Project Requirement Document

An assessment of high level requirements for developing a grading rubric.

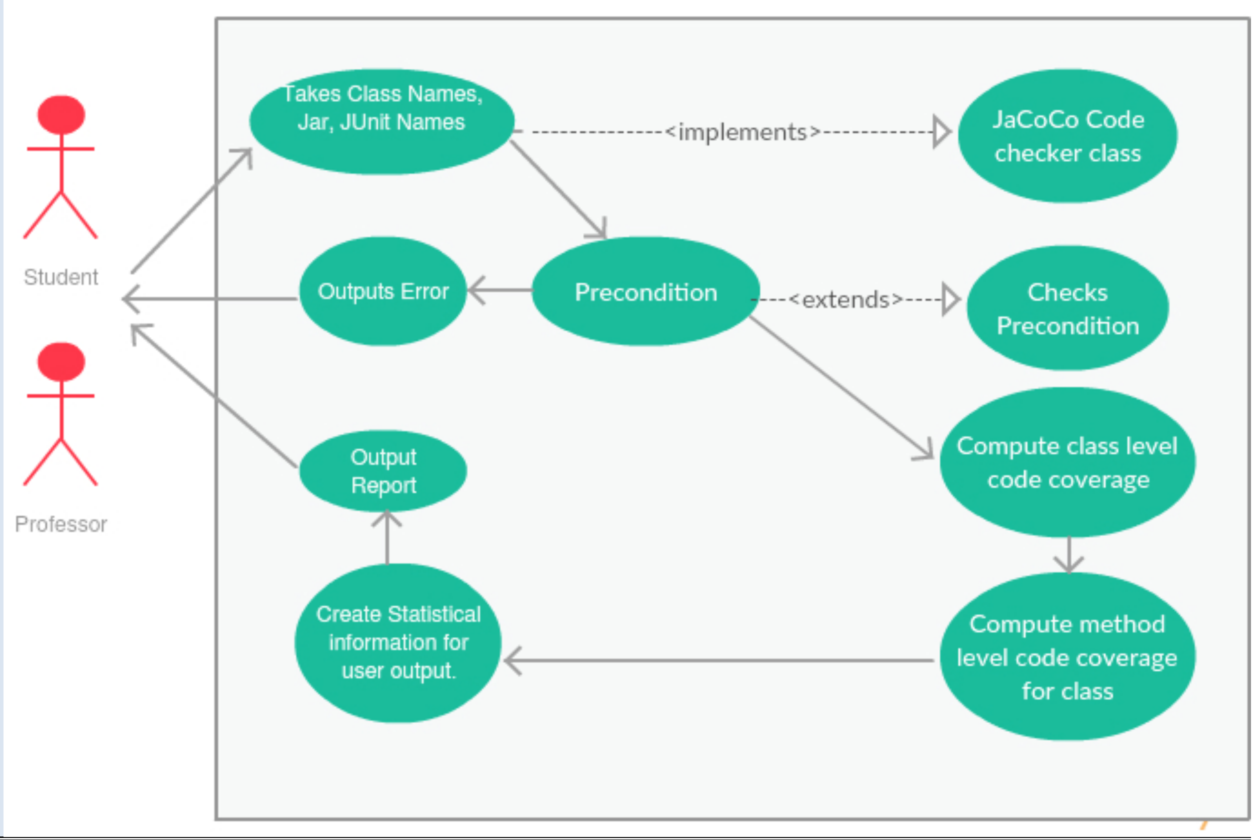
Sep 23, 2016

Contributors:

Manjeet Singh,
Sandeep Shenoy,
Vidhi Shah,
Neerad Somanchi,
Vivek Singh.

CSE 542 Group #21

Use Case Name: Grading Rubric

Goal	A target class or method is put through a Junit test. The software under discussion provides the percentage of the target class/method covered by the JUnit test.
Precondition	The target code (class/method) is written and ready. The corresponding JUnit test is also created. The software under discussion is open and awaiting input, viz., the target code and JUnit test.
Basic Course	1) The user indicates to the software that he wants to find the code coverage (by percentage) of a JUnit test. 2) The software responds by asking the user to provide the following details: • The path of the file which has the class/method whose code is to be covered by the JUnit test. • The required JUnit test is to be selected by the user. • The software checks for the percentage of the class/method covered by the selected JUnit test.
Alternate Course	In step 2), the user can make a few mistakes: A) Providing an invalid path B) Not providing a Java class/method In such a scenario, the software outputs an error and also describes it. Further, it urges the user to go through the step 2) procedure again.
Graphical Representation	 Student Professor Takes Class Names, Jar, JUnit Names <implements> JaCoCo Code checker class Outputs Error Precondition <extends> Checks Precondition Output Report Compute class level code coverage Create Statistical information for user output. Compute method level code coverage for class
Post condition	The percentage of the target class/method that was covered by the JUnit test.

User Stories:

1) As a test lead I want to check the completeness of the test cases prepared by the testers so I submit the test cases and obtain the percentage of the target code covered by the same.

2) As an instructor I want to grade the test cases prepared by my students so I run the software on the submitted test cases to obtain their code coverage.

3) As a student I want to self-evaluate the test cases I prepare and so I rely on the software to provide me with the code coverage guaranteed by my test cases.

4) As a project manager I want to rate the performance of the testers so I submit their test cases in the software and evaluate them based on the code coverage returned.

Discarded user roles:

Discarded role 1: A programmer who wants to test the code coverage based on his test cases
Reasonable: The programmer may need to test whether all edge cases are hit.
Invaluable: The tool reflects the efficacy of the JUnit test cases and not that of the target code.

Discarded role 2: Instructor from C++ class wants to use the same tool.
Reasonable: The tool takes classes, methods and JUnit test cases which are available with other instructors as well.
Invaluable: The tool deals only with Java classes and methods.

Discarded role 3: A manager using it to get employee ratings based on their coding performance as a function of the code coverage.
Reasonable: The code gives test coverage, a grading metric.
invaluable: The tool provides the data for coverage statistics but has no criteria to rank or grade individual people based on any parameters.

Discarded role 4: A developer using the code to get an estimate of the program asymptotic runtime for a particular class.
Reasonable: The library has features to give details about Cyclomatic complexity to indicate level of nested loops.
Invaluable: Functionality can't be used to gauge the runtime of the class. Not supported by the tool.

Discarded role 5: A programmer who uses the tool to check if his code is as per the coding standard guidelines.
Reasonable: The tool checks for a few of his coding standards such as loops and their nesting and if the JUnits are written well.
Invaluable: Since it doesn't give any other information such as naming standards, format and structure it can't be fully used for testing the coding standards.