAVIGATION MODEL

The state transition system is employed to formally model the Web navigation model with Web framesets and Web browser's interactions. It is a type of state transition graph consisting of nodes representing

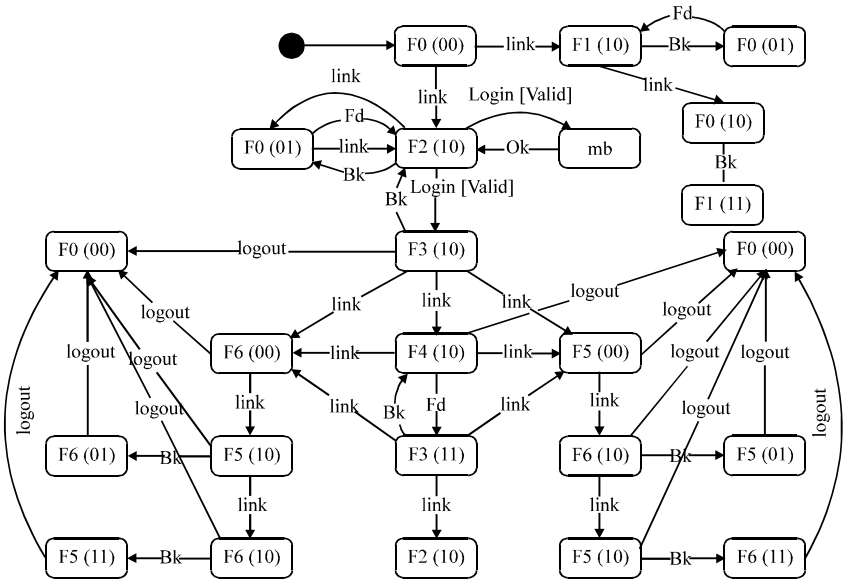

Fig. 6: Model of SGRS with Web framesets and browser's interactions Bk-Back; Fd-Forward

the reachable states of the system and edges representing the state transitions.

Definition 5: The navigation model of a Web application is described as an extended FSM EFSM = (S, S_0, Act, Tr) where:

- S is a finite set of states. Each state consists of one Web Frameset which consists of frames and the behaviors of Web Browser which loaded the frameset or a message box. For example, F0 (00) belongs to S, F0 (00) $\in$ S; a message box mbError belongs to S mb Error $\in$ S
- $S_0 \subseteq S$ is the set of initial states
- Act is a finite set of actions; for example, clicking the links or the buttons on Web frameset, or clicking the buttons on a message box, or pressing the buttons of Web browsers
- $Tr \subseteq S \times Act \times S$ is a finite set of labelled transitions, for every state $s \in S$ there is a state $s' \in S$ such that Tr (s, s')

The Web navigation model with Web framesets and browser's interactions can be formally described by the EFSM.

GENERATING TESTS

The structure of a Web application is often complicated and difficult to maintain. Especially, a lot of Web framesets are used to aggravate this situation. To ensure whether or not the Web framesets or Web page flows and the behaviors of Web browser are appropriate and meet the requirements, a navigation model of a Web application is employed. The navigation model intuitively reflects the relationship among Web framesets or Web pages and the behaviors of Web browser in a Web application.

But the navigation model of SGRS with Web frameset and Web browser's interactions (Fig. 4) has many more cycles, it is hard to use to generated tests without redundancy effectively and directly. And as is well known, one of important characteristics for a tree structure is branching, which provides a means to represent and pick out the set of test sequences with no redundancy (Zeng *et al*., 2007). We refer to such trees as test-trees. For a test-tree, each path from root to a leaf corresponds to a test sequence.

The navigation model can be used to generate tests. But, it is a directed graphic structure (Fig. 6) which brings the problems that we often cannot verdict the termination of a path and the path may be cyclic, as will complicate the testing process, thus, driving the testing process using the EFSM is difficult and unwieldy. And it will generate a set of test sequences with redundancy. Therefore, A FSM-TT (FSM Test-Tree) which is based on the EFSM is used. From the FSM-TT, shorter test sequences can be generated than from the EFSM without the loss of state and transition coverage. A FSM-TT is a spanning tree constructed from the EFSM and the

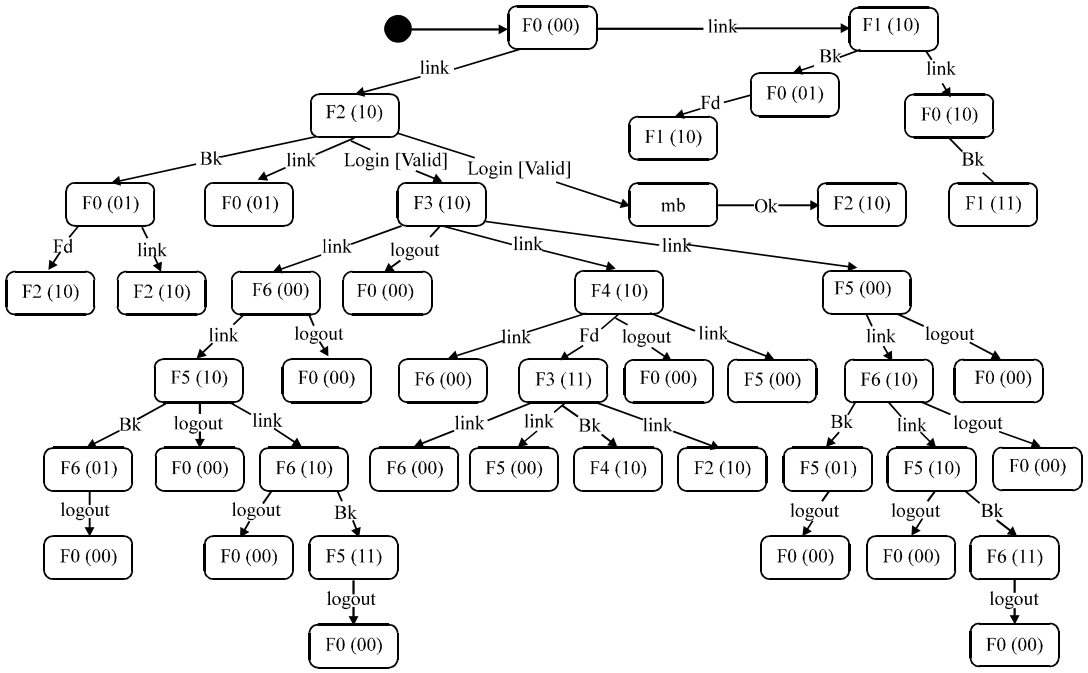

Fig. 7: FSM-TT

construction algorithm is given by Song and Miao (2009). In the algorithm, we employ two tables OPEN and CLOSED. OPEN is a table that contains the frameset (node) identifiers, which are unmarked; CLOSED is a table that contains the frameset identifiers, which have already been marked. OPEN and CLOSED are both empty initially. Note that, frameset identifiers and their copies are allowed to be in OPEN at the same time.

Suppose that there are n transitions in an EFSM, we can derive a FSM-TT in (n + 1) steps using the given algorithm. Figure 7 shows the FSM-TT derived from the EFSM in Fig. 6, where the nodes and edges represent the states and transitions in the EFSM respectively.

A test-tree is intended to construct an efficient set of test sequences each of which corresponds to a path from root to some leaf node, thus duplication and inclusion of test sequences are avoided. Figure 7 shows the corresponding FSM test-tree for the navigation models in Fig. 6. So, by this means 24 test sequences with total 108 transitions (3+3+2+3+6+5+6+7+4+4+5+5+5+5+4+4+4+6+6+7+5+3+3+3 = 108) are generated as listed below, which obviously satisfies states coverage and transitions coverage.

- F0 (00)-«link»-F2 (10)-«Bk»-F0 (01)-«Fd»-F2 (10)
- F0 (00)-«link»-F2 (10)-«Bk»-F0 (01)-«link»-F2 (10)
- F0 (00)-«link»-F2 (10)-«link»-F0 (01)
- F0(00)-«link»-F2(10)-«Login[!valid]»-mb-«Ok»-F2(10)
- F0 (00)-«link»-F1 (10)-«Bk»-F0 (01)-«Fd»-F1 (10)
- F0 (00)-«link»-F1 (10)-«link»-F0 (01)-«Bk»-F1 (11)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F6 (00)-«link»-F5 (10)-«Bk»-F6 (01)-«logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F6 (00) -«link»-F5 (10)-«logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F6 (00)-«link»-F5 (10)-«link»-F6 (10)-«logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F6 (00)-«link»-F5 (10)-«link»-F6 (10)-«Bk»-F5 (11)-«logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F6 (00)-«logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F4 (10)-«link»-F6 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F4 (10)-«Fd»-F3 (11)-«link»-F6 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F4 (10)-«Fd»-F3 (11)-«link»-F5 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F4 (10)-«Fd»-F3 (11)-«Bk»-F4 (10)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F4 (10)-«Fd»-F3 (11)-«link»-F2 (10)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F4 (10) -«link»-F5 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F5 (00)-«link»-F6 (10)-«Bk»-F5 (01)-«logout»-F0 (00)

- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F5 (00)-«link»-F6 (10)-«link»-F5 (10)-«logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F5 (00)-«link»-F6(10)-«link»-F5(10)-«Bk»-F6(11) «logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F5 (00)-«link»-F6 (10)-«logout»-F0 (00)
- F0(00)-«link»-F2(10)-«Login[valid]»-F3(10)-«link»-F5 (00)-«logout»-F0 (00)

RELATED WORK

A number of test methods or test techniques have been already proposed, each of which has different origins and pursuing different goals. Layachi-Badri (2006) proposed a method for structural test of a data basis oriented object after phase of conception and adapted a strategy of test of the applications oriented object was to the data base oriented object. Mustafa *et al.* (2007) considered that the complex and dynamic computing environment, complex user requirements, new features and constraints made the testing of web-based software a challenging task and reviewed the techniques for the testing of web based applications and gave a strategy to use the testing techniques meeting the challenges of testing web-based software. Kosindrdecha and Daengdej (2010) introduced a new test case generation process with a requirement prioritization method and propose a new effective test generation method. Roongruangsuwan and Daengdej (2010) proposed a new classification of test case prioritization techniques, introduce a new continuous test case prioritization process and propose a new test case prioritization method along with a practical set of weight factors.

As for testing Web application for dealing with the unique characteristics of Web, navigation models such as Turine *et al.* (1997), Leung *et al.* (2000) and Oliveira *et al.* (2001) use statecharts notations. They model Web navigation, Web elements and the interactions among them when the user traverses the Web application. However, the Web browser interactions and test generation are not concerned.

Andrews *et al.* (2004) proposed a method for deriving tests from FSMs. Test sequences are generated based on FSMs and they try to use input constraints to restrict the state space explosion.

This method also does not care the Web browser interactions.

Dargham and Nasrawi (2006) presented the model to model the behavior of Web applications using FSMs after extending FSM's constructors to fit its needs. But the web browser interactions and test generation are also not taken into account.

To the best of our knowledge, the issues in modeling Web browser interactions have seldom been addressed with the exception of Chen and Zhao (2004), Di Lucca and Di Penta (2003), Song and Miao (2009), Miao *et al.* (2007) and Song *et al.* (2008). Di Lucca and Di Penta (2003) proposed an approach to integrate existing testing techniques with a state-based testing devoted to discover possible inconsistencies caused by interactions with Web browser buttons. However, the coverage and cacheability are not considered. Miao *et al.* (2007) took special care on Web browser interactions during the user's traversal within hypermedia space in order to specify possible inconsistencies between Web browser interfaces and user cognitions. GFSMs (Guarded Finite State Machines), which are augmented FSMs are employed as a tool to model Web browser interactions. Chen and Zhao (2004) provided a definition of Web model in terms of labeled transition systems by incorporating the abstract behavioral model of the Web browsers. However, the test generation based on the presented model is not given out. Song and Miao (2009) and Song *et al.* (2008) paid special care on Web browser's interactions and proposed an approach to modeling on-the-fly navigation models and test generation with Web browser interactions.

In this study, the Web application is modeled. We not only take the Web browser's interactions into account, but also consider the Web framesets which all influence the Web navigation. Additionally, test generation is accomplished.

CONCLUSION AND FUTURE WORK

The behaviors of Web frameset and Web browser interactions influence the functionalities and navigations of Web applications. They should be taken into account when to model and test Web applications. We pay special care on Web frameset and Web browser interactions in the paper and propose an approach to modeling the Web navigation with Web framesets and Web browser interactions. The extended FSM (EFSM) is employed to formalize the navigation models. A FSM-TT (FSM Test-Tree) which is based on the EFSM is constructed by the way of the algorithm we present. Based on FSM-TT, the test generation is give out which satisfy the coverage criteria, such as state coverage, transition coverage.

There are several research aspects for future work. First, we can instantiate the test cases after we got them and then we can execute these instantiated test cases on

the aimed Web application to find software faults. Second, since a Web application is strictly interwoven to its running environment and the running environment mainly influences the non-functional requirements of a Web application, such as performance, stability and compatibility etc. So we can give abstract mechanisms to describe the abstract behavior of the running environment.

ACKNOWLEDGMENT

This study is supported by National Natural Science Foundation of China (NSFC) under grant No. 60970007 and 61073050, the National Grand Basic Research Program (973 Program) of China (2007CB310800), the Natural Science Foundation of Shanghai Municipality of China (09ZR1412100), Shanghai Leading Academic Discipline Project (J50103), Key Laboratory of Science and Technology Commission of Shanghai Municipality (09DZ2272600) and this is a extended and revised paper of CiSE2009).

REFERENCES

Andrews, A., J. Offutt and R. Alexander, 2004. Testing web applications by modeling with FSMs. Software Syst. Modeling, 4: 326-345.

Chen, J. and X. Zhao, 2004. Formal models for web navigations with session control and browser cache. Proc. Int. Conf. Formal Methods Software Eng., 3308: 46-60.

Dargham, J. and S.A. Nasrawi, 2006. FSM behavioral modeling approach for hypermedia web applications: FBM-HWA approach. Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services, Feb. 19-25, Guadeloupe, French Caribbean, pp: 199-204.

Di Lucca, G.A. and A.R. Fasolino, 2006. Testing Web-based applications: The state of the art and future trends. Inform. Software Technol., 48: 1172-1186.

Di Lucca, G.A. and M. Di Penta, 2003. Considering browser interaction in web application testing. Proceedings of the 5th IEEE International Workshop on Web Site Evolution, Sept. 22, IEEE Press, New York, USA., pp: 74-81.

Fajuyigbe, O., V.F. Balogun and O.M. Obembe, 2007. Web-based Geographical Information System (GIS) for tourism in Oyo state, Nigeria. Inform. Technol. J., 6: 613-622.

Hieatt, E. and R. Mee, 2002. Going faster: Testing the web applications. J. IEEE Software, l: 60-65.

Kosindrdecha, N. and J. Daengdej, 2010. A test case generation process and technique. J. Software Eng., 4: 265-287.

Layachi-Badri, S., 2006. Structural test of a data basis oriented object after phase of conception. Inform. Technol. J., 5: 753-758.

Leung, K.R.P.H., L.C.K. Hui, S.M. Yiu and R.W.M. Tang, 2000. Modeling web navigation by statechart. Proceedings of the 24th International Computer Software and Applications Conference, Oct. 25-27, Taipei, Taiwan, pp: 41-47.

Miao, H., Z. Qian and T. He, 2007. Modeling web browser interactions using FSM. Proceedings of the 2nd IEEE Asia-Pacific Service Computing Conference, Dec. 11-14, Tsukuba, Japan, pp: 211-217.

Mustafa, G., A.A. Shah, K.H. Asif and A. Ali, 2007. A strategy for testing of web based software. Inform. Technol. J., 6: 74-81.

Nielsen, J., 1995. Multimedia and Hypertext: The Internet and Beyond. 2nd Edn., Morgan Kaufmann, San Francisco.

Oliveira, M.C.F., M.A.S. Turine and P.C. Masiero, 2001. A statechart-based model for hypermedia applications. ACM Trans. Inf. Syst., 19: 28-52.

Roongruangsuwan, S. and J. Daengdej, 2010. A test case prioritization method with practical weight factors. J. Software Eng., 4: 193-214.

Song, B. and H.K. Miao, 2009. Modeling web applications and generating tests: A combination and interactions-guided approach. Proceedings of the 3rd IEEE International Symposium on Theoretical Aspects of Software Engineering, July 29-31, Tianjin, China, pp: 174-181.

Song, B., H. Miao and S. Chen, 2008. Modeling web browser interactions and generating tests. Proceedings of the 4th International Conference on Computational Intelligence and Security, Dec. 13-17, Suzhou, China, pp: 399-404.

Turine, M.A.S., M.C.F. Oliveira and P.C. Masiero, 1997. A navigation-oriented hypertext model based on statecharts. Proceedings of the 8th ACM Conference on Hypertext, April 6-11, Southampton, UK., pp: 102-111.

W3C, 1999. HTML 4.01 Specification. http://www.w3.org/TR/html401/

Zeng, H., H. Miao and J. Liu, 2007. Specification-based test generation and optimization using model checking. Proceedings of the First Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering, June 6-8, Shanghai, China, pp: 349-355.