Peer review for Jacob and Walid

from: Henry Pap
Rasmus Skedinger
Seldin Music

- Running the extensible jar file gives a lot of errors.
- Good cohesion, classes are not too large but not too small.
- Logic is good for the classes, no extra classes that helps (MemberList could be considered as a list, or implement list instead of a class that has a list).
- The consistency of the comments is strange. Some areas of the code are well commented while other are not. The commenting could need some structure to it. Different commenting structures are used.
- We found no dead code, except for some code that was commented out.
- Model-view separation is not the best as its both depended on model and boat.

Does the implementation and diagrams conform (do they show the same thing)?
The UML class diagram corresponds well with the code implementation. The interaction diagrams are only showing a small part of the whole project.

Is the Architecture ok?
Model view separation could need some adjustments to be relay pure. Some calls between model and view breaks this concept.
The model is not specified for a specific UI i.e. it does not print any kind of view UI.

Is the requirement of a unique member id correctly done?
Yes they handle the id correctly by using the database, this to prevent duplicate ids.

What is the quality of the implementation/source code?
This project has been implemented using the default java coding conventions. A Good naming convention has been implemented, with the first letter in lower case and with the letter of each letter capitalized. Readability could be better for example methods could be grouped in similar cases, for better way to search in the classes.

As a developer would the diagrams help you and why/why not?
The UML class diagram give a good and easy overview of what dependencies and associations there is between classes. While the interaction diagrams only show two scenarios “Create member” and “show compact list” which is fine, and can give an overview of methods interacting with classes.

What are the strong points of the design/implementation, what do you think is really good and why?
The design and implementation are simple while, while classes are not too large. Smaller classes makes it more readable and easier to maintain. We didn’t notice any unnecessary classes or dead code.

What are the weaknesses of the design/implementation, what do you think should be changed and why?
The lack of comments could make it hard to understand the code in detail. In the controller having a class called Admin can be seen as a deviation from the domain model.

Do you think the design/implementation has passed the grade 2 criteria?
We think this project has met all the requirements for grade 2. While the interaction diagrams could be fleshed out to cover the whole project.