Postmortem Analysis

Group 9:
Feng Liu
Jiajun Ni
Claire Huri
Ang Zheng

The development of SCRS system is a great opportunity for us to practice and utilize the software engineering knowledge to solve real problem. During developing process, not only was programming skills strengthened but also we acquire a deep understanding about how the software process works. Software engineering is definitely not a just programming process. The whole design process is built up in a progressive way by five individual assignments, which requires us to have a rigorous thinking to avoid the conflicts in the following steps. Accomplishing each assignment can be treated as a milestone in our design process. The communication and coordination with different groups of people cannot be avoided. For instance, the user interface and database were designed by our teaching assistants and some design process have to be built on top of those components. We got through the whole process even though we encountered lots of confusion and problems in each stage. In the following paragraphs, we will address the problems in each stage and what we could have done better.

In milestone one, the purpose of constructing the requirements document is to be clear about what the software is used for and what functionality it features. First of all, we extracted all the user cases from the general requirement documents to help us to develop requirements. Requirements are very important since it reflects the need of customers. Moreover, the requirements are written in a natural language, which are easy to be understood by customers, or colleagues who do not have sufficient knowledge in this area. Furthermore, all requirements are written in details without any ambiguities. It is noted that the requirements are divided into two main categories, which are functional and non-functional. It is very straight forward for us to come up with the idea about the functional requirements but from the non-functional requirements' perspective, it is relatively hard for us due to the flexibility and some settings are based on the experience that are what we are lack of.

In milestone two, we were trying to revise requirement document we had created in milestone one and continue to develop a collection of requirement-based tests. The hardest part is to think about all situations including the exceptions during the requirement tests. For the positive requirements tests, it is very straightforward which is to achieve the major functionalities. However, exceptions have to be taken into considerations since the software are not only designed to handle valid input but also invalid or unexpected ones. It is noted that several requirements might have conflict situations, which require us to carefully state all potential conflicting problems and constraint the software to avoid potential risk. Looking back to this stage, we could have done better on the test parts. Even though we have tried very hard to come up with possibilities as many as we can, some interrelationship between different requirements

are still out of the scope, which reflects in the stage five. For positive side, all requirements are well organized and met with customers' purpose.

In milestone three, we went into the design process including design structure, UML diagram and dynamic model sequential diagram. Before we dive into it, all of our teammates contribute their own ideas about how the design structure would look like. After discussion and combining each one's design benefits, architecture is finalized which exhibits four levels. Good design architecture is supposed to have properties of high cohesion and low coupling. On the top level, the classes are used to communicate with predefined user interface and all methods are called in the second level, which are categorized by administrator function, student function and instructor function. On the third level, classes are created based on methods provided in DBcoordinator. The last level is the classes that have direct interaction with database. The initial purpose of having second level is to group all functions that belong to each role that will use the software. However, it turns out to be confusing and useless since those classes will call all methods stored in the third levels. The major problem in this stage is to understand and coordinate with the files provided from our teaching assistant. Even though we have some freedom to our design, it is confined by other factors such as pre-set user interface and DBcoordinaor. It showed us how important the communication between groups. Moreover, in order to make the architecture more efficient, some redundant classes should be deleted such as the second level in our initial design. Accurate understanding about the other module designed by third party is important since they are parts of the final design and are coupled to our design. It gives us better visualization about the dynamic model by creating sequential diagram, where the corresponding functions are invoked in sequential steps.

In milestone four, the most fun part- coding part was undertaken. Even though the design in milestone three is carefully revised, it is inevitable to modify it here and there to adapt for the real coding process. Several redundant methods were got rid of and the system was further refined. Since our implementation is based on the user interface and DBcoordinator, we have to make changes according to the modification from those two modules, which makes the design process not as fluent as we expected.

In final stage, milestone five, the test implementation was undertaken. It is important since the test process guarantees all functionalities work and the software has the ability to handle all exceptions. The coverage of the test code can be calculated by either software tool and hand calculation. Even though this program is not very big, it is still very time consuming to manually count all coverage percentage. As a result, the test code was made to test the whole system. The problem is that the database does

not have sufficient data for us to test all cases. We decided to firstly create all data in the database and then implemented all tests by codes.

Through this semester, we have not only gained sufficient knowledge about the software engineering and implemented the term project by going through from ground to final product. At the beginning of the semester, several software developing models were introduced such as waterfall model and spiral, etc. The waterfall is strict and efficient but far from realistic. It is just impossible to develop every component at one time without any changes later on. It is more common to use incremental model especially for the project where cooperation involves. Moreover, the coordination between different groups, such as TAs, is also very important. Since the design process exhibits the property of modularity, how to couple the imported modules to ours is crucial. Some problems were introduced at the first time such as requirement-based tests at early stages, especially for non-functional requirements. Some other problems were not foreseen until the design process, such as design structure and required functions in different classes. It took a lot time to clarify the user cases and requirements, which seems to be waste of time initially but it benefits a lot in actual design process. The project could not go this well without all teammates' collaboration and contribution.