Lesson Learned

Software Engineering Project Spring 2014

By Chao Li, Bing Li, Yang Zheng

Purpose

This document is for what we learned from this project and what experience we can share with others, Including both technical skills and communication skills.

Lesson Learned

1. Project Organization and Communication
2. Reducing complexity in the System.
3. Using framework but not start from scratch.
4. The interaction between backend and front.
5. Testing
6. Grouping a program team

Project Organization and Communication

Use case Diagram team structure:

1. Algorithm /Backend Code/Database Design By Chao Li
2. Algorithm/Test/UI Design By Bing Li
3. Algorithm /UI Code/UI Design By Yang Zheng

   a. We followed our schedule and tried to finish our weekly tasks. Every week we had a sub-meeting to figure out what we have done and what we are going to do next week.
   b. Everyone in our team is clear with their responsibilities.
   c. Made a conclusion on the problem and make sure every team member understand it.

Reducing Complexity in the System (Backend Experience)

Backend Coding Experience:

1. Using Abstraction and inheritance in the code.
   Here is an example in my code:
   UseCaseAssociation.java
   ExtendAssociation.java
   IncludeAssociation.java
   ExtendAssociation and IncludeAssociation are inherient from UseCaseAssociation

2. Developing a data model to describe the use case diagram data.
3. Using Functional in the code.
   Every task will be finished by single function.
   Different task will be placed in different functions.

The experience we could share with others:

1. How to make your code more readable.
   a. Giving your variable a meaningful name.
   b. Making comment in your code.
2. How to make your code more reusable.
   a. Keeping the code Dry. Dry means less repeat code.
   b. Making class/method do just one thing.
   c. Using framework but not start from scratch.

Using framework but not start from scratch (UI Experience).

The experience we could share with others:

Using popular existing frameworks instead of reinventing wheels, it is an efficient and effective way to build good software. There are two advantages:

a. There are many popular existing frameworks that provide common functionalities in all software systems. Using existing frameworks can save a lot of time than implementing the functionalities in by our own. Then we can focus more attention on our own business logic.
b. Popular existing frameworks are more reliable because they are usually implemented by powerful software company or highly-experienced software communities. These frameworks are widely used in industry so it is well-tested by others and will less likely have bugs than the one implemented by our own.

The interaction between back-end and front-end.

Using JSON to facilitate the communication and interaction between front-end developers and back-end developers:

In web development the communication between front-end developers and back-end developers is very important. But usually the front-end developers cannot understand back-end logic vice-versa. Traditionally, the front-end developer needs to know different servlet/actions in the back-end and the parameters related to these servlets/actions. This usually causes problems. While, in our approach, we use JSON to exchange data between back-end and front-end. JSON can be easily understood by both front-end and back-end programmers. We first define the format of the JSON that both the front-end and back-end will deal with. i.e. the objects and fields in JSON. Then the front-end and back-end is decoupled. They can do their work in parallel.

Testing

Using JUnit to test back-end before the UI is not implemented:

We used JUnit to test the backend, before we finish the integration of backend and front. We could define the test case and expected result. After we finish this test, we could check whether the test result is matched with our expected result. I thought the most advantage is we don't need to build a UI to test your backend. JUnit could have you to test it directly.

Grouping a program team

The new diagram team has good functions assigned, three people who is responsible for back-end, UI and testing respectively. But they have functionally related, back-end and U I are intercepted, also UI will use JavaScript to help back-end to validate data. Testing wi ll confirm every function runs well in the end. It means when a team group members, all objects in the same team should be functionally related.

Communication with other teams

We are not good at communicating with other teams and the project leader. And this is the biggest mistake for us in this project. Because we kept the Chinese culture in this project, we focused on only our assignment, but not communication with other people. So we suggest other foreign groups that please report your processes to the leader and talk to the leader in time, it can make she/he know what you have done in time. Then she/he can get a clearly plan and timeline of the whole project. Also we should communicate with other groups, such as noticing navigation team that we need a tab from them.

Following the basic procedure assigned from management team

It is very important to finish all management software that is assigned from the management team. Or you will get low score even you did a lot of work and spent a lot time on this project. The basic procedure that is decided by the managers is the most important thing.