AGENT BASED TESTING TOOL FOR EVENT DRIVEN SOFTWARE

Shashank Joshi*, Shital Pawar**

* (Department of Computer Engineering, Bharati Vidyapeeth Deemed University, pune, India)
** (Department of Computer Engineering, Bharati Vidyapeeth Deemed University, pune, India)

ABSTRACT:
An event driven software is the class of the software which changes its state with respect to occurrence of sequence of events. The most common examples of this Event Driven Software are GUI and web applications. Therefore it's challenging to test such Event Driven Software. The main goal of the software testing is to uncover the errors. There are various tools are available for automated GUI testing and web application testing. Here our specific aim is to develop a testing tool for testing both GUI and Web Applications together. The web application is built using asp, jsp and GUI is built through the java technology. So our testing tool is applicable to web applications of asp and jsp and swing based GUI applications. By using this testing tool we can also show the different faults present on the web pages in the graphical format to display the accuracy of the web pages.

KEYWORDS: action attribute, action listener, asp page testing, GUI components, jsp page testing, swing based GUI testing, test cases, testing tool.

1. INTRODUCTION:

In this paper we have presented two approaches to test GUI. One is for swing based GUI and another is for web application GUI such as html, jsp, asp pages. There are lots of tools available for GUI testing such as FEST, ABBOT etc. To test the GUI using these tools we have to write test cases for each page and these tools use test scripts to check against the results of GUI components. For example in fest tool we have to write test case class for each swing based frame and then it checks the GUI behavior against each test case. So each tool requires additional work to do. In our testing tool we are overcoming this drawback. In our tool we are checking the developers mistakes which he has done while coding but which are not the programming errors at all. Hence once the developer does the coding and program complies correctly then our tool takes this page as input and finding out the bugs in

that page dynamically without writing any test cases. If developer added some components on the frame then we are checking whether the component has action listener or not and also we are checking that whether the components have same name or not.
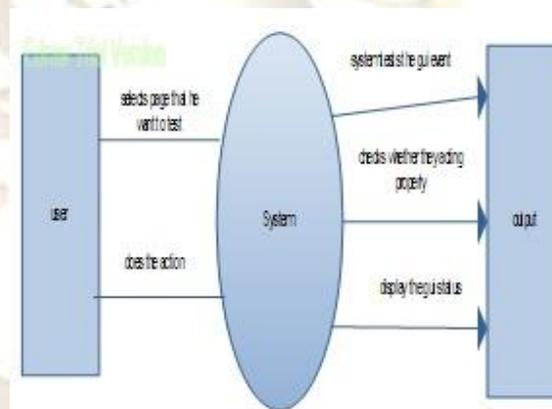The following diagram shows the system architecture:

Fig: System architecture

2. WEB APPLICATION TESTING:

In our testing tool user first select the web page which he wants to test from web application folder. After selecting the page our testing tool separates out the html, java and script code from the page. The user tests the code by clicking the test next button. After that our testing tool checks the all hyperlinks on that page, name of hyperlinks, if there is any form tag in the field then we checks its action attribute values. We store all available web pages in web application folder. If all hyperlinks which the developer provides on the web pages are available in the web application folder then there is no fault. That means there are no any fake hyperlinks or pages. After that we are checking the name given to the hyperlink. If there are same name for two different hyperlinks this will create ambiguity problem. So we are checking this problem also. If there is any such

mistake then we are finding out and displaying this fault.

Next we are checking whether developer has given same hyperlink for different names. That will also create problem. For example for login and home if developer has given the same hyperlink i.e. login.jsp then after clicking on home and login the page will redirect to the login.jsp Now we are checking the form attribute value. If the value specified by the developer in action Attributes is not present in the web application folder then this will create problem. So we are also checking for this problem.

Next we are checking whether developer has given same hyperlink for different names. That will also create problem. For example for login and home if developer has given the same hyperlink i.e. login.jsp then after clicking on home and login the page will redirect to the login.jsp Now we are checking the form attribute value. If the value specified by the developer in action Attributes is not present in the web application folder then this will create problem. So we are also checking for this problem.

2.1 jsp page testing results:

Here we are testing jsp page for example AlreadyLoan.jsp.
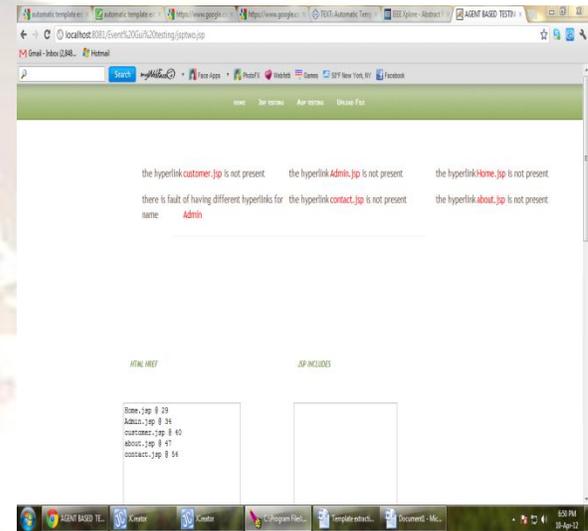
This is AlreadyLoan.jsp page.

In above page there are two types of faults. These faults are:

1) Page contains some hyperlinks which are not present in our application. So the result page displays those faults for example "the hyperlink customer.jsp is not present".

2) Page has   same names for two different hyperlinks.

For example the name Admin is given two different hyperlinks. So result page displays the error message as "there is fault having same name Admin for different hyperlinks".
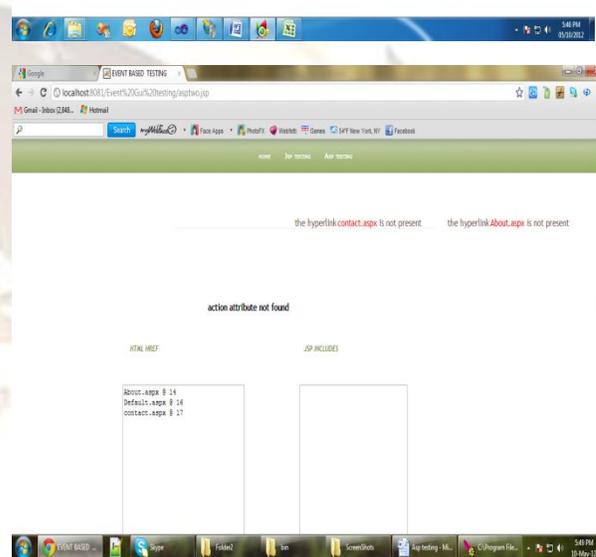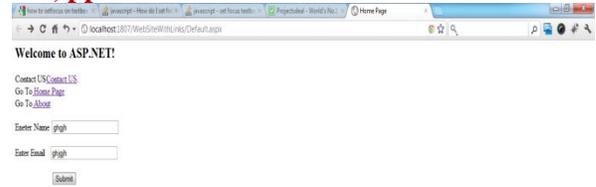
Result page:

In the web application we write action method in the page to denote where the request will be forwarded after submitting the form. So when developer forgot to write the action tag attribute we are checking the code and finding out if code contains action or not. We are also checking that if GUI contains any similar component names. If GUI contains two buttons with the same name, then it will create ambiguity for user to perform further action.

Now we are dealing with Action Attribute error. Following page contains this type of error.

Now we are testing page admin_login.jsp.

This page contains simple form in which user enters his name and password and   clicks the signin button on clicking on signin button the page has to redirect to some another page. But due to missing or unavailability of action attribute page the page will not redirect to another page. So that result page displays the error message as "action attribute is not present" as follows:

Result Page:

2.2 asp page testing:

Above page contains three hyperlinks and one form field.

Above snapshot shows the result of Default.aspx.

Among three hyperlinks one hyperlink for home is present and the action attribute of form field is also not present.
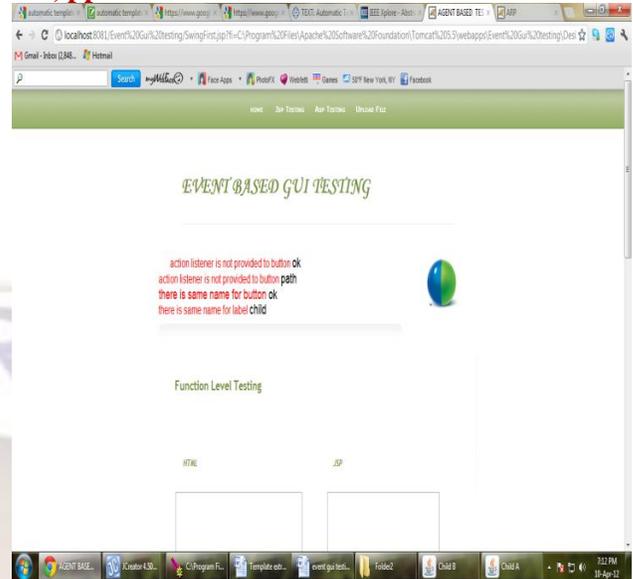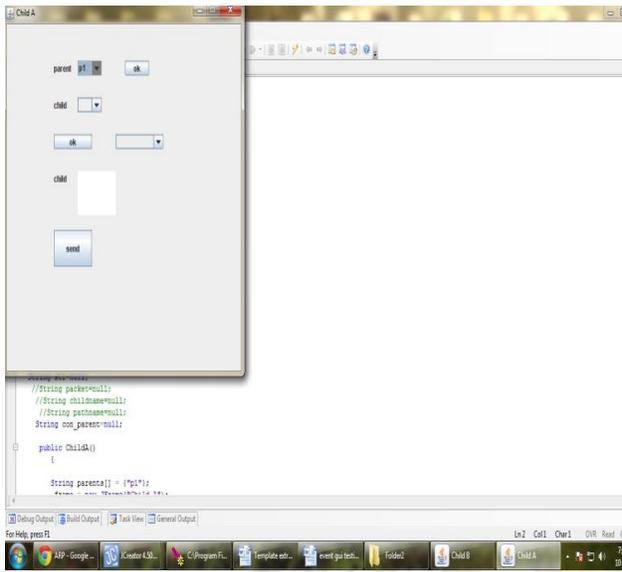
3. SWING BASED GUI TESTING:

In this we are testing swing based application. Each swing Application has window and each window contains the number of components and each component generates the events. So in our system we are checking whether each component generates specified event or not. For example on clicking on submit button the new window will be opened, this is the requirement. Then we are checking that after clicking on the submit button new window is opening or not if opened then there is no fault in the system, if not opened then there is fault in the system. We are checking the event handler of each component.

In swing based GUI testing we are testing GUI components such as button, label, and their names also whether the action Listener is provided or not.

For example: Child A.java

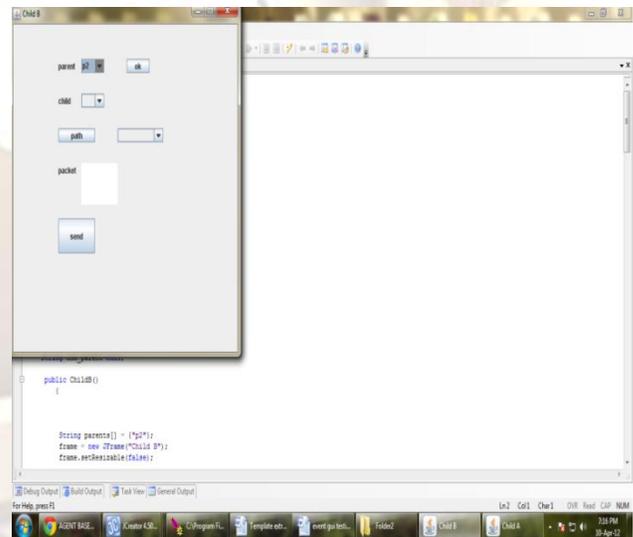On the above page ChildA.java. There are following faults:

1) The childA frame contains same name for labels child.

2) There is same name for button Ok.

3) There is no action Listener provided to button ok that means when you clicks on button ok then no action will be happen.

 The result page will display all those error with error messages as follows.
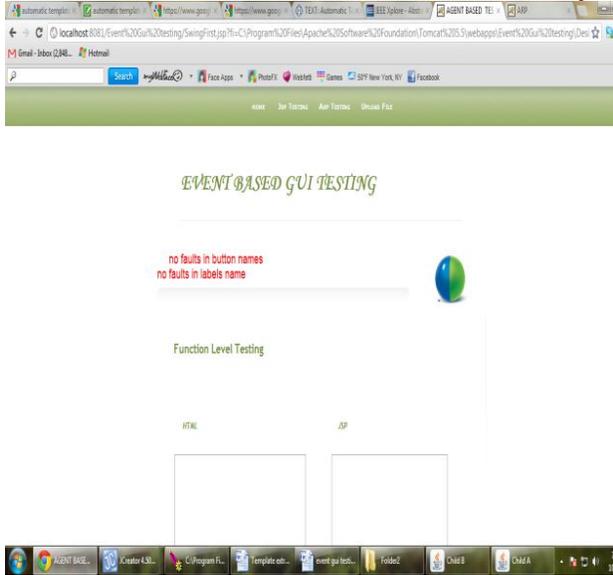
Result Page:

Now we are testing ChildB.java

 This swing Frame contains no faults in label or button name and action Listener is provided to all swing components so when we click on any button the action will be generated.

Result page:

4. CONCLUSION:

Here we are developing a testing tool named as agent based testing tool. We can use this tool for testing both GUI and web based applications together. So our testing tool can be used to test web applications of html asp and jsp also to test swing based GUI applications. Our tool checks whether any fault is present on web page or GUI. There is an upload function is provided to upload the new file into the input folder as input for the testing tool. Here we have to select the page as an input from this folder then our tool which will checks all the faults present on that page and informs the developer about the faults present on that page which he has developed. We can show the faults present on the page in the graphical format to display the accuracy of that web page.

5. REFERENCES:

[1] Bryce RC, Sampath S, Memon A (2011) Developing a single model and test prioritization strategies for event-driven software. IEEE Transactions on Software Engineering 37(1):48–64.

[2] Andrea Adamoli , Dmitrijs Zaparanuks , Milan Jovic , Matthias Hauswirth , "Automated GUI performance testing", Software Quality Journal,Apr 2011.

[3] Xun Yuan, Myra B. Cohen, Atif M Memon, GUI Interaction Testing: Incorporating Event Context, IEEE Transaction on Software Engineering Rothermel G, Untch RH, Chu C, Harrold MJ (2001) Prioritizing test cases for regression testing. IEEE Trans Softw Eng 27(10): 929–948

[4] Srikanth H, Williams L, Osborne J (2005) System test case prioritization of new and regression test cases. In: International Symposium on empirical software engg. pp 64–73.

[5] Li Z, Harman M, Hierons RM (2007) Search algorithms for regression test case prioritization. In: IEEE transactions on software engineering. IEEE Computer Society Press, Washington, pp 225–237.

[6] S. Sampath, S. Sprenkle, E. Gibson, L. Pollock, and A. S. Greenwald, "Applying concept analysis to user-session-based testing of web applications," IEEE Transactions on Software Engineering, vol. 33, no. 10, pp. 643–658, Oct. 2007.

[7] S. Sampath, V. Mihaylov, A. Souter, and L. Pollock, "Composing a framework to automate testing of operational web-based software," in the Intl. Conference on Software Maintenance. IEEE Computer Society, Sep. 2004, pp. 104–113.

[8] Ricca, F., and P. Tonella. Analysis and Testing of Web Applications. In Proceedings of the 23rd International Conference on Software Engineering. Toronto, Ontario, Canada, pp. 25-34, 2001.

[9] Xun Yuan, Atif M. Memon, Generating Event Sequence-Based Test Cases Using GUI Runtime State Feedback , VOL. 36, NO. 1, Feb 2010

[10] A. M. Memon and Q. Xie, "Studying the fault-detection effectiveness of GUI test cases for rapidly evolving software," IEEE Transactions on Software Engineering, vol. 31, no. 10, pp. 884–896, 2005