Literature Review of Test Case Generation Techniques for Object Oriented System

Namita Khurana
Research Scholar, (Deptt. of Comp. Sc. and Applications, M.D.U, Rohtak)

R.S Chillar
Head (Deptt. of Comp. Sc. and Applications, M.D.U, Rohtak)

ABSTRACT
It has already known that software testing is one of the most important and critical phase of software development life cycle assuring the verification and validation process of the software. Testing of software requires a great deal of planning and resources as it is a time-consuming activity. The software testing immensely depends on three main phases: test case generation, test execution, and test evaluation. Test case generation is the core of any testing process and automating it saves much time and effort as well as reduces the number of errors and faults. In this paper a survey on various object oriented testing techniques for generating effective test cases is presented. For example test case generation using genetic algorithm, using UML sequence diagram, using UML activity diagrams, Scenario based test case generation etc.

Keywords
Object oriented software system, Software testing, UML diagrams.

1. INTRODUCTION
Almost everything in day today life has an element of software in it. An organization that develops any product must put an effort to reduce or totally remove the faults before delivering that product. The consequences and impact of each single defect needs testing. It may be acceptable to say that 99.9% of the defects are fixed in the software product for release and 0.1% is outstanding [1].

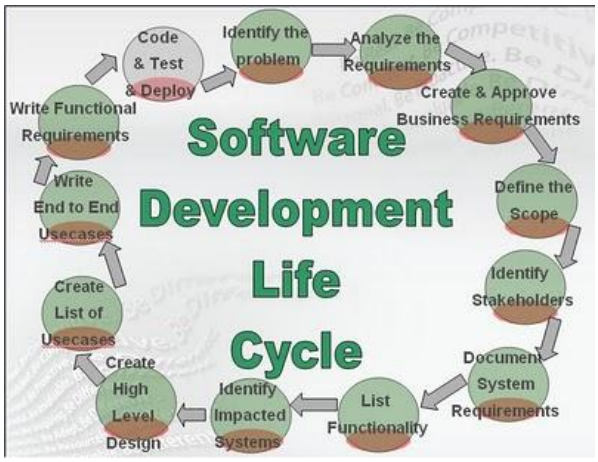

Fig 1: Software Development Life Cycle Steps

Software testing, the inevitable phase in the Software Development Life Cycle (SDLC) plays a vital role in deciding the delivery of the product as well as to ensure the quality of the product. Testing can be performed on requirement, design and code. However, if testing is followed in the initial phase in SDLC most of the errors can be eliminated and can be prevented without disseminating to the next phase.

Software testing can be stated as the process of validating and verifying that a computer program/application/product meets the requirements that guided its design and development; works as expected, can be implemented with the same characteristics, and satisfies the needs of stakeholders.

For a any given part of software we will be writing a set of test cases that called test suites and it is used to group together similar test cases. A properly generated test suite may not only locate the errors in a software system, but also help in reducing the high cost associated with software testing. A test case, in software engineering, is a set of conditions or variables under which a tester will determine whether an application, software system or one of its features is working as it was originally established for it to do. Test suites is a collection of test cases that are planned to be used to test a software program to illustrate that it has some specific set of behaviors.

Fig 2: Software Testing Life Cycle Steps

2. LIFE-CYCLE OF OBJECT-ORIENTED SOFTWARE TESTING
The Full Lifecycle of Object Oriented Testing (FLOOT) is shown in the fig-1[26].

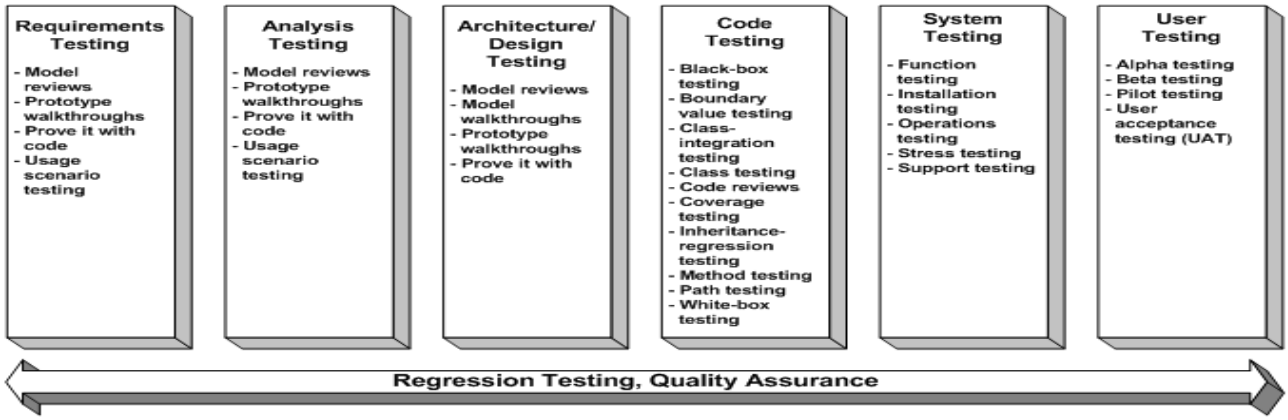

Fig. 1. FLOOT Lifecycle

This paper is structured in five sections; Section 1 covers the basic introduction whereas section 2 presents the test case generation approaches. Section 3 gives the testing techniques for object oriented software. Section 4 gives the problems faced during object oriented testing. Section 5 discusses about literature Review. Conclusion is drawn in the section 6. References are given in Section 7.

3. TEST CASE GENERATION APPROACHES

This is the main part of the testing process. The Approaches involved in generating test cases can be categorized in these three parts : Scenario Based Test Case Generation, Model based test case Generation and Genetic Based test Case Generation .Even though variety of approaches have been proposed yet for a decade there has been constant research on generating test cases based on specification and design models.

3.1 Scenario based Test Case Generation

In Scenario based test case generation test scenarios are used for generating test cases .Baikunt Narayan Biswal has presented a paper. A Novel Approach for Scenario based test case Generation [28]. This paper deals with Test adequacy Criteria for complex transactions or Events, Scenario based testing gives best results. Test case generation UML Activity diagrams presented by Kim are also based on concurrency in Activity Diagram where multiple systems interact with each other.

3.2 Model based Test Case Generation

In model based testing, the testing begins at design phase. So, early detection of faults can be achieved by using this approach further reducing time, cost and efforts of the developer to a large extent. Automatic Test case generation using Unified Modeling Language (UML) state diagram by P.samuel and A.K.Bothra and Rajib Mall published on the basis of Model Based Test Case generation.Test Case Generation by UML Sequence Diagram and labeled Transition System. The procedure is based on the model based testing techniques with test cases generated from UML Sequence diagram converted into Labeled Transition System(LTS).Test Case Generation Based on Use Case and Sequence Diagram by Santosh Kumar Swain, Durga Prasad Mohapatra and Rajib Mall . Test cases are derived from a model based on system Graph integrating UDG and SDG.

3.3 Genetic based Test Case Generation

In Genetic based test case generation technique, the test cases are generated using Genetic Algorithm. Improving GA based Automated Test Data Generation Technique For Object Oriented Software [7] by Nirmal Kumar Gupta, Mukesh Kumar Rohil,. The proposed strategy shows that genetic algorithms are useful in reducing the number of unfeasible test cases by generating test cases for object oriented software.

A Hybrid Genetic Algorithm Based Test Case Generation Using Sequence Diagrams[10] by Mahesh Shirole, Rajeev Kumar. Test cases generated using genetic algorithm improves the method coverage as well as exception coverage.

Object Oriented Test Case Generation Technique using Genetic Algorithms[16] by V.Mary Sumanlatha, G.S.V.P.Raju,. Test cases are generated using sequence Diagram and optimized using Genetic Algorithm.

4. TESTING TECHNIQUES FOR OBJECT ORIENTED SOFTWARE

At various levels of testing of object oriented software, The Techniques which can be applied are

1. Unit Testing
2. Method Testing
3. Class Testing
4. System Testing
5. Integration Testing

5. PROBLEMS IN OBJECT ORIENTED TESTING

5.1 Encapsulation

A wrapping up of data and functions into a single unit is known as encapsulation. This mechanism restricts the access to some of object's components and also restricts observability of intermediate test results. In this case Fault discovery is very difficult.

5.2 Polymorphism

Polymorphism is one of the crucial features of OOP. It simply means one name and multiple forms. In polymorphism, Program entities have been permitted to refer to objects of more than one class, when a hierarchical relationship among these classes exists. Because of polymorphism, all possible bindings, all potential execution paths and potential errors have to be tested.

5.3 Inheritance

Inheritance is the key concept of object orientation. Object oriented language has allowed for defining an abstract data type deriving it from an existing one. It is the mechanism of deriving a new class from an old one. The old class is referred to as the base class and the new one is called the derived class or the subclass. Due to Inheritance there are some invisible dependencies between super/sub-classes. Inheritance results in increased code dependencies due to reduced code redundancy. If there is error in the base class, it will be inherited in the derived class also. A subclass is only tested after testing its super classes.

6. LITERATURE SURVEY

Software developers can't test everything, but they can use combinatorial test design to identify the minimum number of tests needed to get the coverage they want. Combinatorial test design enables users to get greater test coverage with fewer tests. Whether they are looking for speed or test depth, they can use combinatorial test design methods to build structured variation into their test cases. In survey it has been found that these test cases can be generated with different techniques like test case generation using UML models like Activity Diagrams, Sequence Diagrams etc.

The process of generating test cases from design will help to discover problems early in the development process and thus it save time and resources during development of the system. However, it is very difficult to select test cases from UML models. In UML, the behavior of a use case can be represented by using interaction, activity and state machine diagrams. Sequence diagrams capture the exchange of messages between objects during execution of a use case. It focuses on the order in which the messages are sent. Activity diagrams, on the other hand, focus upon control flow as well as the activity-based relationships among objects. These are very useful for visualizing the way several objects collaborate to get a job done.

Ranjita Kumari Swain, Vikas Panthi and Praful Kumar Behera, **"Generation of test cases using Activity Diagram"** have generated the test cases using activity diagrams [2]. In that approach, first an activity flow graph is derived from activity diagram. Then, all the required information is extracted from the activity flow graph (AFG). The activity flow graph (AFG) for the activity diagram is created by traversing the activity diagram from beginning to end, showing choices, conditions, concurrent executions, loop statements. From the graph different control flow sequence are identified by traversing the AFG by depth first traversal technique. Next, an algorithm is proposed to generate all activity paths. Finally, test cases are generated using activity path coverage criteria. Then Case study is being presented on Soft drink Vending Machine (SVM).

Abinash Tripathy and Anirban Mitra **"Test Case Generation Using Activity Diagram and Sequence Diagram"** , presented an approach to generate test cases by using together UML Activity diagram and Sequence Diagram [3]. In this approach first the activity diagram is being converted into activity graph (AG) and the sequence diagram is being converted into sequence graph (SG) and then the two graphs SG & AG are integrated to form system Graph (SYG). Then the System Graph (SYG) is being traversed to form the test cases by using an Graph optimization technique known as Depth First Search Method (DFS).This approach is also applied on an example of ATM card validation. It has been shown that the test cases obtained in this method are not only exhaustive but also optimal but how it is not clear. Also whenever two UML methods are combined it will cover all the possibilities. Activity diagram also solves the problem of concurrent execution problem which leads to state explosion problem.

Supaporn Kansomkeat, Phachayanee Thiket and Jeff Offutt **"Generating Test Cases from UML Activity Diagrams using the Condition-Classification Tree Method"** have focused on a UML diagram called an activity diagram[4]. They have used Condition Classification Tree method to generate test cases from activity diagram. In their paper, they provided a method to automatically gather control flow information from decision points and guard conditions in activity diagrams. This information is used to construct condition-classification trees. These trees are then used to generate a test case table and test cases. Experimental data show that tests generated by the CCTM Method have strong ability to detect faults at reasonable cost and also early in development. The case study is being done on SALES example.

Xiajiong Shen and Qian Wang PeipeiWang and Bo Zhoupropose, **"A Novel Technique Proposed for Testing of Object Oriented Software Systems"**, have proposed a novel technique [5] for testing of object oriented software systems that use the profile and UML state diagrams that all methods in a system are divided into different grades according to integrated values (frequency and significance) and then the methods that obtain the highest integrated value generate test cases from UML state diagrams. To prove the efficiency of the approach, the technique is compared with testing all methods only using UML state diagrams. Finally results prove that the approach is efficient for object-oriented software systems, and find faults that are difficult to find in other ways and reduce the cost of testing dramatically.

Fanping Zeng, Zhide Chen, Qing Cao, Liangliang Mao, **"Research on Method of Object-Oriented Test Cases Generation Based on UML and LTS"[6]** have presented a new technique for object oriented test case generation based on UML state diagrams and Label Transition System(LTS). Test cases are generated from UML state diagrams model that represent state transition. UML state diagram can be a model of modeling software system. It shows all kinds of possible states of a specific object and all the possible changes between states which cause by all kinds of events. Labeled Transitions System (LTS) model is an internal model to precisely represent the state transition. The procedure is based on model-based testing techniques with test cases generated from UML state diagrams translated into LTS.

Nirmal Kumar Gupta, Mukesh Kumar Rohil, **"Improving GA based Automated Test Data Generation Technique For Object Oriented Software"** [7]. This paper put forward a strategy for evaluating the fitness of both feasible and unfeasible test cases leading to the improvement of evolutionary search by achieving higher coverage and evolving more number of unfeasible test cases into feasible ones. The proposed strategy shows that genetic algorithms are useful in reducing the number of unfeasible test cases by generating test cases for object oriented software. Furthermore, we build our Genetic Algorithm for structural testing for generating more suitable test cases. In path testing weight reevaluation strategy is employed to develop unfeasible test cases into feasible test cases at the later generations.

Esmaeil Mirzaeian, Samad Ghaderi Mojaveri, Homayun Motameni, Ahmad farahi, **"An optimized approach to generate object oriented software test case by Colored Petri Nets"[8].** This paper put forward a technique for generating object oriented test cases using Colored Petri Nets extended version of Petri Nets usually used to system modeling and

simulation. They have introduced a new algorithm for to convert UML State chart into CPNs. This method considers net explosion problem and also generated net covers all instances of objects from different classes in the same hierarchy. At last a case study is also shown by a Banking account Example to show the benefits of the approach.

Rajiv mall, **"Automatic Test Case Generation From UML models" [9].** This paper proposed an algorithm, to generate test case from a combination of use case diagram and sequence diagram. First, they convert the use case diagram into use case graph and then sequence diagram into sequence graph. After that the two graphs are integrated and a system graph is generated. That system graph is traversed but not clearly mentioned.

Mahesh Shirole, Rajeev Kumar, "**A Hybrid Genetic Algorithm Based Test Case Generation Using Sequence Diagrams**"[10]. This paper presented a hybrid approach of generating test cases using sequence diagram with genetic algorithm. Sequence diagram shows the method call dependencies that exist among the methods that potentially appear in a method call sequence, which is good for integration testing. Test cases generated using genetic algorithm improves the method coverage as well as exception coverage.

Mahesh Shirole, Rajeev Kumar, "**UML behavioral model based test case generation: a survey" [11]** The objective of this paper is to improve the understanding of UML based testing techniques. It has focused on test case generation from the behavioral specification diagrams, namely sequence, state chart and activity diagrams. Also classified the various research approaches that are based on formal specifications, graph theoretic, heuristic testing, and direct UML specification processing and discussed the issues of test coverage associated with these approaches.

Yamina Mohamed ben Ali, Fatma Benmaiza, **"Generating Test Cases for Object-Oriented Software Using Genetic Algorithm and Mutation Testing Method."[12]** This paper presented an automatic creation of software test cases based on the use of a genetic algorithm and a mutation testing technique.

Philip Samuel, Rajib Mall, Pratush kant, **"Automatic test case generation from UML communication diagrams"[13].** This paper presented a method to generate cluster level test cases based on UML communication diagrams. In this approach, A tree representation of communication diagrams is being constructed. Then a post-order traversal of the constructed tree for selecting conditional predicates from the communication diagram is being carried out. The generated test cases achieve message paths coverage as well as boundary coverage. The technique is being tested on several example problems.

Ranjita Kumari, Vikas Panthi, Prafulla Kumar Behera, **"Generation of test cases using Activity Diagram"[14]** In this paper, test cases are generated using activity path coverage criteria. Here, a case study on Soft drink Vending Machine (SVM) has been presented to illustrate our approach.

Sujata khatri, R.S.Chillar, "**Generating Test Cases for Object Oriented Programs Using Specification based Testing Techniques**"[15] The aim of this paper is to examine the testing of object oriented software and to derive test cases using equivalence partitioning and boundary value analysis technique for triangle problem.

V.Mary Sumalatha, G.S.V.P.Raju, "**Object Oriented Test Case Generation Technique using Genetic Algorithms"[16]** This paper has been proposed to generate test cases for object oriented software using UML diagrams like Sequence diagram. Test cases are optimized using the Evolutionary Algorithm, Genetic Algorithm.

Hyungchoul Kim, Sungwon Kang, Jongmoon Baik, Inyoung Ko, "**Test Cases Generation from UML Activity Diagrams"[17]** This paper proposed a method to generate test cases from UML activity diagrams that minimizes the number of test cases generated while deriving all practically useful test cases.

Boghdady, P. , Badr, N. L., Hashem, M. A., Tolba, M. F., **"An enhanced technique for generating hybrid coverage test cases using activity diagrams"** [18]. This paper put forward an enhanced approach for automatically generating test cases from activity diagrams. Category partition method is applied to generate the final set of reduced test cases. The proposed model validates the generated test paths during the generation process to ensure that they meet a hybrid coverage criterion. The proposed model is automated and applied to around forty different case studies in different domains. Experimental evaluation is demonstrated to prove that the proposed model saves time and cost, thus increases the performance of the testing process.

Yvan Labiche "**Integration Testing Object-Oriented Software Systems: An Experiment-Driven Research Approach"[19]** has discussed about the questions : What integration testing process, indicating in which order classes are (Integration) tested, should be selected? Which test design techniques should be applied to unit and integration test classes when following an integration test order?

G.Suganya and S.Neduncheliyan has given the idea about trouble markers of object-oriented software and object-oriented testing techniques and specialized techniques for OO Environment in "A **Study of Object Oriented Testing Techniques: Survey and Challenges"[20].** They also discussed about how Unit Testing, Integration Testing and System Testing are being carried out in the Object Oriented environment.

Nagendra Pratap Singh, Mrinal Kanti Debbarma has explained about the Life Cycle of Object-Oriented Testing in "**The Review: Lifecycle of Object-Oriented Software Testing"** [21]. This cycle provide us a big point of view to test object-oriented software. Although it is not much differ from conventional testing but helpful for the thorough study of various approaches.

7. CONCLUSION AND FUTURE WORK

This paper provides a review of various test case generation techniques. The techniques generated using different UML Diagrams and using different algorithms and then they are automated . Also explains how these different techniques and Algorithms jointly play a vital role in reducing the errors during the Design Phase. Coverage Criteria is also different in different techniques. With the increase in demand for best quality of the product it is required to produce a better frame work for test case generation technique to work upon and ensure to remove maximum number of bugs. Future work may include integrating two or more techniques or improving the existing techniques.

8. REFERENCES

[1] http://my.safaribooksonline.com/book/software-engineering -and-development/software-testing.

[2] Ranjita Kumari Swain, Vikas Panthi, Prafulla Kumar Behera "Generation of test cases using Activity Diagram" International Journal of Computer Science and Informatics, ISSN (PRINT): 2231 –5292, Volume-3, Issue-2, 2013

[3] Abinash Tripathy and Anirban Mitra, "Test Case Generation Using Activity Diagram and Sequence Diagram" Proceedings of ICAdC, AISC 174, pp. 121-129. springerlink.com © Springer India 2013

[4] Supaporn Kansomkeat, Phachayanee Thiket and Jeff Offutt, "Generating Test Cases from UML Activity Diagrams using the Condition-Classification Tree Method" *2nd* International Conference on Software Technology and Engineering(ICSTE) @ 2010 IEEE.

[5] Xiajiong Shen and Qian Wang PeipeiWang and Bo Zhou, "A Novel Technique Proposed for Testing of Object Oriented Software Systems" @2009 IEEE.

[6] Fanping Zeng, Zhide Chen, Qing Cao, Liangliang "Research on Method of Object-Oriented Test Cases Generation Based on UML and LTS" .The 1st International Conference on Information Science and Engineering (ICISE2009) ©2009 IEEE computer Society

[7] Nirmal Kumar Gupta, Mukesh Kumar Rohil "ImprovingGA based Automated Test Data Generation Technique For Object Oriented Software**"** 2013 3rd IEEE International Advance Computing Conference (IACC) .

[8] Esmaeil Mirzaeian, Samad Ghaderi Mojaveri, Homayun Motameni, Ahmad farahi, "An optimized approach to generate object oriented software test case by Colored Petri Nets", 2010 2nd International Conference on Software Technology and Engineering (ICSTE) @ IEEE

[9] Monalisa Sharma,Rajib Mall, " Automatic Test Case Generation From UML models" The 10th International Conference on Information Technology@2007IEEE.

[10] Mahesh Shirole, Rajeev Kumar, "A Hybrid Genetic Algorithm Based Test Case Generation Using Sequence Diagrams" Contemporary Computing Communications in Computer and Information Science , Springer Verlag,Volume 94, 2010, pp 53-63

[11] Mahesh Shirole, Rajeev Kumar, "UML behavioral model based test case generation: a survey" ACM(DL), July 2013

[12] Yamina Mohamed ben Ali, Fatma Benmaiza, "Generating Test Case for Object-Oriented Software Using Genetic Algorithm and Mutation Testing Method " ACM(DL),2012

[13] Philip Samuel, Rajib Mall, Pratush kant, "Automatic test case generation from UML communication diagrams", ACM(DL)feb,2007

[14] Ranjita Kumari,Vikas Panthi, Prafulla Kumar Behera,"Generation of test cases using Activity Diagram" International Journal of Computer Science and Informatics, ISSN (PRINT): 2231 –5292, Volume-3, Issue-2, 2013

[15] Sujata khatri, R.S.Chillar, "Generating Test Cases for Object Oriented Programs Using Specification based Testing Techniques" Sujata Khatri et al / Indian Journal of Computer Science and Engineering (IJCSE), Feb-Mar 2012

[16] V.Mary Sumalatha, G.S.V.P.Raju, "Object Oriented Test Case Generation Technique using Genetic Algorithms"International Journal of Computer Applications (0975 – 8887) Volume 61– No.20, January ,2013

[17] Hyungchoul Kim, Sungwon Kang, Jongmoon Baik, Inyoung Ko, "Test Cases Generation from UML Activity Diagrams" © 2007 IEEE computer society

[18] Boghdady, P. , Badr, N. L., Hashem, M. A., Tolba, M. F., "An enhanced technique for generating hybrid coverage test cases using activity diagrams" Informatics and Systems (INFOS), 8th International Conference, 2012 in IEEE.

[19] Yvan Labiche "Integration testing object-oriented software systems: an Experiment-driven Research Approach", IEEE 24th Canadian Conference on Electrical and Computer Engineering,2011.

[20] G.Suganya and S.Neduncheliyan " A Study of Object Oriented Testing Techniques: Survey and Challenges", IEEE International Conference on Innovative Computing Technologies(ICICT)2010.

[21] Nagendra Pratap Singh and Mrinal Kanti Debbarma " The Review: Lifecycle of Object-Oriented Software Testing", IEEE 3rd International Conference on Electronic Computer Technology (ICECT), 2011 (Volume:3)

[22] John D.McGregor and David A.Sykes "A Practical Guide to Testing Object-Oriented Software" Addison–Wesley publications.

[23] Roger S.Pressman "Software Engineering –A Practitioner's Approach" McGraw Hill International Edition.

[24] W. Linzhang, Y. Jiesong, Y. Xiaofeng, H. Ju, L. Xuandong,and Z.Guoliang. "Generating test cases from UML activity diagram based on gray-box method. Proceedings of the 11th Asia-Pacific SoftwareEngineering Conference (APSEC04), pages 284 – 291, 2004.

[25] R. Mall. Fundamentals of Software Engineering. Prentice Hall, 3rd edition, 2009.[26]http://www.ambysoft.com/essays/floot.html

[27] Hongfang Gong Junyi Li, "Generating Test Cases of Object-Oriented Software Based on EDPN and Its Mutant" The 9th International Conference for Young Computer Scientists © 2008 IEEE

[28] Baikuntha Narayan Biswal, Pragyan Nanda, Durga Prasad Mohapatra "A Novel Approach for Scenario-Based Test Case Generation" International Conference On Information Technology © 2008 IEEE Computer Society