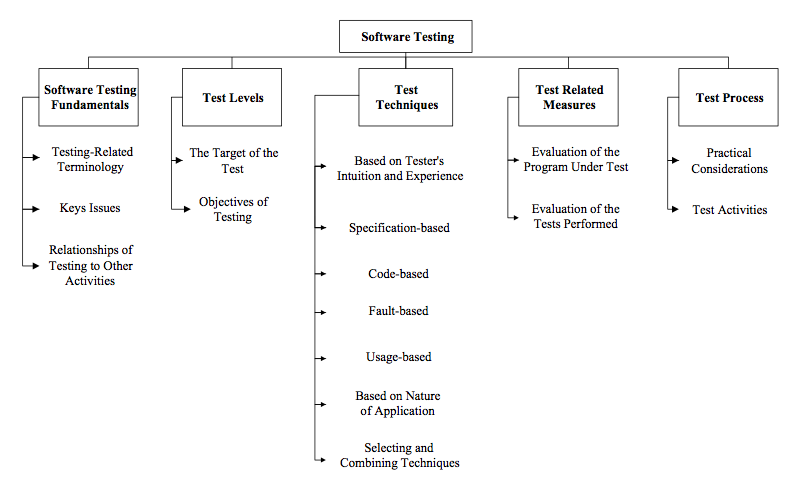
Yeisol Woo

CS460 Senior Capstone Project 1

Homework #4

Review Software Testing (SWEBOK KA-4)



Testing is an activity performed for evaluating product quality performed for evaluating product quality, and for improving it, by identifying defects and problems. The software testing knowledge area of SWEBOK deals with examining software systems for quality. More specifically, testing consists of the dynamic verification of the behavior of a program on a finite set of test cases, suitably selected from the usually initiate executions domain, against the expected behavior. The knowledge area is further broken down into the following sub-topics: software-testing fundamentals, test levels, test techniques, test-related measures, and test process.

As with any software engineering project, our team had to do extensive software testing to ensure a high-quality end product. Since we followed a pseudo rapid application development process, we tested each module and sub-module directly after implementation. For example, we tested the database / persistence layer after it was implemented. To do this, we ran a series of unit tests as well as manual tests to ensure that data transactions were ACID in nature and that data was being accurately stored and retrieved. After the round of tests, we analyzed the results to look for problem areas. We then revisited each problem area, analyzing the code in question and applying a fix to attempt to repair the bug. After the fixes were applied, we re-ran the tests, analyzed the new results, and applied fixes as needed. We continued to do this until the bugs were worked out of the system. We also ran separate tests on each module after completion. These separate tests were performed to test the interactions of the sub-modules and how well they met the software requirements. Similar tests were also done of the system as a whole upon completion to test the interactions of its modules. Each of the tests’ results was analyzed to ensure that the behavior of the tested section matched the expected behavior. For example, if the data entry user interface was supposed to present the user with a certain number and type of controls, then we ensured that each of the controls were present and that each of their behaviors met the required behavior.