IJITKMSpecial Issue (ICFTEM-2014) May 2014 pp. 14-18 (ISSN 0973-4414)

Simulation of Software Modules-Using PERT

Er. ShwetaSharma[1],Dr.RajeshGarg[2], Er.Richa Grover[3]
[1,3] Ganpati Institute Of Technology & Management ,
[2]Seth Jai Prakash Polytechnic Damla,
[1]shweta.ganpati@gmail.com, [2]rajesh_damla@yahoo.co.in, [3]richagrover01@gmail.com

Abstract: Testing is an essential activity in software engineering. In the simplest terms, it amounts to observing the execution of a software system to validate whether it behaves as intended and identify potential malfunctions. Testing is widely used in industry for quality assurance: indeed, by directly scrutinizing the software in execution, it provides a realistic feedback of its behavior and as such it remains the inescapable complement to other analysis techniques. In this paper we will discuss about all the faces and techniques of testing from unit testing to acceptance testing. We will also describe about the use of simulation in testing software modules. Simulating a particular software using Program evaluation and Review technique is efficient technique for software testing. By identifying the critical activities and events in the development of project, we can successfully execute the product to be released.

Index Terms: Software Engineering, Software Testing, Testing Techniques, Simulation, Software Quality.

1. INTRODUCTION

Process of creating program consist of five phases (i) defining a problem (ii) designing a program (iii) building a program (iv) analyze performance of program (v) finally, arranging of product. But software testing is component of third phase. Thus, software testing is defined as the process of executing the software for expected outcomes. The goal of testing is to find errors and the good test case is the one that has maximum probability of finding errors. It amounts to observing the execution of a software system to validate whether it behaves as intended and identify potential malfunctions. Testing is an activity performed for evaluating software quality and improving it.[5] Testing objective is to find out maximum errors with minimum time and efforts. Software testing, as a practice, has been able tosuccessfully evolve over time and provide efficient andconstant support for improvements in software quality. Onthe other hand, testing is still notorious for its massiveresource consumption within software projects. To this date,much of the research efforts on software testing have beenfocusing on designing new techniques. However, during the entire history of software development, testing methods and techniques have struggled to keep up with the ever faster evolution and trends in software development paradigms. There are various methods and techniques of software testing i.e Black box and White Box testing. Although these techniques are widely adopted, but they consume 70 to 80% time and efforts in SDLC [1]. So, there is a requirement for development of some new techniques, hence we can go for testing software modules using simulation. Simulation is a powerful technique for solving a wide variety of problems. Peoples follow many methods for testing but few of them are time consuming, require lots of experience, so using simulation is faster and much accurate method to find faults in software modules. Simulation is used to understand and evaluate the functionality and performance of complex inter-component protocols and algorithms. Typically, they are written in an imperative programming language, such as C++ or Java.

2. METHODOLOGY

TESTING METHODS

Test cases are developed using various testtechniques to achieve more effective testing. Bythis, software completeness is provided andconditions of testing which get the greatestprobability of finding errors are chosen. So,testers do not guess which test cases to choose,and test techniques enable them to designtesting conditions in a systematic way. Also, combining good techniques yields better results then using single method. Testing consist of two methods: Static and Dynamic Testing.

A. Static Testing Techniques:

Stress Testing or Torture Testing: Testing which is done to determine stability of given system. It involve the testing beyond normal operational capacity until breaking point in order to observe the results. .

Load Testing: The testing which conducted to determine system's behavior under both normal and peak load conditions.

Regression Testing: The testing which is done to ensure that changes added in software does not introduce any new faults. It is retesting of new software.

Functional Testing: The testing which is conducted on a complete and integrated system by feeding them input and examine the output and in which internal program structure is rarely considered. It defines 'what' the system does.
Unit Testing: It is a method in which individual unit of source code (software modules) are tested. Its goal to ensure that each and every modules of software are working correctly
Performance Testing: The testing which is done to check the stability and responsiveness of software under particular workload.
Acceptance Testing: It is the testing which is to be done to determine that the requirement specification of end user are satisfied or not.
Security Testing: The testing which determines that an Information System protects data and maintains functionality as intended .

B. Dynamic Testing:
Software can be tested in two ways, in another words, one can distinguish two different methods:
1. Black box testing, and
2. White box testing.

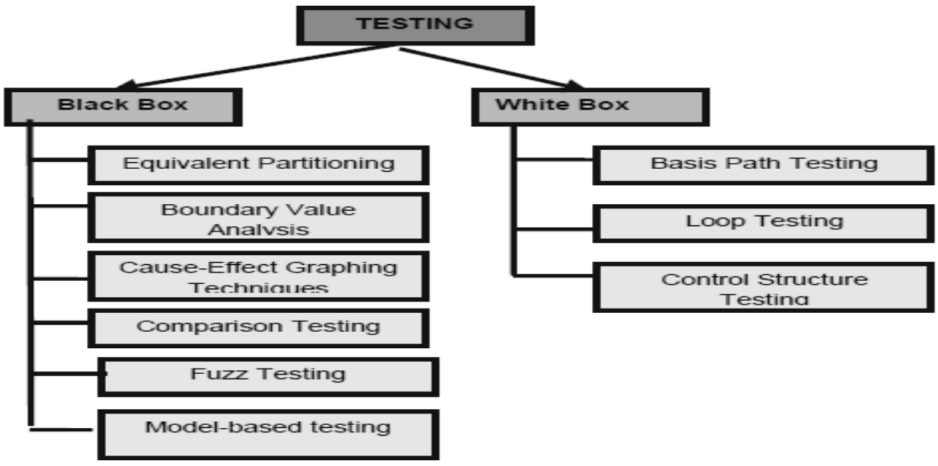

Fig.1 Classification of Testing Techniques

White box testingis highly effective in detecting and resolving problems, because bugs can often be found before they cause trouble. We can shortly define this method as testing software with the knowledgeof the internal structure and coding inside theprogram. White box testing is also called white box analysis, clear box testing or clear box analysis. It is a strategy for software **debugging** inwhich the tester has excellent knowledge of howthe program components interact. This method can be used for Web services applications, and is rarely practical for debugging in large systems and networks.

Black box testing is testing software based on output requirements and without any knowledge of the internal structure or coding in the program. In another words, black box is any device whose workings are not understood by or accessible to its user. A black box is an algorithm that doesn't provide an explanation of how it works. The third testing method has been also considered – gray box testing. It is defined as testing software while already having some knowledge of its underlying code or logic. It is based on the internal datastructures and algorithms for designing the testcases more than black box testing but less thanwhite box testing. This method is importantwhen conducting integration testing between twomodules of code written by two differentdevelopers, where only interfaces are exposedfor test.

FURTHER CLASSIFICATION OF TEST METHODS
In this paper, the most important test techniques are shortly described .Figure 1. shows general classification of testtechniques

(1)Equivalence Partitioning
This technique divides the input domain of a program onto equivalence classes. Equivalence classes – set of valid or invalid states for input conditions, and can be defined in the following way (a) An input condition specifies a range

(b)An input condition needs a specific value (c)An input condition specifies a member of a set (d) An input condition is Boolean value, using this technique, one can get test cases which identify the classes of errors.

(2)Boundary Value Analysis
This technique is like the Equivalence Partitioning, except that for creating the test cases beside input domain use output domain. One can form the test cases in the following way :(1) An input condition specifies a range bounded by values. (2)An input condition specifies various values.(3) Rules 1 and 2 apply to output and the conditions; If internal program data structures have prescribed boundaries, produce test cases to exercise that data structure at its boundary.

(3)Cause-Effect Graphing Techniques
One uses this technique when one wants to translate a policy or procedure specified in a natural language into software's language.

(4)Comparison Testing
In situations where reliability of software is critical and redundant softwaresare produced we use this technique. Software engineering teams produce independent versions of an application, each version can be tested with the same test data, so the same output can be ensured.

(5) Model-based testing
Model-based testing is automatic generation of efficient test procedures/vectors using models of system requirements and specified functionality .In this method, test cases are derived in whole or in part from a model that describes some aspects of the system.

(6)Basis Path Testing
If one uses this technique, one can evaluate logical complexity of procedural design. After that, one can employ this measure for description basic set of execution paths .For obtaining the basis set and presentation control flow in the program, one uses flow graphs **.**Main components of that graphs are: Node – it represents one or more procedural statements. Node which contains a condition is called predicatenode. Edges between nodes – represent flow of control. Each node must be bounded by at least one edge, even if it does not contain any useful information .Region– an area bounded by nodes and edges.

Cyclomatic Complexity is software metric. The value evaluated for cyclomatic complexity defines the number of independent paths in the basis set of a program .Independent path is any path through a program that introduces at least one new set of processing statements. For the given graph G, cyclomatic complexity V(G) is equal to:

1. The number of regions in the flow graph;
2. **V(G) = E - N + 2**, where E is the number of edges, and N is the number of nodes;
3.**V(G) = P + 1**, where P is the number of predicate nodes.

3. TESTING USING SIMULATION

Today lot many of websites & software's are using testing techniques to make their modules free from bugs and faults. Simulation is a powerful technique for solving a wide variety of problems. To simulate is to copy the behavior of system or phenomenon under study. Peoples follow many methods for testing but few of them are time consuming, require lots of experience, so using simulation is faster and much accurate method to find faults in software modules.[4] Simulation is used to understand and evaluate the functionality and performance of complex inter-component protocols and algorithms. Typically, they are written in an imperative programming language, such as C++ or Java. Simulation allow the developer to capture basic algorithmic functionality at the same time as they focus attention on topology, timing, bandwidth, overall scalability and other properties characteristics of distribution.[6] In this paper we have discussed how simulation helps in improving Software Quality by identifying and improving critical activities in the software.
Simulations embody abstractions for the underlying mechanisms and environmental conditions that affect these properties, providing parameters for exploring the functionality and performance space. Unlike many other development artifacts Simulation seem to be used, and therefore well maintained, throughout the development both as early design tools and as late evaluation tools.

Simulating the software using Program evaluation and Review Technique (PERT) and Critical Path Method (CPM) two management science techniques developed in the late 1950s to plan, schedule, and control large, complex projects with many activities. These approaches differ primarily on how the duration and the cost of activities are processed. In the case of CPM, it is assumed that details about these inputs are known with certainty, whereas for PERT, these details are not known with certainty.
Simulation Based Testing is used within a simple and generic testing process. As a First step, the developer assembles a test suite. As usual, a test suite is composed of test cases, each one consisting of an input vector that includes direct inputs to the system, representing functional parameters, as well as inputs to the environment, representing environmental conditions. [7]Then, the developer determines whether the suit is adequate and if it is, they use it to test the implementation.

The virtual world Simulation is difficult to get use to the first time you use it for design, but after that the sky isn't even your limit.

Software engineering comprehends several disciplines devoted to prevent and remedy malfunctions and to warrant adequate behavior. Developers of Distributed Systems routinely construct discrete-event simulations to help them understand and evaluate the behavior of inter-component protocols. Can simulations also be used to help in the testing of distributed system implementations? Because simulations amount to specifications of intended behavior, the code of a simulation can be viewed as an operational, albeit non-traditional, formal model. If we can simulate distributed System then, we can also use it for making our modules in software error free. This new technique if used properly will produce efficient and accurate results.
There are number of simulation and analytical methods which are used for project development and verification. For example Physical, Mathematical models, Monte Carlo simulation, stochastic simulation, Discrete Event simulation, and Continuous Simulation Techniques. Out of which we will use Monte Carlo Method of simulation. The term was Monte- carlo computation was coined during World War II by S. Ulam and J.von Neumann at Los Almos Scientific Laboratory. Some people use Monte Carlo method to mean any computation that involves random numbers. However, a more generally accepted meaning of Monte Carlo is restricted only to those computations in which random numbers are used to obtain solutions of problems which are inherently deterministic. And for those computations that employ random numbers to solve inherently stochastic problems the term ' stochastic Simulation' is used.
Simulation is basically an experimental technique. It is fast and relatively inexpensive method of doing an experiment on the computer. There is no unifying theory of computer simulation. Learning simulation does not consist of learning a few fundamental theorems and then using them and their various corollaries to solve problems. There are no underlying principles guiding the formulation of simulation models. Each application of simulation is ad hoc to great extent. Therefore, we can say simulation is an art, and one often hears the expression **'the art of simulation'**.
Thus simulation is very powerful, problem –solving technique. The simplicity of approach when combined with the computational power of the high speed digital computer makes simulation a powerful tool. Popularity and usefulness of simulation is dependent on the capabilities and availability of the computer. Occasionally, simulation is also used even when an exact analytic solution is possible, but it is too expensive in terms of computation time. Today, simulation is used in science and engineering research, soft sciences like psychology, biology, economics, for business executive. It is widely used for inventory control, facility planning, production scheduling etc.

4. CONCLUSION

Software testing is a component of software quality control (SQC). SQC means control thequality of software engineering products, whichis conducting using tests of the software system .we are using above mentioned testing techniques to enhance the software quality and making it fault free. But many of them are time consuming,require a lot of experience ,so we can go for another reliable method for testing i.e. testing based on simulation described in this paper. It is an inexpensive and efficient method that gives fruitful results if implemented properly.

References

[1] Antonia Bertolino, "Software Testing Research and Practice", Italy
[2] Jovanovic, "Software Testing, Methods and techniques", published on May 26, 2008
[3] Zhang Hongchun ,**"**Research on New Techniques and Development Trend of Software Testing" in Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013).
[4] Thomas J.Ostrands ,ElainJ.Weiyuker " Software Testing research and software Engineering Education" published in 2010 at AT and T-Labs.
[5] Antonia Bertolino, "Software Testing Research: Achievements, Challenges, Dreams", Future of Software Engineering (FOSE'07) 0-7695-2829-5/07 $20.00 © 2007 IEEE
[6] Matthew J. Rutherford, Antonio Carzaniga, and AlexenderL.Wolf," Simulation-Based Testing of distributed Systems" University of Colorado, Department of computer science, technical Report CU-CS-1004-06 January 2006
[7] Mario Vanhouke," An Overview of recent Research results and future Research Avenues Using simulation Studies in Project Management".
[8] K.Tatsumi, S.Watanabe, Y.Takeuchi and H.Shimokawa," Conceptual support for test Case Design", Computer Software Development Group, Numazu Shizuoka, 410-03 Japan
[9] A Practitioner's Guide to software Test design, Lee Copeland, 2003
[10] The Art of Software Testing, 2nd edition, Glenford Myers, et.et, 2004
[11] Software Engineering A Practioner's Approach Sixth Edition, Roger S.Pressaman, 2005
[12] Software Testing Techniques, 2nd Edition, Boris Beizer, 1990
[13] Testing Object- Oriented Systems: Models, Patterns, and Tools, Robert V. Binder, 1999
[14] Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/wiki/
[15] Stracey, D.A., " Software Testing technique"
[16] Guide to the software engineering body of knowledge, Swebok- A Project of IEEE Computer society Professional Practice Committee, 2004.