Miniproject in Test and Verification - Part 1

Jacob Karsten Wortmann
Sam Sepstrup Olesen
Nicklas Andersen
sw805f14

April 1, 2014

We will in our project make an application that functions as a digital cookbook. The user is able to search for recipes by different search functions: by ingredients and by the name of the recipe. Users can share the recipes with friends and relatives, it is also possible to favourite a recipe. Our application also supports conversion between the imperial and metric system to reach a larger audience. We want to create a thin client which does not perform a huge amount of calculation, this means that we can focus on testing the server and the communication layer in the client.

1a - Test Design

Technical specification

Server-side We will use PHP5.3.2 to do the server-side scripting, we will perform unit test with the tool PHPUnit and mutation test with judgedim/Mutagenesis.

Client-side Our target Android SDK is 19 and our minimum supported Android SDK is 18. We will use the Android Testing Framework to perform unit tests.

Test environment

The idea is to create a test environment for the server when doing tests, this way we ensure that we do not accidentally manipulate and corrupt production data. We want to set up a test database that is identical to the database used in production, we could choose to directly copy the database before each test. This way we can insert new data such as recipes and test on these alongside already existing recipes.

Code coverage

We want 100 percent statement code coverage on the server-side because of this being the most crucial part of the system. If we find a failure on the server it can only be because of an error that happened on the server, but a failure on the client could both originate from the server and the client. This is because the client is dependent on the server whereas the server is not dependent on the client. As for code coverage on the client we will make a risk analysis to identify the most crucial parts of the client. Based on the risk analysis we can identify which parts of the client that has the highest priority and then schedule our time plan of testing based on this.

Mutation testing

The tool that we have chosen to use for mutation testing, judgedim/Mutagenesis, supports a lot of operator mutations and also some logical mutations such as inverting an if statement. The tool supports first order mutants, which is sufficient to find a weakness in our unit tests, it is also the method used in practice.