# Peer-Review

## Workshop 1, Group: Sverker Söderlund (ss223ck)

On first sight, your model is quit clean, and easy to understand, especially if the observer has knowledge about the problem description and the requirements. You have not implemented every concept in the problem description, but instead used the requirements as delimiter.

However, “secretary” and his/her responsibilities are not listed in the requirements, but you choose to include this in the model. This is probably not wrong, but to me it looks like a flow chart, or a use case. You have three objects that are not UML conceptual classes (person, secretary and member). My understanding is that you consider these as “roles”, instead of “concepts”. You start with an entry point (the arrow from person to user) and from there, it is a path to either “secretary” or “member”.   
The lines between objects in the model should not represent data flow or user decisions. Nor should it explain how the software should work[[1]](#footnote-1). It represents associations or relationships between classes[[2]](#footnote-2). The Domain Model are supposed to be a static model, and its purpose is to define package, class names and attributes[[3]](#footnote-3).

The class User has two attributes (Secretary, Member). Larman states that -“if we do not think of some conceptual class X as number or text in the real world, X is probably a conceptual class, not an attribute”[[4]](#footnote-4). If an attribute is to complex, it is better to use a conceptual class and associations[[5]](#footnote-5). My interpretation is that secretary and member should not be attributes, but instead conceptual classes.

Between Member and Boat, there is the association “*Register, remove, change boat”.* In my opinion, I think the association is too specific. I would rather have some more generic association.

Also, you have an association between Secretary and Boat, named Assign berth. To me, berth is a rather important concept, and needs to have its own class.

Overall, I think the model is understandable, and has good readability. The weaker point is that your model is to “dynamic” and almost shows user scenarios. In my opinion, your model pass the passing grade.

1. Larman, C (2005). *Applying UML and Patterns (Third Edition)*. Upper Saddle River: Prentice-Hall. Chapter 9.14 (Heading: *Perspectives: Will the associations be implemented in software*) [↑](#footnote-ref-1)
2. Larman, C (2005). *Applying UML and Patterns (Third Edition)*. Upper Saddle River: Prentice-Hall. Chapter 9.14 *Associations* [↑](#footnote-ref-2)
3. Larman, C (2005). *Applying UML and Patterns (Third Edition)*. Upper Saddle River: Prentice-Hall. Chapter 14.4 *Designing Objects: What are Static and Dynamic modeling* [↑](#footnote-ref-3)
4. Larman, C (2005). *Applying UML and Patterns (Third Edition)*. Upper Saddle River: Prentice-Hall. Chapter 9.12 *Guideline: A common mistake with attributes vs. classes* [↑](#footnote-ref-4)
5. Larman, C (2005). *Applying UML and Patterns (Third Edition)*. Upper Saddle River: Prentice-Hall. Figure 9.23 *Don’t show complex concepts as attributes; use associations* [↑](#footnote-ref-5)