Literature Survey on Model Driven Testing-Functional Test Case Generation with Redundancy Check and Model Paradigm Approach

Suparna Naik
Research Scholar,
Bharati Vidyapeeth University College of Engineering, Pune (INDIA)

P A Jadhav
Professor Information Technology,
Bharati Vidyapeeth University College of Engineering, Pune (INDIA)

Abstract - Model based testing is a software testing procedure in that the test cases are obtained from a model that explains the functional feature of the system which is under test. If the system model is explained clearly, reflecting the system performance accurately, it would be utilized as a shareable and reusable object. As example, the model could be utilized to create a proper test suite for the System under Test. This method is called as model based testing (MBT). Context: Inter-operability, output, system feasibleness and flexibility are the main advantages of the modeling of software as well as method of model driven. Though, some effort had been taken for determining the actual difficulty in the system, importance and benefits of the system. Aim: important objective of this paper is given as: (1) To determine relevance and distribution of technique of modeling of software and model driven technique (2) To learn predicted as well as obtained benefits also (3) To recognize the issues that avoid their diffusion.

Keywords: Model Driven Techniques, Model-based testing, software modeling and MD techniques

I. INTRODUCTION

Now a day, academic and industry has been focused by the model driven testing approach to development of software. Model driven testing approach is unlike to traditional technique of software development. Also this technique determines the usage of model at each stage of the software development process. Final outcome of this change have been taken through its essential modification in design of software, tested and maintained. Testing is the essential activity in development process as it consumes time for improving and significant approach. A testing cycle encompasses three main parts: In this paper, most challenging and difficult activity is generating test cases, executing test cases, and evaluating test cases. Utilizing UML in model based testing or development encourages researches for utilizing unified modeling language diagrams like activity diagram, sequence diagram, state diagram, and many others for crating test cases. Productivity as well as quality of the model can be improved by changing activity of testing to prior step of the development process. Also, the generating test case which wouldn't dependent on any special development stage [2]. Several methods of model base testing are available that are different in their particular objectives, supporting tools, applying strategy for evaluation and in their design. To make survey of strategies and techniques used in model based testing and compare these techniques with each other.

Each MBT approach is classified in one of the five categories: [A] Information have been presented by model by utilizing requirements of software and explained by utilizing unified modeling language diagrams. [B] Information obtained from the requirement is represented by model and explained through non UML notation. [C] Information represented by model by utilizing structure, units, interfaces and architecture also explained by diagrams of uml. [D] Information represented by model by using internal architecture of the system and explained by utilizing non uml symbols. [E] While searching, the collected papers are excluded because these are not related to model based testing.

II. METHODOLOGY

MD testing had number of features which will be measured.

In our study, we focus on the following criteria:

• Modeling Language: Unified modeling language is studied during this paper as modeling language.

• Automatic Test Generation: Technique for generating test cases has been studied which will be further utilized by approaches whether automated or not. Also, for generating test case several conditions are required. These conditions required to satisfy to be useful.

• Testing Target: Though survey done during the study of this paper targets on test case creation by using models. Testing features could be change from one method to another method. Few researchers can focus on designing of model while another would focuses on execution.

III. RELATED WORK

Escalona etal-2011 and Denger and Mora-2003[1][2], these are two important surveys done during the creating this paper. They studied the current gap in methods which deals with functional test cases creation by using functional requirements. Most applicable conclusion of these two surveys is summarized in this section.

The above given survey are limited only for functional test case, also some other applicable study for same case has been published such as Anand et al-(2013)[3]. In this paper, the author compared automatic generation of test cases technique. The Dengerand Medina's did not follow any standard template for functional test cases during his study. Overall, every technique utilizes its own format and template. Also, techniques does utilizes mechanism of path analysis, as per given in previous step, the method make an extra formal illustration of functional necessities. The author of this paper studied the Escalonas survey and concluded it as last one. They mostly have the same opinion about several existing techniques that needs to formalize requirements like initial stage to make functional test cases, as utilization of text template and informal language for explaining functional requirements**..** In this technique, this is important stage which presents more demanding supporting tool and systematization. Though, it would be studied that few methods gives efficient way, or also gives some automatic way to create model which is formal for automate process. Here, it's possible due to requirement is explained in natural language. Therefore, they are Meta model sand some transformations from this description enable translating necessities in natural language into activity diagrams.

As per the Escalona's survey, revealed at the end of2011[1], cites twenty four approaches; the previous dates back to 1988 (Category-Partition Method) and also the newest to 2009. Denger's, revealed in 2003[4], cites twelve approaches; the previous from 1988 (it is that the same approach utilized in Escalona's survey) and also the newest

from 2002. Below, there are several samples of the approaches integrated in Denger's and Escalona's surveys and new approaches not enclosed in any surveys thus on update the present scenario.

As Hartmannetal (2002)[5] begin their approach with useful requirements written in natural language. The result is a collection of useful test cases obtained from a coverage criteria supported combinations that based on Boolean proposition.

As Binder's book (Binder, 2000)[6] explains the application of the class Partition technique to use cases. Classes are any point within which the behavior of the use case is also dissimilar in 2 realization of the use case. This application is understood as Extended Use Case Pattern.

In addition, Ibrahimetal (2007)[7] present a tool, referred to as GenTCase, which create test cases automatically by utilizing use case diagram enhanced with each use case tabular text clarification.

As Fröhlich and Link (2000)[8] recommend as approach telling how to translate a useful requirement from natural language into a state chart diagram during a systematic way, furthermore as a way to manufacture a group of functional check cases from that diagram.

Ahlowalia (2002)[9] recommended an approach managing translating functional necessities from natural language into a flow sheet and playing a path coverage technique to get check cases.

Mogyorodi's (2003)[10] approach explains functional necessities as cause impact graphs that manufacture test cases from diagrams.

Bodduetal (2004)[11] suggests an approach divided into 2 blocks: the primary one presents a natural language analyser making a state machine from functional necessities, and these condones however explain to produce test cases from such state machine.

Swainetal (2010)[12]initiate a parallel approach to the one in Briandand Labiche (2002)[13]. Initially, a UML activity diagram is made with execution dependencies among use cases. This diagram represents that use cases should be first executed. UML sequence diagrams define use cases and a coverage criterion is applied to extract implantation situations from these diagrams. Check aces are the Cartesian mixture of the trail from the activity diagram and also the situations from the sequence diagrams. Sharma and Singh (2013)[14] recommended an current work supported an rule for deriving test cases from use case model, class diagram and information dictionary. Info regarding this rule isn't provided during this paper.

A recent paper (Nogueiraetal 2014)[15] provides a technique for the automated creation of test cases of parameterized use cases templates. This approach considers a natural language illustration that mixes management and state illustration, which is getting used to pick out specific scenarios throughout test generation. Unfortunately, it only covers successive features, though it is planned to deal with different options as future work. The progressive introduced during this section shows that no definitive approach closes the problem of making functional text cases automatically in a very satisfactory manner.

3.1 FSM

This literature review paper addresses the utilization of sensible usage model supported finite state machine. As per the FSM usage model, it details the method of generating test cases. Finite state machine are the most

effective technique for implementing computational model for hardware and software. By using the idea of finite State modeling, computational models of internet application are often designed.

Andrews et al. [16] planned in their paper, a system level testing methods that represent test generation supported by finite state machines with constraints. A hierarchical way to model probably large internet applications is getting utilized by researchers. This way of building a Finite State Machines that models subsystems of the online applications then produces test requirements as sub-sequences of states within the FSMs. These sub-sequences were then joint and refined to form complete feasible tests. The constraint was utilized for choosing a reduced set of inputs with the objective of reducing the state space explosion otherwise inherent in using FSMS. This paper explains the mechanism that contains a running example of a web based course student system and initiates image execution to support the technique.

Song et al. [17] explained modeling information Interactions in internet applications with making test Cases. In their paper, on information relations in modeling and testing internet applications particular care was taken. An increased FSMs (GFSMs) were utilized as a tool to form information interaction. This paper planned an internet testing model with database interaction for making test case. It starts by making the GFSM test tree resulting from FSM of the online application. An algorithmic program is then considered for creating a smallest test set of GFSM test tree. Then test ways were manufactured to cover every component of test set. Eventually, an algorithm was explained to understand the test ways by reducing the overlap. The way they have planned will gives extensive results with test paths and state transition were all fewer.

Song et al. [18] proposed Model Composition and creating Tests for Web Applications. Security of Web is guaranteed by the web testing. Generally standard web application has two tiers architecture: the client and server architecture. The current woks related to the web based application testing from users point of view doesn't not considers interaction as well as behaviour of server. For this purpose, this paper explains method for interaction as well as behavior of server also explains the way for composition model and internet application testing was proposed. FSMs are getting used to model net applications from the user's side, and therefore the server's side, severally. Afterward, synchronous product was utilized as a tool for constructing a composition of FSMs. Eventually, supported the composition of FSM, test generation is given out that satisfied the corresponding coverage criteria. net applications were typically utilized in our everyday life. Due to the client interaction or communication with the server by sending message requests and response, the synchronous product is utilized to end the composition of the consumer model and therefore the server side model. Eventually, supported the composition model, the tests were created.

Hierons et al. [19] have explained the theory of mutation testing by utilizing PFSM. Group of mutants are generated by mutating programs into order, this process is known as Mutation testing. Then the groups of mutants are used either to process test creation or to evaluate efficiency of test suit. Though the method explained in this paper has been utilized for specifying test which are derived from FSM. This paper is extended mutation testing to finite state machine models in which transitions had associated probabilities. Hierons had represented some ways for mutating PFSM with the sequence of test which would be produced in output which differ probabilistic finite state machine and their mutants? After that every sequence of test is applied many times for testing and examines the sequence generated in result output. Statistical sampling theory is then utilized for calculating the every output sequence's frequency.

Cavalli et al, [21] explained WebMov: It is the framework developed to model and test the composition of web services. Cavalli and his colleague had explained all theory about their research and final result of the WebMov,

Their primary goal was to learn different strategies and produce new tool which could cover each and every step of development life cycle of internet services. They mentioned that the framework of this tool will allow the interaction which helps to obtain particular verification and modeling activity as complement. The strategy of the WebMov was wrapped in this framework. This strategy contains group of tools that is used to implement the model and passive algorithms which are used to generate test**.** As authors have explained several technique has been used to model web services. These techniques and methods are dependent on various options of TEFSM (Time Extended FSM). For performing the robustness and conformance testing this strategy is mixed up with fault injection.
Kalaji et al, [20] declared the problem of EFSM could be expressed to find appropriate way during the Extended FSM and then test data is derived to follow different paths. Preferred paths might be impossible and therefore it's very important to get methods which will the direct search path from Extended FSM to those which are possible. But still, it is big problem in open research that to create or generate FTP (Feasible Transition Paths) for MBT. In this paper authors have explained the fitness metric. This metric will analyze flow of data dependent on action and condition of path transaction. This method is used for estimating the feasibility. Fitness metric is estimated which is then utilized in genetic algorithm for guiding searches to feasible transaction paths. The strategy explained in this paper has two stages: first is producing FTP and second is search for input sequence to activate this TP. The problem of crating input path sequence is for finding input sequence which passes through feasible path. Genetic algorithm is used in second stage. Fitness function of this algorithm is depending on combination of levels and functions of the algorithm [22].

Wang et al. [21] Here, Wang and his colleague explained the strategy for transforming UML model to Finite State Machine Model for automation testing. The particular technique was proposed which allows creating Finite State Machine with same semantics as author targeted in translating state diagram. One state diagram is generated by Models for all objects and relations among them are represented in another diagram. Unified Modeling Language is very simple, easy for use and understands between various modeling languages. However accurate examination for UML model is difficult because of its lack of accurate semantics. As formal, Finite State Machine gives way to create test cases automatic. Thus author presented method to transform Unified modeling language to finite state machine model. Author explained method to execute technique as well as prototype of tool to maintain method.

IV. CONCLUSION

In this paper we are able to outline some lacunas found in papers consider for survey. In this it is important to be focus on functional necessities of system to generate the test cases using different UML diagram. One more aspect that is yet to be focused is detecting redundant test cases automatically. Now a days it is very general that more number of unneeded test cases are created those are created from referral model and continuously used while actual execution of test cases. This is because of only repeated test case path and repeated functionalities consider. These test cases have to identify manually by test team. The second dimension is model paradigm. This approach refers to the notation used to describe the model of system. In this more preferred practice is to t refers Use cases to generate is finite state machines (FSM) with respect formal notation. From this we will get transaction based notation in the proper value for this dimension.

REFERENCES

[1] Escalona et al, "Generation of test cases from functional requirements. A survey", Aug 1, 2011 - Volume 84 Issue 8, August, *2011*
[2] Denger and Mora, "Test Case Derived From Requirement Specification", Published by IESE, April 2003
[3] Anand et al, "An Orchestrated Survey on Automated Software Test Case Generation", the Proceedings of IEEE/ACM Workshops on Automation of Software Test (AST'06 – AST'12).
[4] Denger, C. Medina M. 2003. Test Case Derived from Requirement Specifications. Fraunhofer IESE Report.
[5] Hartman, Cavarra, "Using UML for Automatic Test Generation", *ISSTA* 2002 Rome, Italy Copyright 2001 ACM
[6] Robert V. Binder, book "2011 Model-based Testing User Survey: Results and Analysis"
[7] Ibrahim et.al, "Automatic generation of test cases from use-case diagram", discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/228576271, 2007

[8] Frohlich and Link, "Automated Test Case Generation from Dynamic Models", 2000
[9] Ahlowalia, "testing from use cases using path analysis technique", International Conference on Software Testing Analysis & Review November 4-8, 2002 Anaheim, CA USA
[10] Mogyorodi, "What Is Requirements-Based Testing?" 2003, **12** CROSSTALK The Journal of Defense Software Engineering
[11] Boddu et.al, "RETNA: From Requirements to Testing in a Natural Way", *Proc IEEE Int Requir Eng Conf*. 2004 September ; 2004: 262–271. doi:10.1109/ICRE.2004.1335683.
[12] Swain et.al, "Test Case Generation Based on Use case and Sequence Diagram", Int. J. of Software Engineering, IJSE Vol.3 No.2 July 2010
[13] L.C Briand and Y .Labiche. "A UML-Based Approach to System Testing". In Proceeding of the 4th International Concepts, And Tools, Springer Verlag, London, pp.194-208, October 01-05,2002.
[14] Sharma, Singh, "Generation Of Automated Test Cases Using UML Modeling" International Journal of Engineering Research & Technology (IJERT) Vol. 2 Issue 4, April - 2013
[15] Nogueira et.al, "Test generation from state based use case models", DOI 10.1007/s00165-012-0258-z BCS © 2012 Formal Aspects of Computing (2014) 26: 441–490, 2014
[16] Bo Song, Shengwen Gong, Shengbo Chen, "Model Composition and Generating Tests for Web Applications", In Seventh IEEE International Conference on Computational Intelligence and Security, 2011.
[17] Hierons R.M. & Merayo M.G., "Mutation Testing from Probabilistic Finite State Machine", In IEEE Conference on Computer Assurance, 2007.
[18] Ana Cavalli, Tien-Dung Cao, Wissam Mallouli, Eliane Martins4,Andrey Sadovykh, Sebastien Salva, Fatiha Za di, "WebMov: A dedicated framework for the modelling and testing of Web Services Composition", In International Conference on Web Services, 2010.
[19] A. S. Kalaji, R. M. Hierons & S. Swift, "Generating Feasible Transition Paths for Testing from an Extended Finite State Machine", In IEEE International Conference on Software and Information Technology, 2009.
[20] A. S. Kalaji, R. M. Hierons & S. Swift, "Mutation Testing of Probabilistic Finite State Modelling", In international Conference on Information and Software Technology, 2011.
[21] Xi Wang, Liang Guo, Huaikou Miao, "An Approach to Transforming UML Model to FSM Model for Automatic Testing", In International Conference on Computer Science and Software Engineering, 2008.
[22] Liping Li, QIAN Zhongsheng, Tao He, "Test Purpose-Based Test Generation for Web Applications", In IEEE International Conference on Web Services, 2009.