Conceptual Study of Various Code Coverage Analyzer Tools

Khushbu[1], Dr. Kamna Solanki[2],Amita Dhankhar[3]

[1]M.Tech Student, M.D.University, Rohtak

[2,3]Assistant Professor, M.D.University, Rohtak

Abstract: Software testing ensures good quality and reliability of software programs/ applications. Software testing is selection and evaluation of different test cases. In software testing process Code Coverage analysis helps by finding areas which are not exercised by set of test cases of a program or to find defects in a program which are not exercised. It ensures that without missing out all key functional areas testing effectively carried out and includes all features. The coverage information is very useful for other many related activities, like unit testing, regression testing, mutation testing etc. Code Coverage Analyses tools are used for Languages Java, C, C++, Python etc. working on testing code of programs helps to find problem/defects in a particular software. Here in this paper my aim is to provide information of code coverage, different code coverage tools, how to choose the correct tool, its usage and limitations of code coverage tool.

Keywords: Code coverage, Code coverage tools, choose right tool, potential usage, limitations.

I. INTRODUCTION

In Software Development Life Cycle(SDLC), Software testing is an essential activity. It is process of detecting the defects and minimize risk associated with software. It is used to improve software quality, determine software quality and its activity is to obtain the code coverage.

Code Coverage analysis is about Finding areas of a program which are not exercised by a set of test cases or find defects, to increase coverage it create additional test cases. It also determine code coverage's quantitative measurement. Code coverage is a promising measure of test effectiveness, there are a large number of coverage analyzers or coverage tools that perform coverage analysis. To improve the testing process coverage measurement can be used in several ways .The coverage can help to find holes i.e. areas which are not covered in test cases, to increase coverage it create additional test cases which determine a quantitative measure of code coverage, which helps to Identify those redundant test cases who not increase coverage but waste time and effort.

Process of Code Coverage:-

1. Find appropriate Code Coverage tool
2. Start Test Execution – Phase 1
3. Measure Coverage and Identify areas where coverage is low
4. Add Test cases or conditions to improve code coverage
5. Do Test Execution – Phase 2
6. Measure that Coverage again

   Keep repeating steps 2 to 6, till we reach our target

Table 1: Code Coverage tools and Languages

Tools	**Language**		**Tools**	**Language**
JCover	JAVA		Quilt	JAVA
Emma	JAVA		Serenity BDD	JAVA
Gretel	JAVA		FrogLogic COCO	C, C++, C#
CodeCover	JAVA		Testwell CTC++	C, C++, C#, JAVA
NCover	.NET		Intel C++ Compiler15.0	C++
Cobertura	JAVA		Parasoft JTest	C,C++,JAVA,.NET
Bulleye Coverage	C, C++		SpiraTeam	Any Language
Clover	JAVA		Vector Software	C,C++
CodeCover	JAVA, COBOL		Devel::Cover	PERL
Coverage.py	Python		DotCover	.NET
Hansel	JAVA		OpenCover	.NET2
JACOCO	JAVA		Visual Studio	C,C++
NoUnit	JAVA		PITest	JAVA

II. Various Code Coverage Analyzer Tools

- **JCover**
  JCover is an code coverage analyzer tool. When an application test run it provides a mechanism which generate statistical information on coverage. It is for Java Programming.

- **Emma**
  Emma is an open-source tool, which detects dead code and verifies an application's parts which are actually defected. It is for Java Programming.

- **Gretel**
  Gretel is an open-source tool. It is test coverage monitoring tool, implements residual test coverage measurement. It is for Java Programming.

- **Code Cover**
  Code Cover is an extensible open source coverage monitoring tool, especially for condition coverage it performs source instrumentation to measure the coverage. It is for Java Programming fully integrated into Eclipse.

- **NCover**
  NCover is an open-source code coverage tool, tells during run of the application how many times each line of code was executed. It is for .NET Programming.

- **Cobertura**
  Cobertura is a free open-source tool. It calculate the percentage of code coverage which is accessed by tests. It is for Java Programming.

- **Bullseye Coverage**
  Bullseye Coverage analyzer tool, tells about how much of the source code was tested. It pinpoints the areas which need attention to be reviewed by programmer. It is for C and C++ Programming.

- **Clover**
  Clover is a cost coverage analysis tool of low cost. It supports statements, method, classes and package coverage. It is available for Eclipse or using ANT script or IDEA plug-ins. It is for Java Programming.

- **Code Cover**
  Code Cover is an open-source tool, which is a glass box testing tool measures statements, branch, loop and strict collection coverage of an application. It is for Java and COBOL Programming.

- **Coverage.py**
  Coverage.py is an open-source tool release on May 2017. It analyses the source to identify code that could have been executed but was not yet. It notes that which of the part has been executed. It is for Python Programming.

- **Hansel**
  Hansel is an open-source tool, whose source code comes from Gretal, only difference is Hansel is compatible with JUnit but Gretel is not. It let developers know how much code was supposed to test is covered. It is for Java Programming.

- **JACOCO**
  JACOCO stands for Java Code Coverage, is an open-source Code Coverage tool, which is an actively maintained tool. It can integrated with Ant, Maven, Gradle, Jenkins, Visual Studio etc. It is for Java Programming.

- **NoUnit**
  NoUnit is an open-source Code Coverage tool, measures your JUnit tests in projects using Java, XML and XSLT. It gives you a picture of code shows how good your JUnit Test are. It is for Java Programming.

- **PITest**
  PITest is an open-source Code Coverage tool release in 2016, which does mutation testing for Java, which point outs that which line of code isn't tested. It modify code and running unit test on the modified code.if test fails after change, test is useful else indicates it hasn't able to detect the changes. It is for Java Programming.

- **Quilt**
  Quilt is an open-source code coverage analysis tool, it use with the JUnit testing package. It can be used with Ant, Maven and others. It has three different versions. It is used for Java Programming.

- **Serenity BDD**
  Serenity BDD is an open-source coverage tool, is an Automated acceptance tool which monitor testing coverage for each of stories and epics written by user. It integrate with the usually buildtools and QA tools(Browser stack, Appium, Jenkins and Jira). It also provide an integrated testing suite based on Selenium. It is used for Java and Groovy Languages.

- **Frog Logic CoCo**
  Frog Logic CoCo is an code coverage tool, integrated with all major builds, CI, test tools and has a visual studio add-in. It is used for C, C++, C# and Tcl code.

- **Testwell CTC++**
  Testwell CTC++ is an code coverage analysis tool release in 2017 that provides coverage for line, statement, function, decision, multi condition, modified condition/ decision condition(MC/DC) and condition coverage. It is used for over 25 years and still actively developing, which used across industry (Aerospace, Transportation and Healthcare etc.). It is easy to customize and used in embedded applications. It is used for C, C++, Java and C# Programming.

- **Intel C++ Compiler 15.0**
  Intel C++ Compiler 15.0 is an Code Coverage tool that displays dynamic execution counts for each basic block of the application provides differential coverage data and presents code coverage information visually with a customized coloring scheme. It is used for C++ Pragramming.

- **Parasoft JTest**
  Parasoft JTest is an Code Coverage tool release on 2017, does code coverage by line, statement, block, condition, decision, path and MC/DC. It has an option for your concerned about your compliance and security for your application. It can breakdown code coverage by file, class, function, method. It can do unit testing, integration testing, plug-in testing, server side testing, automated end to end functional testing, scripting testing and user acceptance testing. It is an old tool integrated with all key tools (like build-in tools, CI tools etc.). This code coverage tool is not developer friendly, it is better suited for traditional corporations. It is used for C, C++, Java, .NET Programming.

- **Spira Team**
  Spira Team is an Application Lifecycle Management tool release in 2016, focuses on improving the quality of testing and coding. It can tackle code coverage (How will your code is tested) by its requirements. It is used for any Programming Language.

- **Vector Software**
  Vector Software is an Code Coverage analysis tool used in Software Development Life Cycle(SDLC) in testing. It is used for C, C++ Programming

- **Devel::Cover**
  Devel::Cover is an open-source Code Coverage analysis tool which test code coverage by statement, condition, subroutine, etc. It is still developing but not actively. It is the only tool for PERL Language.

- **Dot Cover**
  Dot Cover is a tool release in 2016 integrated into visual studio and analysis unit test coverage. It can highlight code coverage right inside the code editor, which is very convenient. It is used for .NET Language.

- **Open Cover**
  Open Cover is an open-source tool works only on windows and for visual studio you have a couple of options for code coverage. It is used in .NET2 and above Languages.

- **Visual Studio**
  Visual Studio is a Code Coverage tool release on 2017 has integrated tools to collect code coverage metrics. It is used for C, C++ Languages.

HOW TO CHOOSE RIGHT TOOLS :

- ❖ First consider Programming Language
- ❖ Consider open-source or commercial tools which is best suited for you
- ❖ Make sure tool is actively developed
- ❖ Choose tool which response customer support
- ❖ Tool must have broad range of development
- ❖ Check tools approach(focus on requirement, mutation testing, unit testing etc.) is according to your development or requirement.

III. POTENTIAL USAGE OF CODE COVERAGE :

In industrial context code Coverage analysis of large applications is a case of using coverage tools.

1. Test coverage increase Reliability. On the basis of testing a logarithmic–exponential model estimate defects in a program, evaluating reliability growth and projecting can help developers to optimally allocate resources to meet a deadline, which include target of reliability.
2. It provides information of coverage related testing progress's quantification. It detect redundant test cases by which execution of redundant test cases can be removed, which saves time and effort.
3. Code coverage analysis becomes a motivation for testers to improve tests/test cases in testing. It provides a quantitative measure to improve testing by reports of testing progress. Its goal of correcting errors may reduce the cost.
4. A significant benefit of coverage analysis is, it discover dead pieces of code. Those codes which are never executed are called dead piece of code in application. These dead pieces of code increase complexity of software maintenance. It should be removed.
5. All data obtained by coverage in different test cases/ test runs can be combined or merged. In it we can reset or drop/dump code coverage data remotely without JVM exit.
6. It is not required to access of the source code and provide decreasing debug information which is available in the input classes.

IV. LIMITATIONS OF CODE COVERAGE:

Code coverage analysis is very important while testing but still these tools are not very popular in this industrial world. Use of these tools facing several challenges in industry.

1. Less importance is given to these (testing and quality assurance) because of market pressure, they want to short the development lifecycle of application/ software.
2. Code coverage looks like an additional expensive because it not ensure immediate response in investment which we ask from developers/ programmers/ testers.
3. There is no direct theory that shows/predicts with use of coverage how much quality improved.
4. We cannot check/ calculate that by increasing coverage how many more defects can be found. There is a fact that full code coverage test (100%) not means the absence of defects/problem. When picking a measure, in a measure between usability and thoroughness there is always a balance.
5. There is a need for developers to learn how to use these tools for testing and to interpret coverage requires investment.
6. Data analyzing in code coverage is a complex activity and is often misused.

7. Code coverage analysis is a guide, it is not a goal/target. It helps you to do right testing/create right test cases which leads to get syntactic execution of your code.

V. CONCLUSION :-

In this paper, we studied about different code coverage analysis tools which used for finding defects in a program or software. These tools can be used in broad ways, and different tools used for different suitable languages selected by user which make it broad. It is found that purpose of selecting particular tool according to a particular software is fulfilled. I hope my goal of providing information about different tool is fulfilled and may help you in future.

REFERENCES:

[1] Asaf, S., E. Marcus, et al., ”Defining coverage views to improve functional coverage analysis**”,** In the Proceedings of 41st Design Automation Conference, 2004.

[2] Williams, B. S. a. L., "A Survey on Code Coverage as a Stopping Criterion for Unit Testing.", Technical report (North Carolina State University. Dept. of Computer Science), TR-2008-22, 2008.

[3] E Kajo-Mece , Megi Tartari **, “**An Evaluation of Java Code Coverage Testing Tools**”,**pp 72-75 , 2012.

[4] Faizah Omar, S. I,. "A Software Traceability Approach to Support Test Coverage Analysis."In the proceedings of Third International Conference on Advances in Human-oriented and Personalized Mechanisms, Technologies, and Services CENTRIC 2010.

[5] Li, J. J., ”Prioritize code for testing to improve code coverage of complex software**”,** In the 16th IEEE International Symposium on Software Reliability Engineering, 2005. ISSRE 2005, 10-pp.

[6] Jones, J.A. and Harrold, M.J. (2003) Test-suite reduction and prioritization for modified condition/decision coverage. IEEE Trans. Softw. Eng., 29, 195–209.

[7] M. Kessis, Y. Ledru, G. Vandome, “Experiences in Coverage Testing of a Java Middleware”, in Proceedings SEM 2005, Lisbon, Portugal.ACM, pp. 3945, 2005.

[8] Lormans M, D. "Reconstructing Requirements Coverage Views from Design and Test using Traceability Recovery via LSI**”,** TEFSE, Long Beach, California, USA, ACM 2005, 2005

[9] Malaiya, Y.K., Li, M.N., Bieman, J.M. and Karcich, R. (Software reliability growth with test coverage. IEEE Trans. Reliab., 51, 420–426,2002.

[10] Shahid Muhammad, Ibrahim Suhaimi**,** “An Evaluation of Test Coverage Tools in Software Testing”**,** International Conference on Telecommunication Technology and Applications, Proc .of CSIT vol.5, Singapore, pp 216-222, 2011.

[11] Priya L.S., Ram Raj N., Askarunisa A., “Measuring The Effectiveness Of Open Coverage Based Testing Tools”**,** Journal of Theoretical and Applied Information Technology, Vol. 5, No.5, pages 499-514, 31 May, 2009.

[12] Qian Yang, J. Jenny Li, David M. Weiss, “A Survey of Coverage-Based Testing Tools”, Published in *The Computer Journal (2009*) , volume 52 (5): pp. 589-597, 2009.

[13] Rauf, A., S. Anwar, “Automated GUI Test Coverage Analysis Using GA”**,** Proceedings of the 2010 Seventh International Conference on Information Technology: New Generations, pp. 1057-1062, 2010.

[14] Whalen, M. W., M. P. E. Heimdahl, et al. “Coverage metrics for requirements-based testing.” 2006, In Proceedings of the 2006 International Symposium on Software Testing and Analysis, ISSTA 2006, pp. 25-35, 2006.

[15] Xia, C. and Michael, R.L. The Effect of Code Coverage on Fault Detection under Different Testing Profiles. Proc. First International Workshop on Advances in Model-Based Testing, ACM Press, St. Louis, MO , 2005.

[16] http://istqbexamcertification.com/what-is-decision-coverage-its-advantages-and-disadvantages/

[17] Joy M. Agustin, ”JBlanket: Support for Extreme Coverage in Java Unit Testing”,2005.

[18] Anju Bansal,Kamna solanki,”A Review on Code Coverage Analysis Tools”, In CiiT International Journal of Software Engineering and Technology, 2013.