JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

A Study on Comparison of Code Coverage Analyzer for JAVA

Subhash Singh Parihar
(Associate Professor)
Department of Computer Science & Enginnering, PSIT
Kanpur,India
ssparihar.psit@gmail.com

Pradumn Kumar
(Assistant Professor)
Department of Computer Science & Enginnering, PSIT
Kanpur,India
Pradumnyadav18@gmail.com

***Abstract*— Software testing is a fundamental action in the software development process, as of late have more sound and predictable testing tools become accessible for testing Java programs and their components. These tools support for the most part useful and control-flow based structural models. Code coverage is a promising proportion of test adequacy. There are countless coverage analyzers or coverage tools that perform coverage investigation. The result of coverage estimation can be utilized in more than one way to further develop the testing system. The coverage can assist with finding openings for example regions that are not covered by test cases. Code coverage likewise helps in Regression testing, test case prioritization, test suite expansion and test suite minimization and so on. This paper intends to give a comparison of code coverage investigation tool. It is as yet really difficult for test supervisors and engineers to recognize the suitable code coverage tooling solution for the framework or component under test. This comparison recognized here may assist with choosing the proficient and successful tool.**

Keywords—JCover, EclEmma, Gretel, Covertura, Code Cover, Clover

I. INTRODUCTION

Software testing is presently a fundamental movement in programming support life cycle. It is a movement used to decide and further develop programming quality. In software testing, software metrics empower the proper quantitative data, to help us in the decision-making on the most proficient and suitable testing tools for our projects. Where advancement is more precise, associations look for proportions of testing fulfillment and goodness to set up test finish measures. Code Coverage is one such measure. As there are different code coverage tools reasonable for various applications, we are focusing in here on six most normally utilized tools.

Errors can sneak in the product in any period of software advancement like requirement analysis, design and coding. Testing is the stage wherein every one of the errors are to be recognized. An appropriate meaning of testing is given by G.J. Myer[1]:"Testing is the process of executing a program with the intent of finding errors ."

Officially in testing stage we will have a program P and details S. Testing ought to guarantee that whether P(x) = S(x),Vx E J, where I is the input space and it should give the set of test cases x where P(x) # S(x). As information space size is by and large.

Extremely huge testing turns into a costly stage which roughly requires half of the all out effort[2]. Programming testing and quality expenses for fabricated things can be pretty much as low as 2% in shopper items or as high as 80% in items, for example, space ships, atomic reactors and airplane, where disappointment undermines life describes[3].

Accomplishment of testing relies upon the test cases that are chosen. We ought to lean toward those experiments which cause disappointments. A few heuristics and strategies have been proposed to pick the issue uncovering test cases. Every one of the heuristics can be comprehensively isolated into two categories:

- Black box testing
- White box testing

Black box testing is also called functional testing. In black box testing the tester sees the program as a black box. The tester doesn't know about the design of the program. The analyzer plans the experiments from the particulars. Black box testing is utilized for framework testing [11].

White box testing is also called structural testing or coverage based testing. In white box testing the tester picks the test cases from the design of the program. White box testing is oftentimes utilized for unit testing. In white box testing, code is noticeable to testers so it is likewise called Clear box testing, open box testing, transparent box testing, Code-based testing and Glass box testing.

Create these components, incorporating the applicable criteria that follow.

II. CODE COVERAGE

Code coverage performs code coverage analysis. Evaluating software coverage and taking proper actions lead to an improvement of software quality. It helps in evaluating the effectiveness of testing by providing data on different coverage items. Testing is done to find bugs; coverage is also a measure of your effort to detect a certain class of potential errors. For example, 100% line coverage doesn't just mean that you've executed every line of code; it also means that you've tested for every bug that can be revealed by simple execution of a line of code. The fundamental motivation behind code coverage code inclusion instrument in this study is to utilize the subtleties of code coverage for SUT in relapse experiment streamlining. Code coverage reviews the parts of code being exercise by the experiments and those requirements to be improved. Code coverage recognizes that piece of code which has not been practiced by experiments and henceforth there is a need to expand the test suite[18].

Each of these runtime analysis tools can be utilized alone or along with the part testing highlights. At the point when the source code is run with any of the runtime analysis tools drew in, either alone or in a part test, the source code is instrumented. The subsequent instrumented code is then executed and the outcome is powerfully shown in the figure1.

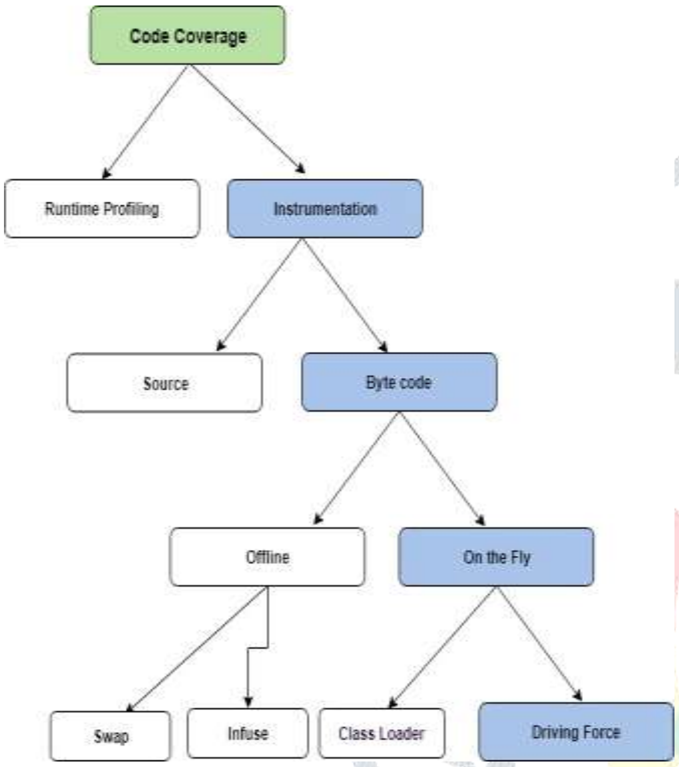

Figure1: Code Coverage

A. Why use a Code Coverage Analyzer?

Elite of redundant change and improvement, software isn't able to keep up with addressing its clients' requirements. However, select of wide-going computerized test-suites, such change can likewise portrayal clients to exceptionally ugly risks: the risk that critical elements never again work as in the past, the risk of information misfortune and the risk of a framework breakdown. Agile software development systems are working on the nature of software. Test-driven turn of events, where tests are composed to test highlights before those elements are added, guarantees that each piece of software is combined with a test-suite. Code coverage is a metric that can assist you with seeing the amount of your source is tested. An extremely valuable measurement can assist you with surveying the nature of your test suite, and we will see here the way that you can begin with your ventures. No subject how top notch such procedures are, and how completely they're continued in an association, it isn't feasible to guarantee that product testing is really comprehensive. That is the place where tools can help.

Test suites are intended to approve the activity of a framework against necessities. One significant part of a test suite configuration is to guarantee that framework activity rationale is tried totally. This is a troublesome undertaking. Code coverage tools support test suite originators by giving the data concerning what portions of source code are covered during framework shown in figure 2.

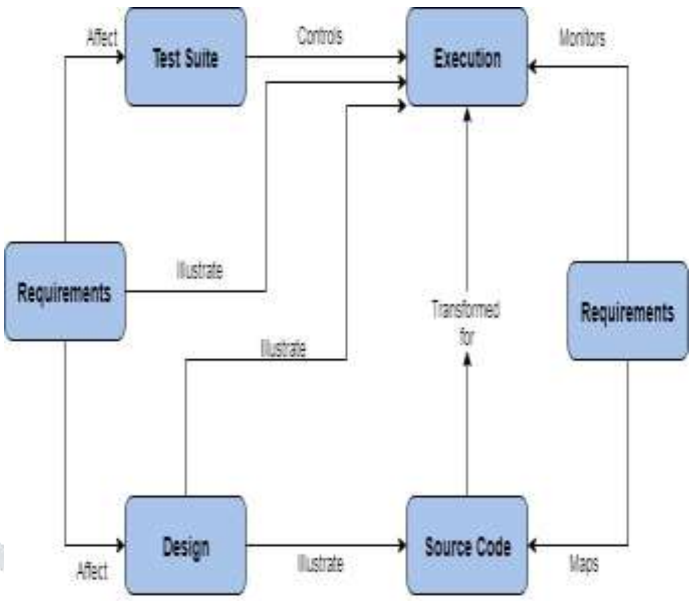

Figure2: Code Coverage Analyzer

B. Code Coverage analyzer (CCA) and Test Coverage analyzer (TCA):

Test Coverage Analyzer is a tool which helps in doing coverage based testing. A test coverage analyzer can show the portions of a program that have not been covered during testing. This information is essential for performing coverage based testing. Some of the uses of the coverage information for a tester are:

- By looking at the uncovered portions of a program for a given test suite, the tester can prepare additional test cases to achieve more coverage. So, it helps in accelerating the testing activity.
- In regression testing we can prefer those test cases which essentially execute the customized statements. So, it helps in regression test case selection [11].
- The tester can identify the undeveloped code in the program easily.
- We can associate color with the covered statements depending on the percent age of passed test cases so that fault localization can be done easily [6].
- Number of defects present in the program can be estimated [7].
- The quantitative measure of coverage information can be used to know the progress of testing.
- Quantitative measure is also helpful in identifying redundant test cases. Redundant test cases are those test cases that will not increase the coverage.
- Quantitative measure can also be used to know the test suite is adequate or not.

III. Basic Coverage Measures

There are many coverage measures, over a 101 are listed in [14]. Brief description some of the coverage measures are given below.

Table1: Coverage Measures

S.no	Coverage Measures	Indicates
1	Statement Coverage	Statement coverage technique is **used to design white box test cases**. This technique involves execution of all statements of the source code at least once. It is used to calculate the total number of executed statements in the source code out of total statements present in the source code[11].
2	Branch Coverage	Test every line, and every branch on multi-branch lines[11].This criteria checks for all the branches that are formed with if statement, if else statement,for loop statement, while statement, do while statement, switch statement and exception handlers [24].
3	Condition Coverage	Condition Coverage or expression coverage is a testing method used to test and evaluate the variables or sub-expressions in the conditional statement. In this coverage, expression with operands are just thought of.
4	Path Coverage	Test every path through the program, from entry to exit. The number of paths is impossibly large to test[14]
5	Multi-condition or predicate coverage.	Force every logical operand to take every possible value. Two different conditions within the same test may result in the same branch, and so branch coverage would only require the testing of one of them [14].
6	Loop coverage	Loopcoverage "Detect bugs that display themselves just when a loop is executed at least a time or two."
7	Data coverage	At least one test case for each data item / variable / field in the program
8	Relational coverage	Relational coverage "Checks whether the subsystem has been practiced in a manner that will in general distinguish off-by-one errors. Every boundary on every output variable. • Every boundary on every variable used in intermediate calculations [14].
9	Fuzzy decision coverage	If the program makes heuristically-based or similarity-based decisions & uses comparison rules or data sets that evolve over time, check every rule several times over the course of training[14].
10	N-length sub-path coverage.	Test every sub-path through the program of length N. For example, in a 10,000 line program, test every possible 10-line sequence of execution[14].
11	Multi-condition or predicate coverage	Force every logical operand to take every possible value. Two different conditions within the same test may result in the same branch, and so branch coverage would only require the testing of one of them[14].
12	Decision Coverage	**Decision Coverage** is a white box testing technique which reports the true or false outcomes of each boolean expression of the source code. The goal of decision coverage testing is to cover and validate all the accessible source code by checking and ensuring that each branch of every possible decision point is executed at least once.

IV. SURVEY OF CODE COVERAGE TOOLS

L. Shanmuga [12] selected four software code coverage tools such as JCover, Emma, Gretel and Code Cover .They concludes that out of these four tools Code Cover tool is comparatively better in calculating the coverage metrics and helps in the selection of best test cases based on coverage for regression testing of Java programs.

Muhammad and Suhaimi [13] give an assessment of different test inclusion instruments in programming testing. The assessment models comprise of language support, instrumentation, Coverage Measurement, GUI and Reporting. The board and test administrators require fitting tools for the software under test.

Following is a brief description of some code coverage tools:

A. JCover

JCover is a code inclusion analyzer for Java programs. It gives a component to produce factual data on the coverage of an application during a trial. It tends to be utilized to work out the level of code that was executed, rate not executed, what sources were not utilized in records, etc. JCover supports statement and branch coverage. [8].

B. EclEmma

It was developed by Vlad Roubtsov. EclEmma is called code coverage analysis. It's a free java code coverage tool for eclipse. EMMA can instrument classes for coverage either offline (before they are loaded) or on the fly (using an incrementing application class loader). It Supported coverage types: class, method, line, basic block. It Coverage stats are aggregated at method, class, package, and "all classes" levels and Output report types: plain text, HTML, XML. The HTML report supports source code linking and Output reports can highlight items with coverage levels below user-provided thresholds. Coverage data obtained in different instrumentation or test runs can be merged together [16].

C. Gretel

Gretel is a free code coverage tool for Java program via Carls Howells at the University of Oregon. The essential contrast among Gretel and other coverage checking tools is that Gretel carries out leftover test coverage observing: After you run a program that has been instrumented with Gretel. Since most projects invest the greater part of their energy in a couple of little locales, which are effectively canvassed in the initial not many trials, remaining re-instrumentation with Gretel incredibly diminishes the exhibition punishment of test inclusion monitoring [11].

D. Cobertura Cobertura

It is a free Java tool that computes the level of code got to by tests. It tends to be utilized to distinguish what portions of your Java program are deficient with regards to test coverage. It depends on jcoverage. Cobertura is an open source code inclusion instrument for Java. This is a Jcoverage based instrument. To utilize this instrument one ought to pronounce Maven module in POM.XML file [9].

E. Code Cover

Code Cover estimates proclamation, branch, circle, term inclusion (subsumes MC/DC), question mark administrator inclusion and synchronized inclusion The Code Coverage analysis is the testing activity (May be part of Unit Testing, Component Testing, Regression Testing), where in uncovered (untested) code is highlighted. The metrics give the quantitative measure of tested code and untested code[17]. Code Cover device is an extensible open source glass box testing instrument that can be utilized as a code inclusion for Java programming. It was created in 2007 at the University of Stuttgart. It very well may be executed in the order line, Eclipse, and Ant.19 [10].

F. Clover

Clover is an incredible device for producing code coverage reports from your unit tests. It very well may be executed as a module in shroud, expert or subterranean insect. Nonetheless, not every person realizes that it can likewise be utilized to gather inclusion information of reconciliation tests [15].

This section describes a comparative evaluation of six code coverage analysis tools based on few important criteria such as type, coverage measures supported, memory space, graphical representation, html support, reports and type of instrumentation.

TABLE2: COMPARISON OF CODE COVERAGE TOOLS IN JAVA

Feature	**JCover**	**EclEmma**	**Gretel**	**Cobertura**	**Code Cover**	**Clover**
Type	Prorietary	Open Source	Open Source	Open Source	Open Source	Prorietary
Memory	8 MB	460 MB	502 KB	NA	3.63 MB	NA
Coverage measures support	Statement,Branch, Method, File, class coverage	Method, Class, Package and all classes levels	Statement Coverage	Statement and Branch Coverage	Statement, Branch, loop and condition coverage	Statement, Method, class and package coverage
Graphical Represent-ation	Yes	No	No	Yes	Yes	Yes
HTML Supports	Yes	Yes	No	Yes	Yes	Yes
Reports	Plain Text, HTML format, Graph	Plain Text, HTML format, XML	Line, Table, Hit Table	XML, HTML	Coverage, correlation, Boolean analyzer, views and HTML format	XML, HTML
Type of Instrument-ation	Source code Instrumentation	Byte code Instrumentation	Source code Instrumentation	Byte code Instrumentation	Source code Instrumentation	Source code Instrumentation

V. CONCLUSION

This paper assesses six code coverage tools. Comparison has been made in light of certain models, for example, type, coverage measures upheld, memory space, graphical portrayal, html backing, reports and type of tools and Table 2 sums up the outcome. Each tool has some solid point just as some flimsy spot. Clients and designers can choose the tool as indicated by their need. This comparison will help in more proficient choice of as well. The effectiveness of the test case can be determined by the level of code coverage. Hence EclEmma is a very popular open source code for java software to measure code coverage than other tools. EclEmma helps to identify untested parts of a code base and improve the corresponding tests. Some possibilities for future improvements in the code coverage tools are : De-instrumentation, regression test case selection and customized reports.

REFERENCES

[1] G. J. Myers, "The art of Software Testing" Wiley Interscience, New York, 1979.

[2] P. Jalote. An Integrated Approach to Software Engineering. Narosa Publishing House, 1991.

[3] Boris Beizer. Software Testing Techniques. Van Nostrand Reinhold, New York, 1990.

[4] Christina Pavlopoulu, Michal Young. Residual Test Coverag e Monitoring In Pro- ceedings of the 21st International Conference on Software Engineering (ICSE'99), pages 277- 284 , May 1999.

[5] Atul S. Paldhikar. Coverage Based Testing and Test Data Generation. Master's Thesis, Depart ment of Computer Science, IIT Kanpur, Janu ary 1993.

[6] Venkata Sreenivasa Rao N. Test Coverage Analyzer for Java Ph.D Thesis Department of Computer Science & Engineering,IIT,Kanpur,March 2003.

[7] Jules Kouatchou, "Code Coverage Tools" Available on https://modelingguru.nasa.gov/docs/DOC-1828, March 5, 2010

[8] JCover, http://www.mmsindia.com/JCover.htm

[9] http://cobertura.github.io/cobertura/

[10] http://codecover.org/

[11] http://www.testingtoolsguide.net/tools/gretel/

[12] E Kajo-Mece , Megi Tartari , "An Evaluation of Java Code Coverage Testing Tools",pp 72-75 , 2012

[13] 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.

[14] Cem Kaner, "Software Negligence and Testing Coverage".

[15] https://confluence.atlassian.com/clover/about-code-coverage-71599496.html

[16] http://emma.sourceforge.net/faq.html

[17] Abhinandan H. Patil1,∗ and Nandini S. Sidnal2, "CodeCover: A Code Coverage Tool For Java Projects", Proceedings of International Conference on " Emerging Research in Computing, Information, communication and applications"ISBN-9789351071020.

[18] Priyanka Dhareula* and Anita Ganpati, "Open Source Code Coverage Tools for Java: A Comparative Analysis", Indian Journal of Science and Technology, Vol 9(32), DOI: 10.17485/ijst/2016/v9i32/100202, August 2016