Peer review for Rikard Österlund ro222cm

from: Henry Pap
Rasmus Skedinger
Seldin Music

- Setting length of a boat does not have a safety as it crashes if not an integer.
- Can't update a boat with new type (even though it says it should).
- Setting type of a boat in xmlHandlers class, takes a switch case that checks chars and then based on the char value sets a boat with a number. In the Boat class it checks again with a switch to see what number, instead this could be achieved not check the type in xmlHandlers but only in boat.
- MemberInfo, MemberList and XmlHandlers servers no purpose of a class, MemberInfo could be in Member class, and XmlHandlers could be in MemberDB, MemberList is a list class, it should implement a list instead of having it because if not it could just be a list be a list in RegistryCtrl
- What purpose does UpdateMember have (as a class)? Its like a menu class but it only handles Member events, why not just have some sort of a menu class that handles all the menu things instead of splitting it up in both RegistryCrtl and UpdateMember
- In some menus there is no back or exit to abandon.
- There is 2 MemberDB in UML class diagram.
- Member number and member ID is used interchangeable without explanation.
- The names of the some classes is misleading.
- Good model view separation.

Does the implementation and diagrams conform (do they show the same thing)?
As much as we can see the diagrams and the code correspond in to each other.

Is the Architecture ok?
A good structure regarding model view separation.

Is the requirement of a unique member id correctly done?
It is correctly done.

What is the quality of the implementation/source code?
Code naming convention are in almost all the cases good except for some names are to short and can be hard to understand. We feel like some classes could be put into other classes to make it more readable and understandable, as noted above.

What is the quality of the design? Is it Object Oriented?
Your controller is implemented correctly from the definition from GRASP. Being a middle man between model and view.

As a developer would the diagrams help you and why/why not?
The UML class diagram has the all the core concepts. It explains some dependencies and associations regarding the Grade2. While the UML class diagram is pretty accurate it is in need of some clean up to make it more readable for people who are not familiar with this project.

What are the strong points of the design/implementation, what do you think is really good and why?
The strongest points of the code is its commenting, it has a god structure while maintaining readability.

What are the weaknesses of the design/implementation, what do you think should be changed and why?
The core structure of the project could need some more planing e.g MemberInfo could be placed in Member to make a better structure in the project.

Do you think the design/implementation has passed the grade 2 criteria?
Rikards Grade2 has met all the functionality for workshop2 Grade2 but it could need some clean up when it comes using his software. In some cases one can’t go back or exit some proceses instead one has to finish that task and then exit. Sometimes it asks for member ID but it’s never printed out for the user, but a Member No.