Overview on Software Testing Methodology

Gaurav Kumar Srivastav, Dileram Bansal, Manoj Kumar Sharma

***Abstract*— The objectives of this paper we are compress the testing methodologies and techniques. In the real world, various type of testing methodology executed at this time. In this paper we are analysis of the two testing methodology first is the White box testing and another Black box testing….**

- ✓ **White Box Testing is such as a software testing method in which the internal structure/ design/ implementation of the item being tested is known to the tester.**
- ✓ **Black Box Testing is a software testing method in which the internal structure/ design/ implementation of the item being tested is NOT known to the tester.**

Both are used the following techniques which is improve the testing methodology in software.

1. **Comparison Testing**
2. **Equivalence partitioning**
3. **Loop testing**

***Index Terms*— methodologies, techniques, Black box testing.**

I. Introduction, Definition of Testing

Software testing is work as the third phase in SDLC(software development life cycle), that means it is only checking if a software program for specified inputs gives correctly and expected results according to requirements of user. There are various an important component of software testing methodology [fig 5]…

i. 1st component of software testing is SQA (software quality assurance).

ii. 2nd is the many software organizations are spending up to 40% of their resources on software testing.

The life cycle of software testing can be highly expensive. Because of that, these are studies about risk analysis .This term is define the software project will experience undesirable events, such as schedule delays, cost overruns, or outright cancellation, and more about this in [10].

In the real world, the various type definitions about software testing, but one can shortly define that as: A process of executing a software program with the goal of finding errors see [3]). So, testing means that one inspects results of a software program on a finite set of test cases (a set of inputs, execution pre-conditions, and expected outcomes developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement, see [11]) for which valued inputs always exist.

The whole set of test cases are considered as infinite, therefore theoretically there are too many test cases occur for the simple programs. In this case, testing could require months and months to execute. So, how to select the most relevant set of test cases? In test case used to various techniques, and some of them are correlated with risk analysis, while others with test programmer. Testing is an activity performed for evaluating software quality and for improving the performance of software. Hence, the main objective of testing is systematical process of different classes of errors [see [12]) in a minimum amount of time and with a minimum amount of effort. We decided the test result (see [2]):

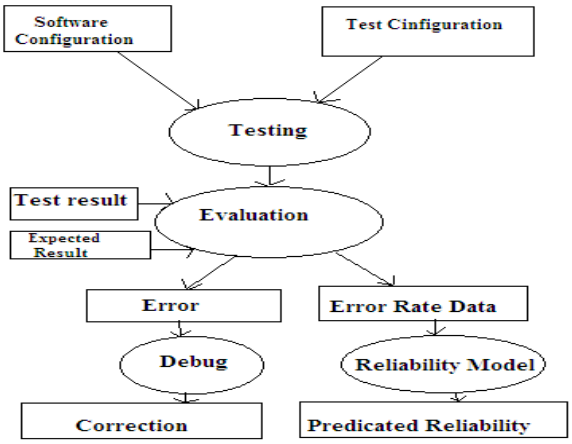


Fig (5) Software Testing Methodology

II. Methodology for Software Testing:

Software Testing are developed the various test techniques to achieve more effective test result. Software testing is used to techniques and conditions for testing which get the maximum number of errors in software program. So, testers do not guess which test cases to choose, and test techniques enable them to design testing conditions in a systematic way. Also, if one combines all sorts of existing test techniques, after that we will obtain better results rather if one uses just one test technique. Software can be tested in two ways, in another words, one can distinguish two different methods:

1. White box testing, and
2. Black box testing.

White box testing is highly effective in detecting and resolving problems about testing [12], because bugs can often be found before they cause trouble. Testing software, define this method as *testing software with the knowledge of the internal structure and coding inside the program* ([13]). White box testing is also called white box analysis, clear box testing or clear box analysis. It is a strategy for software ***debugging*** [14]) in which the tester has excellent knowledge of how the program components interact among them. This method can be used in WWW (Such as Web services applications), and is rarely practical for debugging in large network systems [14] about security. It is also known as security testing [15]. Security testing is provide the data and maintains functionality as intended (see [6]) method that can

Gaurav Kumar Srivastav (Asst. Professor), ACNCEMS, Aligarh
Dileram Bansal M.Tech(SE), SSSIST Sehore,Bhopal
Manoj Kumar Sharma, (H.O.D of CS Deptt.),ACNCEMS, Aligarh

be used to validate whether code implementation follows intended design, to validate implemented security functionality, and to uncover exploitable vulnerabilities (see [15]).

Black box testing is testing software based on output requirements and without any knowledge of the internal structure or coding in the program (see [16]). In another words, a black box is any device whose workings are not understood by or accessible to its user. In data mining, a black box is an algorithm that doesn't provide an explanation of how it works.

The main characteristics and comparison between white box testing and black box testing are follows.

A. *Black Box Testing Versus White Box Testing*

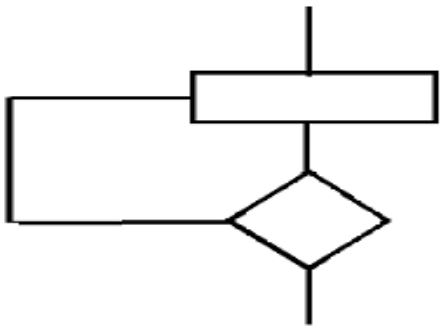
1) *Black Box Testing:*

- Performing the tests which exercise all Functional requirements of a program;
- Finding the following errors:
  1. Incorrect or missing functions;
  2. Interface errors;
  3. Errors in data structures or external database access;
  4. Performance errors;
  5. Initialization and termination errors.
- Advantages of this method:
  1. 1.The number of test cases are reduced to achieve reasonable testing;
  2. 2.The test cases can show presence or absence of classes of errors.

2) *White Box Testing:*

- Considering the internal logical arrangement of software;
- The test cases exercise certain sets of conditions and loops;
- Advantages of this method:
  1. All independent paths in a module will be exercised at least once;
  2. All logical decisions will be exercised;
  3. All loops at their boundaries will be executed;

Internal data structures will be exercised to maintain their validity



Fig(2)Simple Loop

III. SOFTWARE TESTING TECHNIQUES

A. *Equivalence partitioning:*

Equivalence partitioning is an important software testing technique. It is divided the input data of a software program into partitions of equivalence classes from which test cases can be derived, and test cases are designed to cover each partition at least once.

This technique divides the input data of a program onto equivalence classes. An equivalence class is a set of valid or invalid states for input data, and can be defined in the following way:

1. An input data specifies a range → one valid and two invalid equivalence classes are defined;
2. An input data needs a specific value → one valid and two invalid equivalence classes are defined;
3. An input data specifies a member of a set → one valid and one invalid equivalence class are defined;
4. An input data is Boolean → one valid and one invalid equivalence class are defined. Well, using this technique, one can get test cases which identify the classes of errors.

B. *Comparison Testing*

Comparison testing is an important part of testing, where testers compare a software product's strengths and weaknesses with other software products that are currently available in the market. Comparison testing is a very good indicator of how competitive and useful the software product will be to the end users soon after its commercial release.

C. *Basis Path Testing:*

Basis path testing is also white box testing technique that is used to test the code based on control flow. The method uses a control flowchart and a control flow graph to convert the code into a model and then derive independent test paths from it. For obtaining the basis set and for presentation control flow in the program, one uses flow graphs (Figure 3 and Figure 4). Main components of that graphs are:

- Node – it represents one or more procedural statements. Node which contains a condition is called predicate node.
- Edges between nodes – It is representing the 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. Cyclomatic complexity is computed using the control flow graph of the program The value evaluated for Cyclomatic complexity defines the number of independent paths in the basis set of a program.

For the given graph G, Cyclomatic complexity M is equal to:

$$M = E - N + 2,$$

Where, E = the number of edges of the graph, N = the number of nodes of the graph, P = the number of connected components.

The control flow generated from the program would look like Fig (1). In the above figure shows seven nodes (shapes) and eight edges (lines).

The formal formula the Cyclomatic complexity is 8-7 + 2 = 3.

D. *Loop Testing:*

Loop testing a white box testing technique performed to validate the loops. There are three kinds of loops as mentioned below:

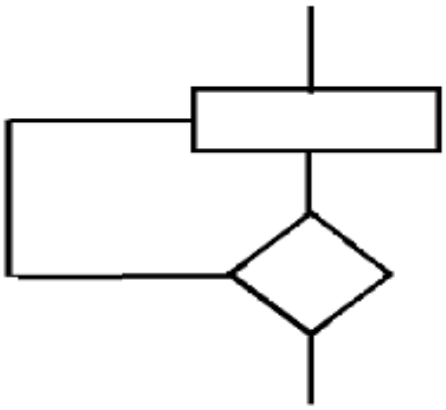
1. Simple loops;
2. Nested loops;
3. Unstructured loops.

4.1.1 Simple Loop Testing

It is possible to execute the following tests:

- Skip the loop entirely;
- Only one pass through the loop;
- Two passes through the loop;
- m passes through the loop where m<n;
- n-1, n, n+1 passes through the loop,

Where n is the maximum number of allowable passes through the loop. A typical simple loop is depicted in Figure 2.



Fig(2)Simple Loop

4.1.2 Nested Loops

If one uses this type of loops, it can be possible that the number of probable tests increases as the level of nesting grows. So, one can have an impractical number of tests. To correct this, it is recommended to use the following approach:

- Start at the innermost loop, and set all other loops to minimum values;
- Conduct simple loop tests for the innermost loop and holding the outer loop at their minimum iteration parameter value;
- Work outward, performing tests for the next loop;
- Continue until all loops have been tested.

A typical nested loop is depicted in Fig (3).

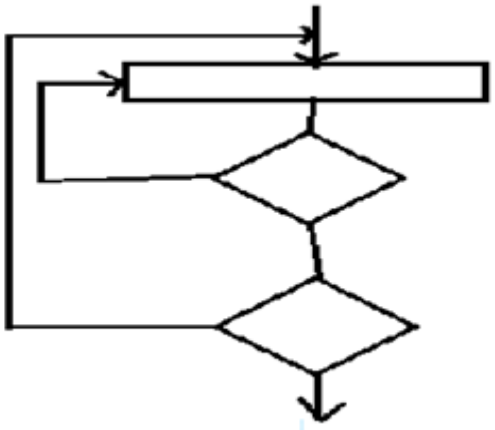


Fig (3) Nested Loop

4.1.3 Unstructured Loops

This type of loop should be redesigned. A typical unstructured loop is depicted in Fig (4).

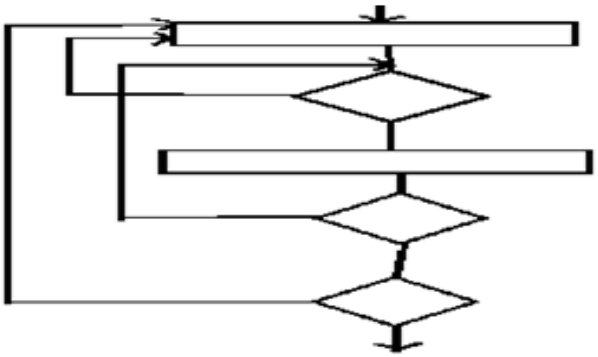


Fig (4) Unstructure Loop

REFERENCES

[1]http://www.his.sunderland.ac.uk/~cs0mel/comm83wk5. doc, February 08, 2009.
[2] Stacey, D. A., "Software Testing Techniques"
[3] Guide to the Software Engineering Body of Knowledge, Swebok – A project of the IEEE Computer Society Professional Practices Committee, 2004.
[4]"Software Engineering: A Practitioner's Approach, 6/e; Chapter 14: Software Testing Techniques", R.S. Pressman & Associates, Inc., 2005.
[5]Myers, Glenford J., IBM Systems Research Institute, Lecturer in Computer Science, Polytechnic Institute of New York, "The Art of Software Testing", Copyright 1979. by John Wiley & Sons, Inc.
[6]Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/wiki/
[7]http://www2.umassd.edu/CISW3/coursepages/pages/CIS311/outline.html
[8]Parezanovic, Nedeljko, "Racunarstvo i informatika", Zavod za udzbenike i nastavna sredstva – Beograd, 1996.
[9] Wei–Tek, Tsai, "Risk – based testing", Arizona State University, Tempe, AZ 85287
[10] Redmill, Felix, "Theory and Practice of Risk-based Testing", Software Testing, Verification and Reliability, Vol. 15, No. 1, March 2005.
[11] IEEE, "IEEE Standard Glossary of Software Engineering Terminology" (IEEE Std 610.12-1990), LosAlamitos, CA: IEEE Computer Society Press, 1990.
[12]http://www.testingstandards.co.uk/living_glossary.htm# Testing, February 08, 2009.
[13]http://www.pcmag.com/encyclopedia_term/0,2542,t=white+box+testing&i=54432,00.asp, February 08, 2009.
[14]http://searchsoftwarequality.techtarget.com/sDefinition/0,,sid92_gci1242903,00.html, February 08, 2009.
[15]Janardhanudu, Girish, "White Box Testing",https://buildsecurityin.usert.gov-/daisy/bsi/articles/bestpractices/white-box/259-BSI.html, February 08, 2009.
[16]http://www.pcmag.com/encyclopedia_term/0,2542,t=black+box+testing&i=38733,00.asp, February 08, 2009.
[17]http://www.pcmag.com/encyclopedia_term/0,2542,t=gray+box+testing&i=57517,00.asp, February 08, 2009.
[18]http://searchsoftwarequality.techtarget.com/sDefinition/0,,sid92_gci816126,00.html, February 08, 2009.
[19] Vilkomir, A, Kapoor, K & Bowen, JP, "Tolerance of Control-flow testing Criteria", Proceedings 27th Annual International Computer Software and applications Conference, 3-6 November 2003, 182-187, or http://ro.uow.edu.au/infopapers/88
[20]http://www2.hawaii.edu/~roever/wbt.htm, February 08, 2009.
[21]http://www.businessdictionary.com/definition/real-timetesting. html, February, 2009.
[22] Mikucionis, Marius, Larsen, Kim, Nielsen, Brian, "Online On-the-Fly Testing of Real-time systems", http://ww-w.brics.dk/RS/03/49/BRICS-RS-03-49.pdf, February, 2009.
[23]Tretmans, Jan, "An Overview of OSI Conformance Testing", http://www.cs.aau.dk/~kgl/TOV03/iso9646.pdf
[24]http://www.ifp.uiuc.edu/~hning2/protocol.htm, February 2009.
[25]http://it.toolbox.com/blogs/enterprise-solutions/betterobject-oriented-testing-21288, February 2009.
[26]http://pages.cs.wisc.edu/~bart/fuzz/fuzz.html, February, 2009.
[27]http://www.goldpractices.com/practices/mbt/index.php, February, 2009.
[28]http://blogs.msdn.com/nihitk/pages/144664.aspx, February, 2009.