

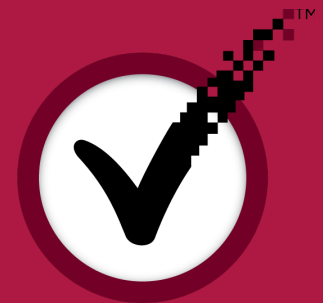


NoVaTAIG

Code Coverage - NOVATAIG

Kiran Ramineni <kramineni@verisign.com>

Principal Engineer VeriSign



Agenda

- + Terminology
- + What is Code Coverage All About ?
- + Why use a code Coverage Tool ?
- + Should your goal be 100% coverage? Why? Why not?
- + Considerations in choosing a code coverage Tool
- + The basics of using code coverage tools
- + Different Tools and Rationale
- + Q & A

Terminology

- + Lines of Code
 - Number of lines of uncommented code
- + Branches
 - Possible number of paths ex: switch/case, if/else
- + Continuous Integration
 - Integration of code as often as possible - several times a day
- + Automation
 - Reducing the need for human intervention for repetitive tasks so that they can be run more often.
- + Cyclomatic complexity
 - is a software metric used to measure the complexity of a program

What is Code Coverage All About ?

- + Tests drive code quality But who/what drives test quality ?
- + Code coverage measures how much of your source-code is being exercised by your test suite
- + Code coverage measures if all the possible branches have been exercised by your test suite
- + Code coverage lets you find hotspots
- + Code coverage should be part of continuous integration and automated.
- + Code coverage provides information on Cyclomatic complexity based on number of branches.

Why use a code Coverage Tool ?

- + “No matter how good methodologies are, and how diligently they're followed in an organization, it isn't possible to ensure that software testing is as comprehensive as it could be. That's where tools can help.”
- + James Gosling - "I don't think anybody tests enough of anything" (A Conversation with James Gosling).

Should your goal be 100%? Why? Why not?

- + Well this can be a very religious debate.
- + The norm is that 80 - 90% of coverage is good .
- + Anything below 60% is usually unacceptable
- + Reaching for 100% - if you have the time and resources why not but it would take a considerable effort to be there
 - Simulation of events and faults might be needed to achieve this
 - Example:
 - Database down
 - Out of Memory
 - Out of File Descriptors

Considerations in choosing a code coverage Tool

+ Essential aspects

- **Instrumentation**
 - Pre Compilation Instrumentation - Source code is being modified
 - Post Compilation Instrumentation - Binaries are modified
- **Execution**
 - Run/Compile in Standard Environment
 - Need special execution Environment
- **Integration**
 - Support automation and IDE
- **Reporting**
 - Simple understandable reports
 - Ease of navigating to the source code
 - Flexibility in merging results and trending.
- **Free ?**

+ Tools

- Cobertura
- EMMA
- Clover
- JCoverage

The basics of using code coverage tools

+ Demo

Different Tools and Rationale

+ Cobertura

- Byte Code instrumentation
- Standard Execution environment
- Ant/Maven Integration
- XML/HTML Reports including Code Complexity
- Supported by Several Continuous Integration Tools
 - Atlassian Bamboo

+ Emma

- Byte Code instrumentation
- Standard Execution environment
- Ant/Maven Integration
- XML/HTML Reports doesn't support Branch or path coverage
- No support for Eclipse

Different Tools and Rationale Cont ..

+ Clover

- Source Code instrumentation
- Standard Execution environment
- Ant/Maven Integration
- XML/HTML Reports
- Commercial Product from Atlassian

+ jCoverage

- Commercial - Cobertura was forked from the jCoverage Code base.

Cobertura Report - 1

Cobertura Report

D:\work\dev\workspaces\codecoverage\workspace\CoberturaPitfalls\reports\cobertura-html\index.html

Coverage Report - All Packages

Package /	# Classes	Line Coverage	Branch Coverage	Complexity
All Packages	3	73% 11/15 2/6	33% 1/2 1/4	1.6
com.verisign.caf.demo.example1	1	100% 5/5	50% 1/2 1/2	2
com.verisign.caf.demo.example2	2	60% 6/10 4/10	25% 1/4 3/4	1.5

Report generated by [Cobertura](#) 1.9 on 9/23/08 1:30 PM.

Packages

[All](#)
[com.verisign.caf.demo.example1](#)
[com.verisign.caf.demo.example2](#)

All Packages

Classes

[BranchCoverageExample](#) (75%)
[LineCoverageExample](#) (100%)
[SomeObject](#) (0%)

Cobertura Report - 2

Coverage Report - com.verisign.caf.demo.example1

Package /	# Classes	Line Coverage	Branch Coverage	Complexity
com.verisign.caf.demo.example1	1	100% 5/5	50% 1/2	2

Classes in this Package /	Line Coverage	Branch Coverage	Complexity
LineCoverageExample	100% 5/5	50% 1/2	2

Report generated by [Cobertura](#) 1.9 on 9/23/08 1:30 PM.

Conclusion

- + Code coverage report is not a silver bullet for code quality
- + Lines of code
 - can be misleading
 - Coverage – all it means is more lines of code exercised
- + Pit falls to watch out for
 - Integration Testing is still needed
 - Scenario Testing is still needed
 - Watch out for Test code generation tools that could mislead
- + Pick tools that best suit your needs
- + Scenario tests coupled with code coverage could workout.
- + Insist on code quality reports with all code drops and releases.
- + Bug detection tools come handy - example: FindBugs

Articles of Interest

- + Measure test coverage with Cobertura
 - <http://www.ibm.com/developerworks/java/library/j-cobertura/>
- + In pursuit of code quality
 - <http://www.ibm.com/developerworks/java/library/j-cq01316/index.html?ca=drs>

Q&A

+ Q & A