Jovydas Urbanavicius, Jean-Pierre Salum, Mantas Remeika | 1DV607 | October 12, 2016

Peer review

Simon Palmqvist

Clear instructions for running the application was provided. It proves that there were no problem running application.

Member class is supposed to implement serializable interface, but it’s not implemented.

**Architecture:**

* **Is there a model view separation?**

We can see that view and model classes are separated.

* **Is the model coupled to the user interface?**

Models are not coupled to view.

* **Is the model specialized for a certain kind of UI (for example returning formated strings to be printed)?**

In our opinion the model used in this application is not reusable, since it was created specifically for this assignment.

**Is the requirement of a unique member id correctly done?**

Implementation of unique member id in this application considerably satisfies requirement in the assignment.

**What is the quality of the implementation/source code?**

* **Code Standards**

Classes and methods in this application do not have any documentation which could help other developers understand what is the purpose of each class and method. Documentation would also help for developer of this program to remember purpose of each unit in the application.

* **Naming**

Code is written according to all Oracle Naming Conventions [1].

* **Duplication**

We couldn’t find any code duplication.

* **Dead Code**

We couldn’t find dead code in the application code.

**What is the quality of the design? Is it Object Oriented?**

In this code objects are connected using associations and not keys/ids.

Router class in the controller package handles all system events, which according to Larman is a feature using GRASP pattern [2, p.431]. Classes have high cohesion and low coupling. We did not notice any hidden dependencies. Encapsulation is done properly using get/set methods and using private variables and methods avoiding global.

**As a developer would the diagrams help you and why/why not?**

The sequence diagram and class diagram would give us basic information before starting implementation

**What are the strong points of the design/implementation, what do you think is really good and why?**

The structure in this application is clearly made using Model View Controller pattern, which gives better organization and allows easier maintenance. Encapsulation is done properly using get/set methods and using private variables and methods avoiding global variables.

**What are the weaknesses of the design/implementation, what do you think should be changed and why?**

We think this implementation is satisfactory and we have no suggestion to make changes.

**Do you think the design/implementation has passed the grade 2 criteria?**

Our team thinks that this application passes requirements for the grade 2.

# References

* 1. <http://www.oracle.com/technetwork/java/codeconventions-135099.html>
  2. Larman C., Applying UML and Patterns 3rd Ed, 2005, ISBN: 0131489062