**Peer Review Kristoffer Karlsson**

**Architecture**

The structure follows the rules of the MVC pattern, and is structured in such a way that it is easy to differentiate what the code does and therefore I conclude that there is a model view separation. The model does not know of the functionality in the view, however is reliant of the controller. I would argue that there is some coupling; changing the user interface should not affect the model. The model views also do not seem to be specialized for a certain UI, other than the exception messages are in English, which may making an interface in an different language more time-consuming.

Furthermore, I have concluded while reading the code that the UI does have a read-only access to the model classes User and Boat, which is allowed, as they are iterated through and presented to the user, but not written to.

Each instance of a member is given a ID, by using the method random in the class System.Math, however the uniqueness of it can be questioned. An preferred option may be to use System.Guid[[1]](#footnote-1) or getHashCode[[2]](#footnote-2) according to what I have found when quickly researching the subject[[3]](#footnote-3).

**Code quality and implementation**

While reviewing the source code, I found the readability to be fine. I did however find a desperate lack of comments, as comments for each method makes it easy to get an overview of a class when all sections are collapsed. The naming is more than satisfactory however, which increases the readability significantly.

However, the naming of the method *DoControlOperation* is quite generic and does not imply what it does – which is handling events provided by the user. What is a control operation? To me it is quite unclear. This method also calls upon itself, instead of making use of a do while loop.

In regards to code quality, I would prefer strings not to be concatenated, but rather to use string format and binding variables to the string. Other than that, I did not find anything to remark upon, as there was no duplication of code that I found, and no dead code.

**Design Quality**

I found the design quality to be fine, as it is Object Oriented, data was encapsulated and objects such as User and Boat in this task, were not connected using ID’s but rather by association, as should be. It does however contain a singleton class which contains static methods, which was to be avoided.

The level of cohesion was high, as the classes responsibility was confined to what was expected by the class in question- no more, no less, “lagom”, as one would say in Swedish. The application has some degree of coupling; I do not however find it to be that problematic.

**Documentation**

At first glance, I found the readability of the class diagram to be low at first glance. However, it was a very positive attribute to the documentation to divide the classes into MVC in the diagram as well. With more intricate knowledge of the system, provided by the sequence diagram it is easy to follow along the operation and overlook the system class by class. However, not all operations are provided by the sequence diagrams, only the creation of and listing of members are provided, which only gives partial overlook of the system in the sequence diagram. This is balanced out by the class diagram and the source code, which has good readability despite the lack of comments, but I think I would like to have the full view as a developer if I were to complete or update the system.

**Conclusion- the strong and weak points**

The level of cohesion is good, and the source code is well organized and readable, due to good naming mainly. However, I found the documentation to be lacking somewhat – not severely, mind you, as it is still possible to get an overlook of the system quite easily. If this were an bigger application though, with many more classes and thousands of lines of code more, this would become an problem and I would suggest that it is seen to. I would also recommend spending a few minutes on proper comments, as it is helpful for the readability – which luckily is not a problem. I therefore conclude that the grade 2 criteria are met, though the sequence diagrams perhaps should be extended.

1. MSDN Guid Structure (2015-10-08) https://msdn.microsoft.com/en-us/library/system.guid(v=vs.110).aspx [↑](#footnote-ref-1)
2. MSDN GetHashCode (2015-10-08) https://msdn.microsoft.com/en-us/library/system.object.gethashcode%28v=vs.110%29.aspx [↑](#footnote-ref-2)
3. # StackOverFlow [Generate a unique id](http://stackoverflow.com/questions/11313205/generate-a-unique-id) (2015-10-08)

   http://stackoverflow.com/questions/11313205/generate-a-unique-id [↑](#footnote-ref-3)