Reflections report

1. Difference between individual programming and working in a team

This is the first software project for every member in the group where the number of developers are more than two. The biggest difference is the increase of communication paths. In a group of four people, there are 6 possible paths of two-way communication (N*(N-1))/2 = 12/2 = 6) and the risk of misunderstanding in different matters is high. When you are working by yourself, you never need to tell someone else what you have done or what you are doing. This is crucial and critical when working in teams. In order to facilitate the communication work, we have used Google docs (in order for everyone to see the same documents and write documents together) and WhatsApp (communication chat tool for direct and fast communication to everyone in the team). Teamwork also demands better and more commenting in the code, and for this we have used JavaDoc. GitHub has been a major help in the version control and code writing. This project would not be as successful as it is without the help from a shared web-based repository.

All four members in the project have enjoyed the extra challenge of working in a team. You learn a lot when working together with others. The learnings are not only code and programming related, you also learn how other people communicate, solves problems, plans and order working tasks etc. The best thing with working in teams compared to working by yourself is the possibility to get help from others. When you get stuck in a task, you can always ask one of the others for help, and often you can solve the problem together. Especially when you work together in the same room. However, this can also be a disadvantage when you get disturbed of other team members when working. The benefits exceeds the disadvantages, however.

Another big difference with working in teams is the fact that you don't need to do everything by yourself. In a efficient and high performing team, all members are motivated to work and have tasks do to. This means that even if you have a busy day with other things, the project could still be going forward because other members are working.

A couple of things we are really proud of in this group, is

the fact that we successfully used SCRUM-methodology for this project. Also, we are proud of the fact that we have succeed in working together during the whole project. We stated from the beginning that we wanted to work together, rather than working by ourselves at home. This has enabled better group dynamics, communication, and more effective synergy-effects in the group work.

2. Methods used to create the documents

We have used Google docs in order for everyone to see the same documents and write documents together. Google docs enables commenting on the documents and therefore easy proofreading.
A disadvantage with Google docs is the low functionality of version control. Although it is possible to retrieve old versions of a document, it is not very easy or useful. GitHub has a lot of better version control.

However, the need for version control with regard to documentation writing in this project has been relatively small, but still very useful. It would be of even higher importance in a larger project.

We have also used WhatsApp (a communication chat tool for direct and fast communication to everyone in the team) for instant and effective communication. This has been useful when creating the documents since it is easy to ask the others to proofread a document or a piece of text.
GitHub has been a major help in the version control and code writing. This project would not be as successful as it is without the help from this web-based shared repository. GitHub has also been useful for uploading weekly hand-in documentation to the course administrators using tags.

3. Appropriateness of the adopted SE techniques

The waterfall method has been used to some extent in this project. The method states that a software project consists of five steps (see figure 1) where one should move to the next step only when its preceding step is completed and perfected.

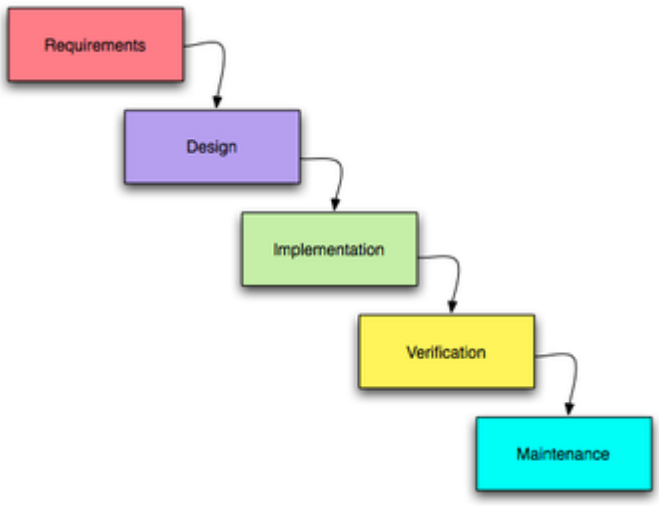

Figure 1 - waterfall method
http://en.wikipedia.org/wiki/File:Waterfall_model_(1).svg

The waterfall process has been useful for this project. It was a good start to write requirements early in the process, which forces you to define the functionality of the application at an early stage. To start with writing requirements helped us to identify some special cases and functionality which we wouldn't have thought of without writing the requirements.

However, the steps in the model above have not been performed in the way the waterfall process states. The steps/phases have been conducted more iterative and not in the strict order presented by the methodology. Some requirements have been changed during the process and some have been added in the implementation process when a need for additional functionality have arisen.

A methodology more appropriate and more used in this project is Scrum. All members of this development team have enjoyed using and practicing Scrum. It took a couple of weeks in order to get used to the work with sprints, but after that, the methodology has been effective and useful. Everyone knows what to do and in which order. Scrum has facilitated the prioritization work and enabled a task-driven working process.

Another SE-technique used in this project, to a low extent, is test driven development (TDD). TDD was not supposed to be used in this course and for this project. However, one of tests in this project derived new functionality and therefore, TDD has

been tried and used in some extent. This has been interesting and useful according to the developers in this project. One of our internal goal was to practice and learn about different SE-techniques, and using SDD helped us achieving this goal.

In summary, the use of SE-techniques in this project has been interesting and educational. In some extent, the documentation work in the project have been relatively time consuming. Some of the work with the documentation and different techniques, has been over-ambitious for a relatively small software project like this one (for example code coverage). It has however been very educational and interesting to use many of the methods used in large software projects.

4. Coverage

The technique used for code coverage was time consuming and quite difficult to get started with. A lot of time was used trying to get the code coverage software EMMA function with Eclipse in a Android-project. Although, we succeeded with the EMMA-tool using Jenkins. And when the technique was implemented, we found the code coverage tool helpful. It helped develop UNIT-tests and get an overview of how many UNIT-tests you need.

If we would have conducted a larger project with higher complexity, the code coverage work (and formulating good tests) would have been of even higher importance.

5. General comments about the project as a whole

The members of this team have enjoyed the practice in software development widely. It have been fun to both code a Android application and practice different software engineering techniques. Using waterfall method, Scrum, GitHub, code coverage - EMMA, nightly builds - Jenkins etcetera has been very educative and useful. Although, a lot of time has been used in order to get the techniques and different tools to work. It would have been good if more time could have been placed on using the tools, instead of installing the tools.

One of the hardest parts within software development is the time planning and how long time a specific implementation takes. This

has been present in this project which gave us useful experience for the future.

Working with this Android-project have also been interesting due to the future potential and hype regarding application development for smartphones. This project have made us familiar with all major tools for developing applications, which has been very attractive and appealing. This course will definitely be useful for us in the future!