iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 5 **Issue:** V **Month of publication:** May 2017

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

Volume 5 Issue V, May 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Approaches for Software Testing Using UML

Md Khalid Hussain[1], Dr. Krishnanandan Prasad[2]
[1]Research Scholar, Inst. of Information Sciences & IT M. U. Bodh-Gaya, Gaya, Bihar, India
[2]Associate Professor, Deptt of Mathematics T. P. S. College Patna, Bihar, India

Abstract: ***this testing process is based on using the modeling techniques that are defined in Unified modeling language. Software Testing is a important role for software development .The software development are two important phases first is Designing and second is Testing Designing phase is done with the help of the UML models, which consists of the use case diagram, object diagram, sequence diagram, collaboration diagram, and activity diagram. One of the important criteria of testing is test case generation which describes tests that need to be run on the program to verify that the program runs as expected. This paper aim to explore few of the approach based on the UML Diagram to produce test cases for efficient software testing.***
Keyword: UML, Use Case Diagram, Object Diagram, Sequence Diagram and Activity Diagram.

I. INTRODUCTION

The Unified Modeling Language (UML) is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system [1]. Software testing is the procedures of exercise a program with well planned input data with the intent of observe failures. Testing identifies faults, whose removal increases the software quality by increasing the software's potential reliability. Software Testing also dealings the software feature in terms of its capability for achieve correctness, consistency, usability, maintainability, reusability and testability. The various Objectives of testing are as follows:

A. Software testing is a process of execute a plan with objective of finding a fault.
B. Software testing should also seek at signifying modifications if required, thus adding value to the entire process.
C. The objective is to design tests that thoroughly find out different classes of errors and do so with a minimum amount of time and endeavor.
D. Software testing requirements are required as it specified in specification document.
E. Software dependability and software quality based on the data together during testing.

II. UNIFIED MODELING LANGUAGE

UML is not a development method by itself;[2] however, it was designed to be compatible with the leading object-oriented software development methods of its time, for example OMT, Booch method, Objector and especially RUP that it was originally intended to be used with when work began at Rational Software. The set of diagrams need not completely cover the model and deleting a diagram does not change the model. The model may also contain documentation that drives the model elements and diagrams UML used during software development by specify, visualize, and document models of software systems, including their structure and design [3].The structural view includes class diagrams and composite structure diagrams and the dynamic view includes sequence diagrams, activity diagrams and state machine diagrams [4]

III.USE CASE DIAGRAM BASED TESTING

Using test cases based on use cases and are referred as scenarios with capability to identify gaps in the system which would not be found by testing individual components in isolation [5]. Adriana Carniello et al. [6] introduced a novel set of testing criteria based on the structure of use case diagrams. The structure of the use case diagram is defined by the relationships it contains, namely, association, include and extend relationships. They designed Use Case Tester (UCT) tool determines the new criteria test requirements, emulates the use cases behavior and analyzes the test coverage with respect to the criteria for a given use case diagram. As UCT emulates the behavior of use cases it can be utilized to simulate tests cases execution as well to validate requirements with the final user. UCT's novelty resides in this aspect, particularly useful in requirements engineering. They defined testing criteria based on the relationships which are All- Associations-Inclusions-Extensions Criterion (c1) and All-extended-combinations Criterion (c2). Criteria c1 suggest that given a test set T and a use case diagram D, for each use case in D extended by at least two other use cases, T must cause all the combinations of exercising and non-exercising the extend relationships to be exercised at least once. Criteria c1 suggest that given a test set T and a use case diagram D, T must cause each association, include

and extend relationship in D to be exercised at least once.

IV. SEQUENCE DIAGRAM BASED TESTING

Ashalatha Nayak et al. [7] propose an approach of synthesizing test data by using the attributes and constraint information associated with the model elements such as class diagrams, sequence diagrams and Object Constraint Language (OCL) constraints. By using this derived information they augment the sequence diagram and then map it onto a structured composite graph (SCG) from which the test specifications are generated. For each test specification, they follow a constraint solving system to generate test data. The test data generated using three steps: deriving constraints for the specified scenario, solving the constraints along the scenario and then generating test data for finding test input to the variables involved in the scenario.

V. COLLABORATION DIAGRAM BASED TESTING

A collaboration diagram describes interactions among objects in terms of sequenced messages. This type of software testing is used to send and receive message. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system. Collaboration diagrams provide the following six pieces of information wiz are: objects, sequences of operation, the semantics of an operation, imported operation from other classes, the communication pattern of objects and the execution characteristics of objects.

VI. ACTIVITY DIAGRAM BASED TESTING

The Activity Diagram can help to describe the flow of control of the target system, such as the exploring complex business rules and operations, describing the use case also the business process. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Debasish Kundu [8] proposed approach to generate test cases from an activity diagram by using following three steps: augmenting the activity diagram with necessary test information, converting the activity diagram into an activity graph, and generating test cases from the activity graph. Their approach is significant due to following reasons: it is capable to detect more faults like faults in loop, synchronization faults than the existing approaches; test case generated may help to identify location of a fault in the implementation, thus reducing testing effort; it inspires developer to improve design quality, find faults in the implementation early, and reduce software development time; and finally it is possible to build an automatic tool which will reduce cost of software development and improve quality of the software. Dong Xu et al. [9] propose an automated approach using adaptive agent‟s exploration to directly generate test scenarios from the activity diagrams. When using the proposed agents to search an activity diagram, special attentions need to be paid to two types of nodes, final nodes and branch nodes, in an activity diagram. They define a scenario tree as an interim storing structure. The initial node of an activity diagram is the root of the tree while each tree node denotes the elements of the activity diagram such as the action states, the forks, the joins, the branches, the merges, the guard conditions, and the final states. Test scenarios can then be derived from the scenario tree by traversing the tree. SG Shukla et al. [10] proposed an approach for generating the test cases from activity graphs. Activity graph is a directed graph and transformed form of activity diagram where each node represents a construct and each edge represents the flow. Activity diagram are augmented with the necessary test information. Test cases generated from the activity graph by following the proposed test coverage criterion. Their work considers concurrent activity path for an activity diagram that contains concurrent activities. For effective testing with limited resource and time, they test only relative sequence of the concurrent and non-concurrent activities that is, set of precedence relations exist among these activities. The generation of entire set of concurrent activity paths by finding representative concurrent activity path from activity graph. Instead of deriving test cases from the activity diagram directly, an indirect approach is suggested in their approach which selects the test cases from the set of the randomly generated test case according to a given activity diagram. Then, by running the program with the generated test cases, they get the corresponding program execution traces.

VII. CONCLUSIONS

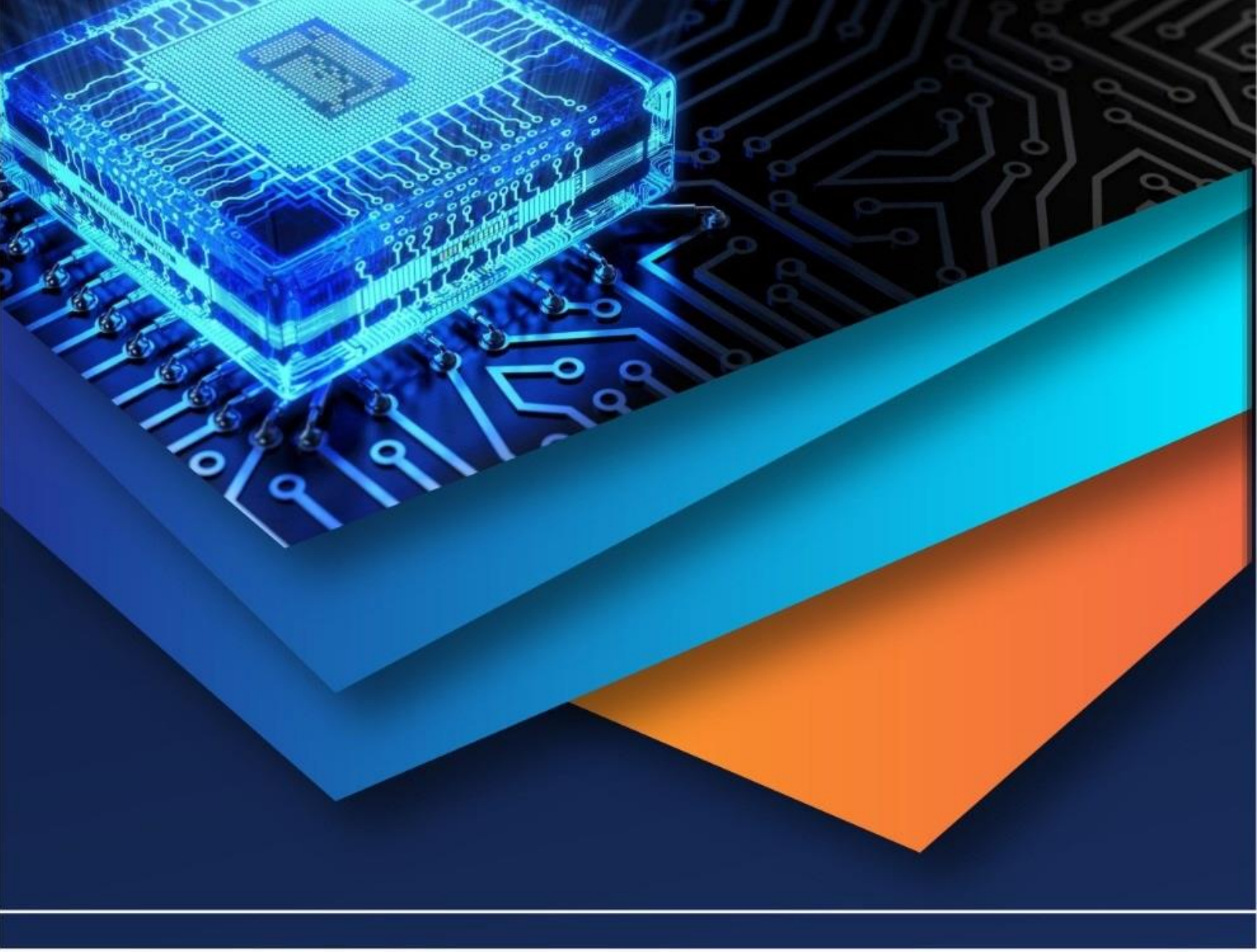
Software testing is very important software development. So there is a need to automate software testing process by using UML that incorporates model based testing. In this paper, different models and approaches design to generate test cases or testing approached based on the Unified Modeling Languages has been described. Every come near has advantages as well as disadvantages in different programs of software testing.

REFERENCES

[1] https://en.wikipedia.org/wiki/Unified_Modeling_Language#History
[2] John Hunt (2000). The Unified Process for Practitioners: Object-oriented Design, UML and Java. Springer, 2000. ISBN 1-85233-275-1. p.5.door

Volume 5 Issue V, May 2017
ISSN: 2321-9653

[3] A.V.K. Shanthi and G. Mohan Kumar, “Automated Test Cases Generation from UML Sequence Diagram”, International Conference on Software and Computer Applications (ICSCA 2012) IPCSIT vol. 41, IACSIT Press, Singapore.
[4] http://www.omg.org/gettingstarted/what_is_uml.htm (Access on 22nd Feb, 2014)
[5]]http://en.wikibooks.org/wiki/Introduction_to_Software_ Engineering/UML (Access on 22th Feb, 2014)
[6] Adriana Carniello, Mario Jino, Marcos Lordello Chaim, “Structural Testing with Use Cases”, JCS&T Vol. 5 No. 2 August 2005
[7] Ashalatha Nayak, Debasis Samanta: "Automatic Test Data Synthesis using UML Sequence Diagrams", in Journal of Object Technology, vol. 09, no. 2, March-April 2010, pp. 75-104.
[8] Debasish Kundu, Debasis Samanta, "A Novel Approach to Generate Test Cases from UML Activity Diagrams", in Journal of Object Technology, Vol. 8, No. 3, pp.65 -83, May-June 2009.
[9] Dong Xu, Huaizhong Li, Chiou Peng Lam, “Using Adaptive Agents to Automatically Generate Test Scenerios from the UML Activity Diagram”, IEEE Proceedings of the 12th Asia-Pacific Software Engineering Conference (APSEC˝05), 2005
[10] Sandeep Girijashankar Shukla, Gajendra Singh Chandel, “A Systematic Approach for Generate Test Cases Using UML Activity Diagrams”, IRACST-International Journal of Research in Management & Technology (IJRMT), Vol. 2, No. 4, August 2012



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY