ISSN: 0975-833X

Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 9, Issue, 06, pp.51827-51829, June, 2017

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

RESEARCH ARTICLE

LITERATURE REVIEW ON MODEL BASED SOFTWARE TESTING TECHNIQUES

*Supriya S. Patil and Prof. Pramod A Jadhav

Information Technology, Bharati Vidyapeeth University College of Engineering, Pune (INDIA)

ARTICLE INFO

Article History:
Received 25th March, 2017
Received in revised form
09th April, 2017
Accepted 10th May, 2017
Published online 20th June, 2017

Key words:
Model Based Testing,
Extended Finite State Machine,
UML sequence,
Coverage and event based testing.

ABSTRACT

Testing of Software is an important phase in the SDLC. This phase not only help to check the quality of the software. This process of testing is divided in two categories, one is Model Based technique and other is code driven testing. Code driven testing is based on the testing complete code (each and every line). Code driven testing is based on a data flow and control flow that is flow is sequential. In model, based testing approach focus is on executable module not on every line of code; reason of this approach is to support third party components, APIs, modules. One need not to be expert in all domains, the tester will concentrate on the functionality of individual component not on the details of a code used in that component. In this paper the combination of extended finite state machine and sequence diagram is discussed.

Citation: ***Supriya S. Patil and Pramod A Jadhav, 2017.*** **"**Literature review on model based software testing techniques", *International Journal of Current Research*, 9, (06), 51827-51829.

INTRODUCTION

Quality of Software is checked with software testing is an important phase of software development life cycle. This phase is helps to check the whether software is implemented according to the requirement specified by the user or not. This can be done by following number of testing mechanisms; ultimate form of these methods is to generate test cases. These tests cases will check all components or functionalities. All parameters associated with these components or functionalities should meet standards; these standards are expected to reach the benchmark. Testing techniques used depends upon type of software development life cycle adopted, that is waterfall model, spiral model etc. The type of testing technique also depends upon the stage at which it is emphasized, here stage indicate stage of software development life cycle. For example, in some case testing is applicable for all phases, in some cases it is restricted for some selected stages.

Literature Survey

This literature survey is made to make clear understanding of testing methodologies discussed in the paper. Common example of Registration system is considered and presented in a way so that readers can understand it easily. This is a simple system, where user enters credentials for login, if a user is registered user then user will login into the system and proceed for authentication. If the criteria at client side and server side are satisfied, then access is allotted. Event sequence Graph (MdAzaharuddin Ali *et al.,* 2014). mentioned that set of events could be mentioned with this graph, in case of registration system mentioned above; four modules, login, validation, authentication, accession could be considered.

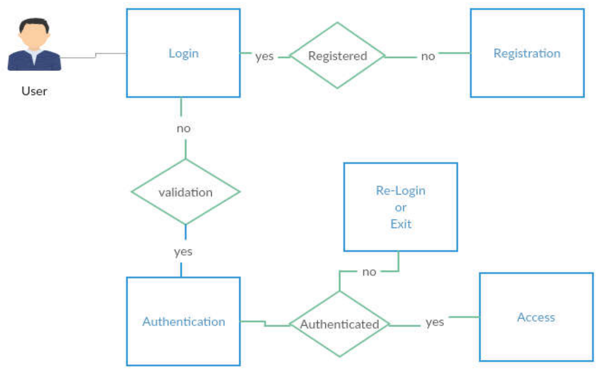

Figure 1. Example of Registration System

Control ow standards

All Nodes: Every node in the registration should be reachable at least once.

All Edges: Every edge of the registration system should be reached at least once.

Corresponding author: **Supriya S. Patil,**
Information Technology, Bharati Vidyapeeth University College of Engineering, Pune (India)

Data ow standards:

All Uses: Every use should be reachable.

All Pot Uses: Every potential use should be reached.

According to the author of this paper, this work could have been extended to a system, where fault detection could have been much easier. Decisions made with the help decision tree, on vertical side parameter values are used and on horizontal side rules referred to test successful functioning of functionality is used (VikasPanthi and Durga Prasad Mohapatra, 2012).

Table 1. Decision Tree for Login module of the Registration System

Login	Rule 1	Rule 2	Rule 3	Rule 4
<UN,PW>	T	T	T	T
UN∈ N	T	T	T	F
PW ∈W	T	F	T	T
Login Successful	T	T	F	T

According to the decision tree plotted the successful test case is considered on the basis of rules followed by the parameters mention. For example, to use login system, Login should be successful (Rule 1,2,3 and four must be satisfied and third need not to be satisfied). Results are verified with help of positive and negative values. In case of login model of registration system, the percentage of visiting all nodes by user is 70 % which is acceptable.

Table 2. Verification of results login module of registration system

	All Node	All Edges	All use	All POT use
Positive	70	80	78	90
Negative	30	20	22	10

Combination of extended finite state machine and UML sequence diagram

- Extended finite machine diagram is accompanied by trigger condition and necessary data
- The process of EFSM could be categorized in three parts as
- E-block- To evaluate trigger for all conditions.
- FSM-block – To compute state next to current state &a signals, which controls A-block.
- A-block - To perform the required data operations and movements of a data.

Along with the state diagram sequence diagram is appended because, when the emphasis is on functionality based technique then dependency needs to be checked. This part is extremely important because as coding is not explored in MBT, dependency check is needed to ensure the appropriate call of the embedded entity.

The table 2 shows path coverage of Extended Finite State Machine, Sequential Test Model, for every step the associated message, number of states and transitions are mentioned.

All path, all state and all transitions is nothing but the product of paths covered. For example, total number of tests performed for state = 2*1*2*3*2*1=24.

All this parameter gives information about total number of tests in respective field.

Table 3. Path coverage by EFSM_SeTM

Service Name	Message	# state	# transition
Authentication	verify()	2	2
Validation	validate()	1	3
Registration	signup()	2	2
All path	P1*P2…*Pn	3	1
All state	S1*S2…*Sn	2	1
All transitions	T1*T2…*Tn	1	2

There is scope for improvement in EFSM_SeTM is that the system may work only for web services designed using java.

Metamodels and Transformations

(Gutiérrez, 2016) mentioned important concepts in MBT, that is Metamodel and Transformations. In the system proposed there are five transformations are mentioned about five different metamodels. Metamodel gives information about the functionality, especially first two metamodels describe functional requirement and the third one specifies the test scenarios to be tested and fourth deals with the values associated. Fifth metamodel combines, all inputs into segregated test cases format. In case of registration example of login module mentioned in figure 1, functional requirement model will demand username and password in valid form, test scenarios are whether user is registered or not test value is the actual value of username and password entered and test case is combination of all login, validation, authentication and accession module. This system demand input in the form of interaction between external user and system, there could be least applicability where users are not actually involved; system like compiler. As it is based on functional requirement completely, quality of system depends upon quality of functional requirement collected.

Conclusion

The systems discussed in this paper especially for Model based testing. All these systems heavily rely on functionality; if functionality input is not recorded correctly then system may get affected with this. There is a necessity of a model based testing technique with detailed testing coverage, consideration of events, inter module communication. There is also need to clearly design template or a system to conduct test cases, these test cases should be strictly examined with the metrics to check quality of a system.

REFERENCES

Benjamin, M., D. Geist, A. Hartman, Y. Wolfsthal, G. Mas and R. Smeets, "A study in coverage-driven test generation", In Proc. of the 36 th Conference on Design Automation Conference, pp. 970-975, 1999.

Born, M., I. Schieferdecker, H.-G. Gross, and P. Santos. “Model-Driven Development and Testing – A Case Study”. In Proc. of the 1st European Workshop on Model Driven Architecture with Emphasis on Industrial Application, pp. 97-104, 2004

Bouquet, F., C. Grandpierre, B. Legeard, and F. Peureux, ”A Test Generation Solution to Automate Software Testing”, In Proc. of the 3rd international workshop on Automation of software test, pp. 45-48, 2008.

Bouquet, F., C. Grandpierre, B. Legeard, F. Peureux, N. Vacelet, and M. Utting, “A subset of precise UML for Model-based Testing”, In Proc. of the 3rd International Workshop Advances in Model Based Testing (AMOST), pp. 95-104, 2007.

Calame, J. R. 2005. “Specification-based Test Generation with TGV”, Technical Report SEN-R0508, Centrum voor Wiskundeen Informatica.

Ching-Seh Wu, Chi-Hsin Huang," The Web Services Composition Testing Based onExtended Finite State Machine and UML Model", 2013 Fifth International Conference on service Science and Innovation

Crichton, C., A. Cavarra, and J. Davies, “Using UML for Automatic Test Generation”, In Proc. of the Automation of Software Testing, 2007.

Farooq, Q., M. Z. Z. Iqbal, Z. I. Malik and A. Nadeem, "An approach for selective state machine based regression testing", In Proc. of 3rd International Workshop Advances in Model Based Testing (AMOST), pp. 44-52, 2007.

Ganov, S. R., C. Killmar, S. Khurshid, and D. E. Perry. “Test Generation for Graphical User Interfaces Based on Symbolic Execution”. In Proc. Proc. of the 3rd International Workshop on Automation of Software Test, pp. 33-40, 2008.

Garavel, H., F. Lang, R. Mateescu, and W. Serwe, "CADP 2006: A Toolbox for the Construction and Analysis of Distributed Processes", In Proc. of the 19th International Conference on Computer Aided Verification, pp. 158-163, 2007.

MdAzaharuddin Ali *et al*. “Test Case Generation using UML State Diagram and OCL Expression”, International Journal of Computer Applications (0975 – 8887) Volume 95– No. 12, June 2014

ShanmugaPriya S. *et al*, “ Test Path Generation Using UML Sequence Diagram”, Volume 3, Issue 4, April 2013 ISSN: 2277 128X International Journal of Advanced Research in Computer Science and Software Engineering

VikasPanthi, Durga Prasad Mohapatra, 2012. “Automatic Test Case Generation using Sequence Diagram”, International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868 Foundation of Computer Science FCS, New York, USA Volume 2– No.4, www.ijais.org.

****