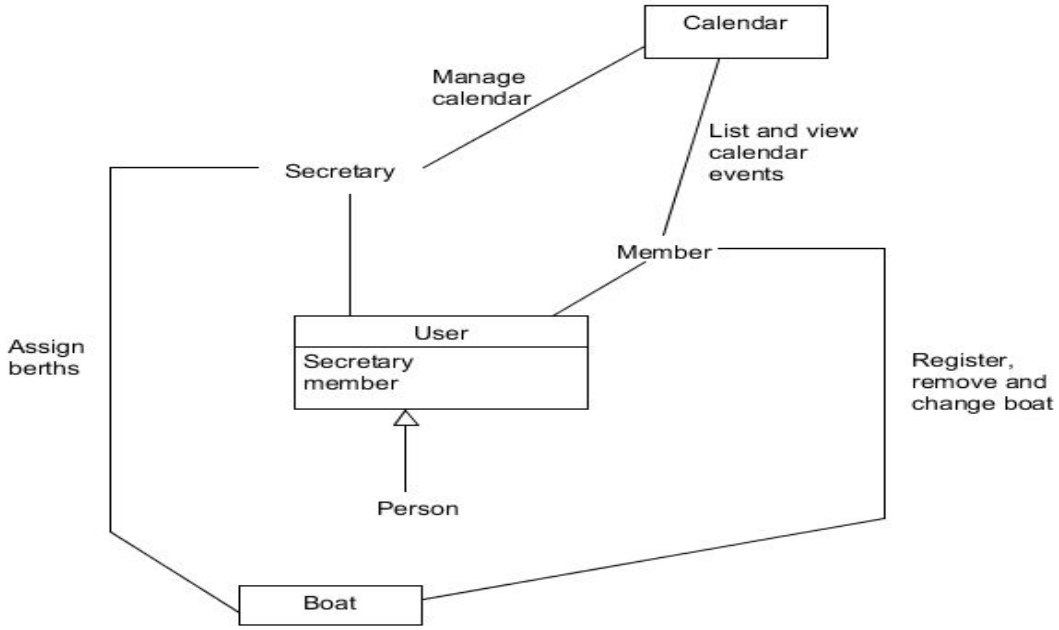

Domain Model 2 (Sverker Söderlund)

1. We think that the attributes within the User class should be separate classes. This is because a Member and a Secretary are real world objects with their own specifications and attributes.

“If we do not think of some conceptual class X as a number or text in the real world, X is probably a conceptual class, not an attribute.” (Larman 2005, p. 146).

2. We also think that the Berth should also be its own separate class considering that you decided to mention it and the fact that it is a real world object. (Larman 2005, p. 146).

3. Since you use Secretary and Member as attributes with associations, it makes even more sense to show them as their own classes. (Larman 2005, p. 146)

4. Calendar should extend "Event" either as an attribute or as a seperate class considering it is more than just an attribute it should probably be its own class as stated in (Larman 2005, p. 146)

5. Member should probably include the association "owns" with the boat class. (Larman 2005, p. 152)

6. The amount of boats a member could own should be listed. As it is part of a "need-to-remember" criteria (Larman 2005, p. 155)

As a developer would the model help you and why/why not?
As a developer who has no insight in how the yacht club functions and no earlier insight in the requirements we feel that it helps me to some extent to understand the fundamental basis of the system. Considering it contains and describes the most fundamental classes and associations of the system. However if some greater insight was needed we would have to go back and read through the requirements, for example to see if there was a limit to see how many boats a member could own. If the member has to own a boat to be considered a member and so on.

Do you think a domain expert (for example the Secretary) would understand the model why/why not?
It depends on the insight the secretary already has, and what it actually needs to know. To understand the basics of what the role of the secretary does we think this model covers it all.

What are the strong points of the model, what do you think is really good and why?
We think that the designer has taken into account that 'less is more' in this case. Many of the conceptual classes derived from the use cases are unnecessary to grasp the picture of how the system works and the designer has taken advantage of this creating a easy-to-read Domain Model.

What are the weaknesses of the model, what do you think should be changed and why?
While we think most of the associations are correct, we would like to see some more (Member -> owns -> boat). We also think multiplicity values could have been used in some instances, for example in the relationship of Member -> Boat. While the model takes advantage of not having too much visual noise we think it could benefit from adding one or two more classes and a bit more detailed associations.

Do you think the model has passed the grade 2 (passing grade) criteria?
We believe that the model could do with just a few more details before we would comfortably pass it. However the designer seems to be on the right track when it comes to excluding too much visual noise which leads us to believe that just a few changes in the structure and a few more details would let it pass with a decent margin.

References

1. Larman C. , Applying UML and Patterns 3rd Ed, 2005, ISBN: 0131489062