RESEARCH ARTICLE OPEN ACCESS

A Review on Software Testing Methodology

Khushboo Pandey
Department of Information Technology,Mahakal Institute Of Technology Ujjain

Abstract
Software engineering played important role in today's life.Now in these days in our daily life we encountered with various software tools and applications these applications are provide ease in business invigilation, banking and other domains. These applications are results of effort of software engineering;any quality software application development needs a large amount of planning, team work and money. Thus to maintain the quality of software, testing is an optimal way to produce a quality software application.There are various testing tools and techniques are previously developed. This paper providesa study about various testing methodologies. On the basis of performance of methodologies adopted previously a review of about testing methodology is given in this paper.

Keywords—software engineering, testing, testing methodology.

I. INTRODUCTION

Software engineering is a process stack, when after finalizations of one process, other process takesplace; in software engineering these processes work in an ordered way to achieve quality software. Given diagram (figure 1) shows software process stack, where first requirement is gathered from client, in this step a team of software engineers negotiating with clients to get the real world challenges, these challenges are analysed during next analysis process where feasibility of the problem is estimated. After analysing the requirements and challenges some solution steps and designs are prepared. After finalizing the design part this solution are implemented using codes and techniques. Then after the quality of designed software is measured using various testing tools and methodologies. After testing if any bug or problem is found than bugs are fixed and prepared for deployment.

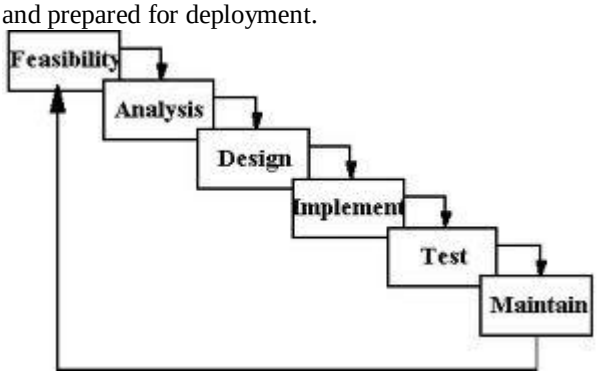

Figure 1 shows the process model

Software Testing is the process of executing a program for finding errors. Or, it involves any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. Software is not unlike other physical processes where inputs are received and outputs are produced. Where software differs is in the manner in which it fails. Most physical systems fail in a fixed (and reasonably small) set of ways. Software can fail in many bizarre ways. Detecting all of the different failure modes for software is generally infeasible. [1]

Thus testing is important and responsible process in this process stack. In this paper we are focus on various techniques and methods that are previously adopted and implemented, in the next section of the paper we introduces some article and tools.

II. BACKGROUND

Software testing is a process of verifying and validating that a software application or program [2]

1. Meets the business and technical requirements that guided its design and development, and
2. Works as expected.

Software testing also identifies important defects, flaws, or errors in the application code that must be fixed. There are various testing tools and methodologies are found during the study, some of them that are provides us guidelines are given in this section.

An Evaluation of Test Coverage Tools in Software Testing is provided by ***[Muhammad 2011]*** in this article Tools like,JavaCodeCoverage, JFeature, Bullseye, Clover, Cobertura, Emma, eXVantage, JCover, Gretel, Code Cover are analysed.According to their analysis each tool has some strong and weak points. Users anddevelopers can select the tool according to their need.

Various tools are evaluated in [3], and a way to select an tool on the basis of the tools characteristics is summarized in the given article, but testing and its strategy is also decided by the utilization of the current software scenario and depends upon who want to use testing is also an

effective parameter in software testing. A similar research work is given by ***[Mika 2011]***this research empirically investigates how testing involvesemployees in varying organizational roles in software product companies. They studied theorganization and values of testing using an exploratory case study methodology through interviews,defect database analysis, workshops, analyses of documentation, and informal communicationsat three software product companies. [4]Analysed which employee groups testsoftware in the case companies, and how many defects they find. Two companies organizedtestingasateameffort,andonecompanyhadaspecializedtestinggroupbecauseofitsdifferentdevelopmentmodel. They foundevidencethattestingwasnotanactionconductedonlybytestingspecialists.Testingbyindividualswithcustomercontactanddomainexpertisewasanimportantvalidation method. We discovered that defects found by developers had the highest fix rateswhilethoserevealedbyspecializedtestershadthelowest.Thedefectimportancewassusceptibletoorganizationalcompetitionofresources(i.e.,overvaluingdefectsofreporter'sownproductsorprojects).Weconcludethatitisimporanttounderstandthediversityofindividualsparticipatinginsoftwaretestingandtherelevanceofvalidationfromtheendusers'viewpoint. Futureresearchis required to evaluate testing approaches for diverse organizational roles. Finally, to improvedefect information, author suggests increasing automation in defect data collection.

After analysing the role based system and case study we discuss about various testing methodologies.

Random testing is a useful testing technique in itself. It also plays an important role in other testingmethods too. Therefore, any significant improvement to random testing has an impact throughout the software testingcommunity. Recently, **Adaptive Random Testing** (ART) was proposed [5]which is an effective alternative to random testing.This paper presents a synthesis of the most important research results related to ART. Here author have realised how the techniques and concepts of ART can be applied in a muchbroader context. Author believes such ideas can be applied in a variety of areas of software testing,and even beyond software testing. Amongst these ideas, particularly note the fundamental role of diversity in testcase selection strategies.

Many software testing problems involve sequences of events. This paper [6] applies **combinatorial methods** to testing problems that have n distinct events, where each event occurs exactly once. The methods described in paper is motivated by testing needs for systems that may accept multiple communication or sensor inputs and generate output to several communication links and other interfaces, where it is important to test the order in which events occur. Although pairwise event order testing (both A followed by B and B followed by A) has been described, our method ensures that any t events will be tested in every possible t-way order.

Search-based techniques have been shown usefulfor the task of generating tests [7], for example in the caseof object-oriented software. But, as for any meta-heuristicsearch, the efficiency is heavily dependent on many differentfactors; seeding is one such factor that may strongly influencethis efficiency. In this paper, author evaluates new and typicalstrategies to seed the initial population as well as to seedvalues introduced during the search when generating tests forobject-oriented code. They report the results of a large empiricalanalysis carried out on 20 Java projects (for a total of 1,752public classes). In given experiments show with strong statisticalconfidence that, even for a testing tool that is already able toachieve high coverage, the use of appropriate seeding strategiescan further improve performance.

To assess the quality of test suites, mutation analysis seedsartificial defects (mutations) into programs; a non-detectedmutation indicates a weakness in the test suite. Author, [8] presentan **automated approaches** to generate unit tests that detectthese mutations for object-oriented classes. This has two advantages: First, the resulting test suite is optimized towardsfinding defects rather than covering code. Second, the statechange caused by mutations induces oracles that preciselydetect the mutants. Evaluated on two open source libraries, our µtest prototype generates test suites that find significantly more seeded defects than the original manuallywritten test suites.

In this section we learn about various techniques and previously made efforts in the domain of software testing. Which involve various methods by which the test case writing is enhanced and implemented using automated and semi-automated testing tools? In the next section we discuss about the automated testing method and genetic algorithm based technique which provides guidelines to develop a new strategy for developing new technique.

III. LITERATURE STUDY

This section presents a brief analysis of theautomated testing method and genetic algorithm. Which describes how automated testing works and how genetic algorithm search solution in a problem space.

Automated testing: Automation testing which is also known as Test Automation is when the tester writes scripts and uses software to test, this process involves automation of a manual process, Automation Testing is used to re-run the test scenarios that were performed manually, quickly and repeatedly.Apart from regression testing, Automation testing is also used to test the application from load, performance and stress point of view. It increases the test

coverage; improve accuracy, saves time and money in comparison to manual testing.

It is not possible to automate everything in the Software; however the areas at which user can make transactions such as login form or registration forms etc., any area where large amount of users. can access the Software simultaneously should be automated.Furthermore all GUI items, connections with databases, field validations etc. can be efficiently tested by automating the manual process.

Automation is done by using a supportive computer language like vb scripting and an automated software application. There are a lot of tools available which can be used to write automation scripts. Before mentioning the tools lets identify the process which can be used to automate the testing: [10]

1. Identifying areas within a software for automation.
2. Selection of appropriate tool for Test automation.
3. Writing Test scripts.
4. Development of Test suits.
5. Execution of scripts.
6. Create result reports.
7. Identify any potential bug or performance issue.

Genetic algorithm: Genetic algorithm is one of the popular approaches for making search for large amount of data. The bio informatics knowledge is used to find the fittest answers in number of repetitive or iterative calculations. So first we discuss the primary functioning of the genetic algorithm. The evolutionary algorithms use the three main principles of the natural evolution: reproduction, natural selection and diversity of the species, maintained by the differences of each generation with the previous. [9]

Genetic Algorithms works with a set of individuals, representing possible solutions of the task, the selection principle is applied by using a criterion, giving an evaluation for the individual with respect to the desired solution. The best-suited individuals create the next generation.

Generate initial population –the algorithms in first generation randomly generated, by selecting the genes of the chromosomes among the allowed alphabet for the gene. Because of the easier computational procedure it is accepted that all populations have the same number (N) of individuals, Calculation of the values of the function that we want to minimize of maximize.

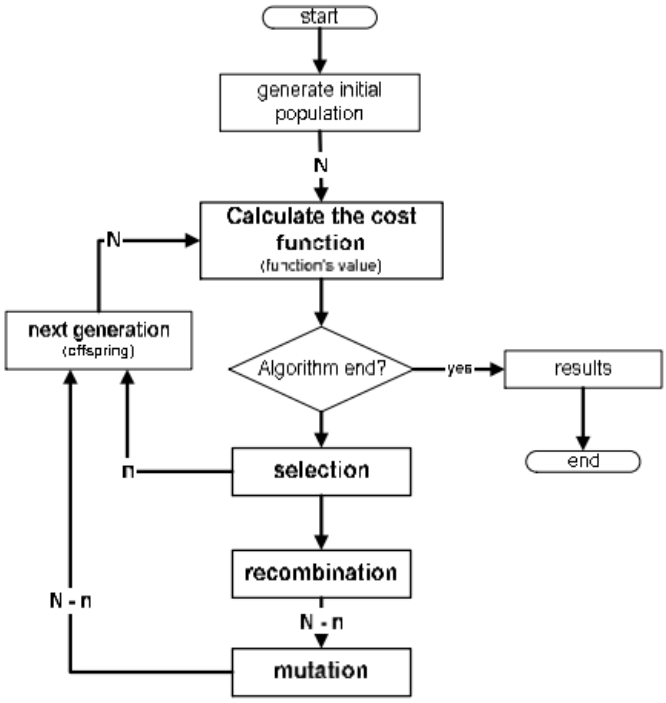

Fig shows the processing of genetic algorithm

Check for termination of the algorithm – as in the most optimization algorithms, it is possible to stop the genetic optimization by: Value of the function, Maximal number of iterations and Stall generation

Selection – between all individuals in the current population are chose those, who will continue and by means of crossover and mutation will produce offspring population. At this stage elitism could be used – the best n individuals are directly transferred to the next generation. The elitism guarantees, that the value of the optimization function cannot get worst (once the extreme is reached it would be kept).

Crossover – the individuals chosen by selection recombine with each other and new individuals will be created. The aim is to get offspring individuals, which inherit the best possible combination of the characteristics (genes) of their parents.

Mutation – by means of random change of some of the genes, it is guaranteed that even if none of the individuals contain the necessary gene value for the extreme, it is still possible to reach the extreme

New generation – the elite individuals chosen from the selection are combined with those who passed the crossover and mutation, and form the next generation.

IV. PROPOSED METHOD

In this section of the paper we describe our problem domain and solution steps adopted.Regression testing can be used not only for testing the appropriateness of a program, but often also for tracking the quality of its output. Some times when programs are in development stage, the old errors are come back and create problems during execution of program. So the technique required to test the progressive code and suggest a model to

minimize the error. We propose a technique to minimize the error in the progressive code model. In addition of that the manual testing consumes much time to write test cases and generating parameters for all individual classes methods. writing manual test cases for all the classes which exist in the source package is quite time consuming and requires more efforts, thus required to finding a new automated testing method by which overhead of writing test cases are reduced and in addition of that by auto generated parameters system measure quality of software.

In any product development quality of product depends upon testing and their test cases involved in evaluation of any product. In software engineering each product is too much sensitive, cost effective and requires a large amount of hard work to develop quality software, thus testing is an assurance of software product.

To overcome the above defined problem we propose a solution based on regression testing model. In our solution we involve bug and faults detections in the progressive code, which finds the code bugs and problems in the given source packages. In addition of that here we propose an automated testing tool which accepts only source code and write test cases self-using random walk algorithm. These generated test cases are consumes assert method to compare the desired and actual outcomes to compare with and evaluate the results.

We are going to implement a complete automated Regression testing tool, with defect minimizing technique. So that required to implement complete tool in some steps and each step produces some output after completion of steps. The outcome generated by the system is listed below.

1. Import code directory to test.
2. test all code correctness
3. perform code test one or more times
4. create output for each fault
5. produces approach to minimize error

The retesting of a software system that has been modified to ensure that any bugs have been fixed and that no other previously working functions have failed as a result of the reparations and that newly added features have not created problems with previous versions of the software. Also referred to as verification testing, regression testing is initiated after a programmer has attempted to fix a recognized problem or has added source code to a program that may have inadvertently introduced errors. It is a quality control measure to ensure that the newly modified code still complies with its specified requirements and that unmodified code has not been affected by the maintenance activity.

So we add the principal goals in our project.

1. test all source code
2. find bugs
3. get suggestion

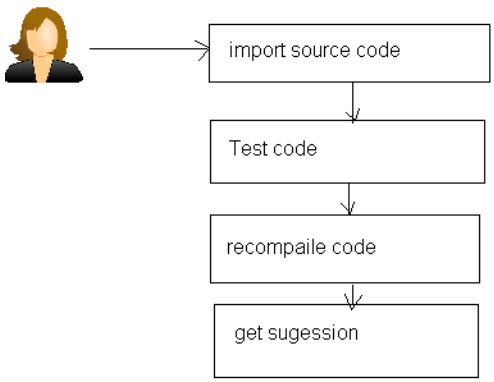

Fig shows the general concept of proposed tool

As above shown fig user interact with the system using these four phases.

1. **Import source code:** import source code directory to the system, in this phase of system processing system accepts user input and validates the java source code, in addition of that explore all the packages and sub packages of the input source code.
2. Test code: run test over mounted directory, once all the source package source code and classes are listed and validated correctly, system scan all source codes and classes to obtain main classes in packages. After finding main method system compile source directory to generate class files of all source code.
3. Recompile code: to get over all error summery and suggestions required to rerun the test. There system accepts previous stages outcomes and works with the random walk algorithm for generating input sequences for the methods containing class. Each class and methods are compiled for number of generations or iterations.
4. Got output: get final output to minimize error, system here generates the report of test conducted for the selected source code.

V. SYSTEM ARCHITECTURE

the proposed system to implement the desired system is given below the system execution starts with the input source phase where an actor import JAVA source code to test, the complete source directory can be imported for test. In the next section the system validates the code is JAVA files are not if source code contains the JAVA files system list them and method extraction is called for further process. Here methods and their input type and return types are evaluated for test case generation.

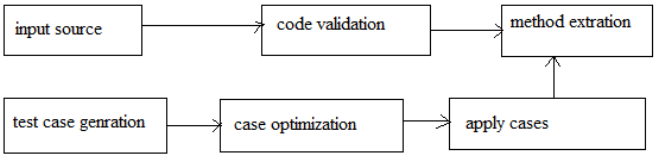

Fig shows the system architecture

In the background a random walk algorithm working to generate the test cases, the generated cases are optimized using optimization phases where test cases are filtered to obtain the most optimum test cases where the probability is higher to get the error from test cases the filter cases are than applied to the methods available and performance is calculated. In the next section we provide the complete system flow of the desired system.

In the conceptual view the overall system is working into two main modules first test case generation and second applies test cases to the classes. The complete work flow is given using the below given diagram. And the steps involve is given below.

1. Source code input: here user input is required to provide desired source code as input, this source code maybe inform of individual JAVA code file or a complete source directory. This code is provided as input for next phase.

2. Source code validation: in this phase system works on code and validate the codes are in java format or not if it is valid source code directory the process it else system message error and quite working.

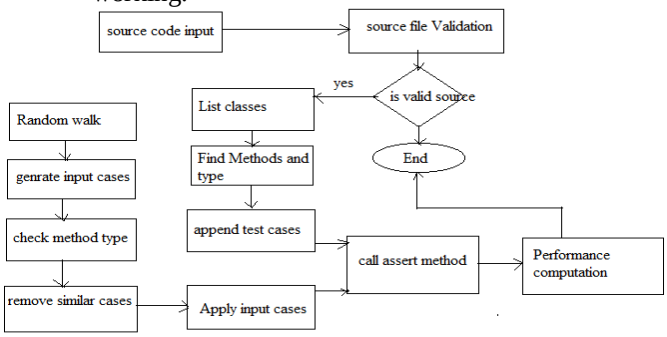

Fig shows the flow diagram

3. List classes: that is a list of all classes found under the input source directory and their sub-directories.

4. Find methods and type: here all the classes are analysed and methods and functions written under a class is listed first, then the input parameters are evaluated in terms of their data types and returned types.

5. Append test cases: system generates test cases for individual classes and append these cases into the end of class.

6. Random walk: to generate the input test parameters this algorithm is provides direction to build the string which is put into test cases and evaluation is performed. These generated input parameters are then forwarded into next phase.

7. Check method type: before applying the test parameters on a class system first find the type information for input and remove all instances of input parameters from the cases in next phase.

8. Apply input cases: all the input cases are now ready in a list to apply over the test cases.

9. Call asserts method: using this block the system applies parameters and validates the outcomes of the system.

10. Performance evaluation: based on the validation of test cases system generates the performance of the system.

VI. CONCLUSION AND FUTURE WORK

In this given paper we provide a conceptual model of automated testing tool which is derived using study of different testing methodologies and techniques that previously designed and implemented.This model consumes traditional assert method of manual testing, an intelligence algorithm of soft-computing. This paper includes simple illustration of proposed technique, in future we provide a detailed description of implementation of the proposed approach using java framework to test and write automated test cases.

REFERENCES

[1] Software Testing,Carnegie Mellon University,spring 1999,www.ece.cmu.edu/~koopman/des_s99/sw_testing/

[2] Software Testing Fundamentals—Concepts, Roles, and Terminology, John E. Bentley, Wachovia Bank, Charlotte NC,http://www2.sas.com/proceedings/sugi30/141-30.pdf

[3] An Evaluation of Test Coverage Tools in Software Testing,Muhammad Shahid, Suhaimi Ibrahim,Proc .of CSIT vol.5 (2011) © (2011) IACSIT Press, Singapore

[4] Who tested my software? Testing as an organizationallycross-cutting activity,Mika V. Ma¨ntyla¨, Juha Itkonen Joonas Iivonen,Published online: 21 August 2011

[5] Adaptive Random Testing: the ART of Test Case Diversity,Tsong Yueh Chena, Fei-Ching Kuoa, Robert G. Merkela, T.H. Tseb,Journal of Systems and Software. Copyright © Elsevier 2010

[6] Combinatorial Methods for Event Sequence Testing,D. Richard Kuhn, James M. Higdon, James F. Lawrence, Raghu N. Kacker, Yu Lei, 2012 - ieeexplore.ieee.org

[7] The Seed is Strong: Seeding Strategies in Search-Based Software Testing,Gordon Fraser,Andrea Arcuri, 2012 - ieeexplore.ieee.org

[8] Mutation-driven Generation of Unit Tests and Oracles,Gordon Fraser,Andreas Zeller,Copyright 2010 ACM 978-1-60558-823-0/10/07

[9] GENETIC ALGORITHMS FOR OPTIMIZATION, Programs for MATLAB ® Version 1.0 User Manual

[10] Software Testing Types, http://www.tutorialspoint.com/software_testing/testing_types.htm