Test Case Generation and Minimization using UML Activity Diagram in Model Driven Environment

Ms. Hetal J. Thanki [#1], Prof. S.M.Shinde[*2]

[#](M.E. COMP), Department of computer engineering, JSCOE, Hadapsar, Pune. University of pune. India

***Abstract*—** Test driven design (TDD) and design driven testing (DDT) are used for test case generation. TDD generates so many duplicated test cases at the end of the project. DDT is novel approach to generate test cases based on design model of application. Comparative study indicates batter result using DDT. White-box traditional regression testing depends on analysis of impact of changes in source code. This practice minimizes the amount of testing required to validate code changes but they do not influence on requirement specification. Black-box testing supports ability to test from higher level design and requirement. Test optimal is a tool to generate test cases based on model driven environment using UML activity diagrams. Traditional approach generates many numbers of large and duplicate test cases. Proposed approach will minimize generated test cases and generate optimal test suite using model driven testing. Changes in design of project, modified activity diagram element will identify common and uncommon test cases. Where uncommon test cases are focused for further testing. Filtering will lead to better available resource utilization and will improve software project management.

***Keywords*—**Activity diagram, Model driven testing, Regression testing, Test minimization, common test cases, uncommon test cases.

I. INTRODUCTION

Model based testing approaches support a top-down or black box testing where design and requirement models are used for test case generation. In this approach test cases are developed by focusing on the use of models rather than source code as its primary artifact. Model-based testing approaches have been developed to simplify the process of test development and execution. Test occurs in every stage of software development process. If some changes in model design occurs than whole or part of test suite for some project or functionality can be change. Much large software has lots of test cases generated at the end of design and development by developer and tester. Different modules are developed by number of people; this will lead to duplication of generated test cases. To develop and execute large amount of test cases is very time consuming task. It also increases cost of organization and effort of people of organization. It is necessary to reduce number of test cases in such a way that it checks whole features of the software, cover whole part of software and also reduce time and effort of developer.

In our work test cases for different projects are generated through a tool called test optimal tool. That generated test cases are stored in either in excel file in table format or in HTML format. Data of generated files are extracted through code and store it into database. Stored test cases are larger in number which are minimized with the help of changes done in original activity diagram of a project. Original and modified activity diagrams are compared through software to identify common and uncommon test suites. We will test our project through different activity diagram of number of projects of same domain and finally analytics of change and minimization is presented through graph. With the advent of technology software becomes very crucial part in all the institute and industries. To develop particular software and to test it as per customer requirement is very important because the software which is not able to satisfy customer requirement after development will leads to increase and waste of cost, time and effort of all parts of organization. Model-based test case generation is gaining acceptance to the Design and implementation of new technique for optimum number of test cases generation and minimization using model driven testing.

Advantages of this are the early detection of faults, reducing software development time etc. Reduce test case suite RT may provide same structural coverage as in original test suit OT with comparatively fewer test cases. In recent times, researchers have considered different UML diagrams for generating test cases. Little work on the test case generation using activity diagrams is reported in literatures. To increase the productivity batter test case suit generation is very important for any small or large software application. To generate batter test cases of any project which reduce number of test cases to be executed and also all part coverage is necessary now a day. Traditional software testing techniques consider only static view of code which is not sufficient for testing dynamic behavior of object-oriented system, use of code to test an object-oriented system is complex and tedious task. In contrast, models help software testers to understand systems better way and test information only after simple processing of models compared to code, model-based test case generation can be planned at an

early stage of the software development life cycle, allowing to carry out coding and testing in parallel. Due to that, model-based test case generation methodology becomes an obvious choice in software industries. Main advantage of this model is its simplicity and ease of understanding the logic of the system. Highly critical, large and very important software application generation is task of responsibility. If any failure occurs due to wrong basic design of the whole software or part of the software will lead to failure of project and very big loss to organization. To overcome this problem testing is very necessary part of software development life cycle. If this testing is perform with the scratch level of system development than faults, errors, bugs are detected at early stage of software development. Model driven testing and best test suite generation through different techniques will help developer to increase productivity, profit and decrease cost, time and efforts.

Filtering of test cases will minimize number of test cases to be tested further. Based on highly critical part of software risk factor can be specified for particular test procedure or test case of activity diagram. Change in that part of software affect whole operation of software. This type of critical part must be test first to identify effect of changes in critical part as early as possible. If this test cases are given higher priority than task can be easy to overcome failure and to identify faults early. Recently Infosys develop a software which identifies customer usage criteria and based on that test case prioritization is done.

II. RELATED WORK

Our proposed work is to generate optimum test case suit. By observing previous work of different authors we divide this work in three parts like how Test cases are generated for different domains, how platform independent model transform to executable test cases and how to minimize test cases using different criteria which are given in our algorithm in latter part.

Different model driven testing techniques and its comparison with different criteria like modeling language used, tool support, the testing targets etc. given in [1]. Roberto S. et al. [2] Authors have extend TDE/UML is a tool to generate test cases based on model driven environment using UML diagrams. Traditional regression testing procedure is bottom- up that depends on changes in source code. This approach and model driven testing supports to generate and prioritize test cases based on top-down testing approach. This approach generates test cases based on user defined concerns and depend on trace-ability link between models, test cases and code and also user defined properties associated with model elements. It extends model based testing environment.

Testing of applications of different domains is important and crucial task. Use of Web application increases day by day. To test web application demand of systematic methodology is increasing. Framework for supporting such methodology which is loosely coupled using different models is describe the system under testing web application model is generated. Based on web application model test case models are generated. To describe the environment and process of test execution test deployment and test control model are generated. The test engine execute test cases automatically and results are reflected to test case models [3]. In network domain network management interface model driven testing technique is developed [4]. In this method from platform independent model (PIM) and platform specific model (PSM) defined in network management interface specification, test case model and test scenario model constructed either automatically or manually. Interface testing of 3G mobile communication networks is done through implemented automatic testing platform tool. Model driven conformance method [5] is used for 3G network management north bound interface to cope with technological and specification changes. In this method interface technology independent test model (PIT) is transformed to interface specific test models (PSTs) which can be used in conformance testing by proper test tools. PIT is derived from platform independent model. For Collaborative Embedded System Design (for embedded domain) testing is key issue. It is difficult to design test cases for Security protocols for security critical application (security domain) SecureMDD is combined with design of functional and security test. By this method it is easy to define test cases during modeling stage [14]. It also generate runnable test for application. Case study is described for open source data structure using zellers algorithm for failing test case minimization which reduce length of sequences of method calls [6]. Based on analysis of relationship among testing requirement and test cases, software faults and changes, priority of testing requirements one method called regression test case design is developed which achieves the function of checking the regression test case suite [7]. To generate batter test suit including longer test cases which achieves higher fault detecting and higher coverage of code experimental studies in a scenario of specification based testing for reactive systems but application of test case minimization will create opposite effect [8]. Automatic test cases in software product line [9] uses standard UML 2.0. Automatic test case generation based on model driven architecture is done through system model to test model conversion and test model to test code generation which is done through model to model conversion method and model to code conversion method respectively.

III. UML ACTIVITY DIAGRAM

"The Unified Modeling Language (UML) is a language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems" [12].

Activity diagram depicts flow of behavior of a system that extracts the central idea from flowchart. It applies to number of domain. The activity diagram contains activity states that are made up of smaller actions which are represented in the implementation of a statement in a process or the performance of an activity in a workflow [10,11]. To describe flow in activity Nodes of activity diagram are used like **Initial node, Decision Merge node, Fork node, Join node , Final nodes**

Fig. 1 shows general example of activity diagram including whole control nodes. Where A1 to A7 are activities or actions of activity diagram. Node A2 has input and output pin. Activities in nodes A1, A2, A4, A5 are concurrent activities. Activity path A4, A5 is terminated with flow final node. Activity A2 and A3 are join through join node as shown in figure. All the data or token of A2 and A3 are passed to A6. Detail activity diagram with data store, loop, streaming actions, expansion region, and buffer node can be drown[12]. This activity diagram describes expected behaviour of an operation.

IV. IMPLEMENTATION DETAIL

Test optimal tool is used to generate test cases from activity diagram or state diagram of the project. That output generated test cases are stored in EXCEL file in table format or it will be in HTML format. Proposed project extract test case data from activity diagram of project through this files and store it in database. Than this test cases are further analyze for minimization and optimization with the help of criteria mention in algorithm. Figure 3 Depicts architectural design of the system. We will consider different projects of same domain with activity diagram and will do analysis of their model driven testing using test-optimal and my dissertation. Manual testing will be done in my dissertation.

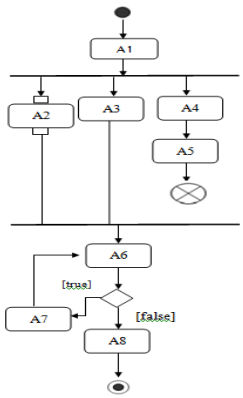

Fig. 1: General example of activity diagram

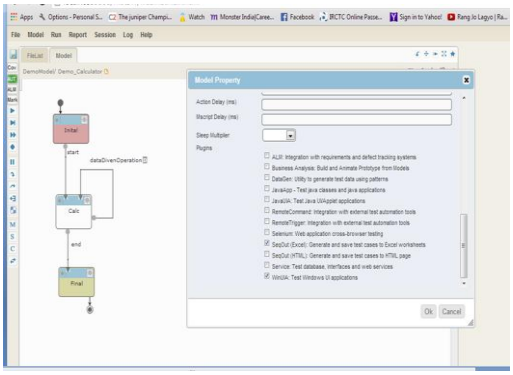

Fig. 2: Test optimal tool

A. *Block Diagram*

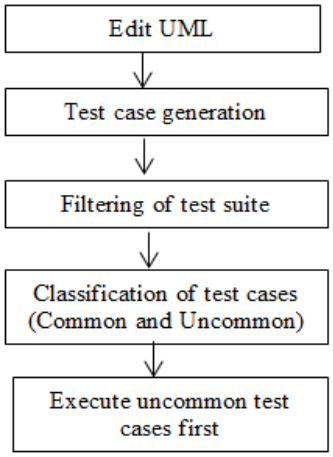

Fig. 3: Proposed design

B. *Algorithm*

1. During editing of model, it is monitored for changes and modifications in UML diagram are recorded.
2. Based on OTS, MTS collected during edit time structural and semantic changes are identified. Then test cases are generated and labeled as common (CO) and uncommon (UC).
   CO = CO U {A1}
   UC = UC U {A2}
   A1 ← changed test cases
   A2 ← no change in test cases during model modification
3. Uncommon test cases are filtered based on comparison of OTS and MTS.
4. Execute UC first to test modification part of activity diagram.

V. RESULT

With the use of test-optimal software with our project, It will generate Minimize number of test cases, Test case analytics. Different graphs can be generated through proposed system like Change in test cases number with respect to changing activity diagram element, various numbers of test cases for common and uncommon test cases. Fig 2. Shows test optimal tool. Draw original activity diagram and execute it. Proposed system will back up original model. Now modify original model (Activity diagram) and execute it. Proposed system will compare two diagram and divide original test suite in two parts like common and uncommon test cases. Fig 4. Shows activity diagram Hall booking with modified portion. Test suites, common and uncommon test cases are represented in Fig 5. Where test cases of modified part is shown in uncommon part because that are the test cases which we want to execute first to test modified part of design. Fig 6. Shows graphical results for number of common and uncommon test cases.

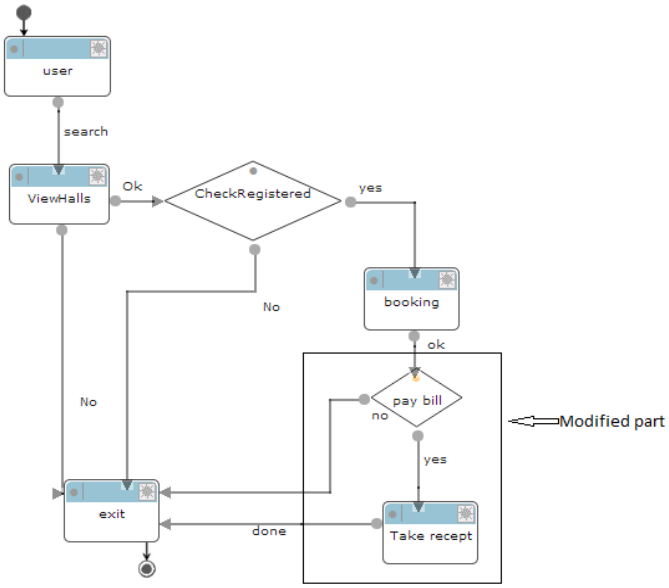

Fig 4. Shows activity diagram for Hall booking

Common test cases testcases	Uncommon
CheckRegistered->No CheckRegistered->yes exit->exit user->search ViewHalls->No ViewHalls->Ok	pay bill->yes pay bill->no booking->ok Take recept->done

Fig. 5 Results for Hall booking system

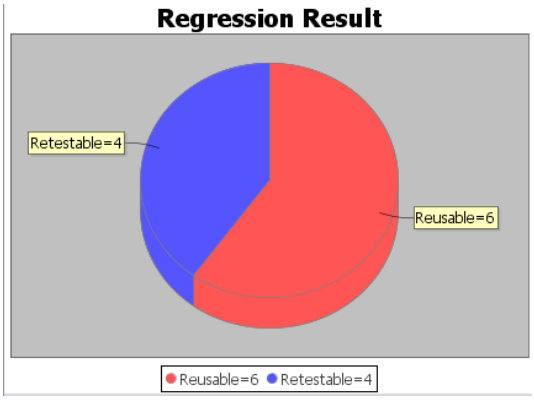

Fig. 6 Number of common and uncommon test cases.

CONCLUSION AND FUTURE SCOPE

Software testing has been critical part of entire SDLC where success of software projects is heavily depends. Therefore there have been wide spread efforts to improve SDLC practices especially during testing. In our paper we present a model driven testing approach with improved prioritization of test cases. We have prioritized test cases based on various criteria like code coverage, test and time efforts required for executing test cases. We believe that such prioritization of test cases will result in better utilization of available resources and improved software project management. Time complexity of proposed project is depends on number of input value (test cases).

In future test cases can be prioritize by considering various domains for which software being developed and profile of users.

REFERENCES

[1] Mohamed Mussa, Samir Ouchani, Waseem Al Sammane, Abdelwahab Hamou-Lhadj, "A Survey of Model-Driven Testing Techniques", *2009 Ninth International Conference on Quality Software, IEEE*, pp 167-172, 2009

[2] Roberto S. Silva Filho, Christof J. Budnik, William M. Hasling, Monica McKenna, Rajesh Subra-manyan, "Supporting Concern-Based Regression Testing and Prioritization in a Model-Driven Environment, 34th *Annual IEEE Computer Software and Applications Conference Workshops*, PP 323-328, 2010

[3] Nuo Li, Qin-qin Ma, Ji Wu, Mao-zhong Jin, Chao Liu, "A Framework of Model-Driven Web Application Testing", *Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC'06), IEEE*, 2006

[4] Yinghui Chen, Limin Hou, Yichang Liu, Yongping Zhang, Feng Qi, "Study and implementation of model driven testing method for network management interface", *Proceedings of ICCTA2009, pp. 259-263, IEEE, 2009*

[5] Liu Yi-chang, Chen Ying-hui, Qi Feng, Qiu Xue-song, "A Model-Driven Conformance Testing Method for 3G Network Management North Bound Interface", *IEEE*, pp. 323-326, 2010

[6] Yong Lei ; Andrews, J.H., "Minimization of randomized unit test cases", *16th IEEE International Symposium*, pp. 267-276, *IEEE*

[7] Wenhong Liu ; Xin Wu ; Yuheng Hao, "Research and Application of Regression Test Case Design Methods Based on the Analysis of the Relationship", Fifth *International Conference, IEEE*, pp. 233-236

[8] Fraser, G. ; Gargantini, A., "Experiments on the test case length in specification based test case generation", *IEEE*, pp. 18-26, 2009

[9] BeaBeatriz Pérez Lamancha, **"**Model-Driven Testing in Software Product Lines"**,** *Proc. ICSM 2009*, Edmonton, Canada, pp. 511-514, *IEEE*

[10] Yang Liu , Yafen Li, Pu Wang , "Design and Implementation of Automatic Generation of Test Cases Based on Model Driven Architecture", *2010 Second International Conference on Information Technology and Computer Science*, 2010, pp. 344-347, *IEEE*

[11] Puneet E. Patel, Nitin N. Patil (2013). "Testcases Formation using UML Activity Diagram". *2013 International Conference on Communication Systems and Network Technologies*, *IEEE*, pp. 884-889

[12] D. Pilone and N. Pitman. UML 2.0 in a Nutshell. OReilly, June 2005.

[13] Kuzman Katkalov, Nina Moebius, Kurt Stenzel, Marian Borek and Wolfgang Reif, Model-Driven Testing of Security Protocols with SecureMDD, IEEE, 2012

[14] Matt Stephens and Doug Rosenberg, Design Driven Testing, Test Smarter, Not Harder, Apress, 2010