Linnéuniversitetet | Fakulteten för teknik | 1DV607

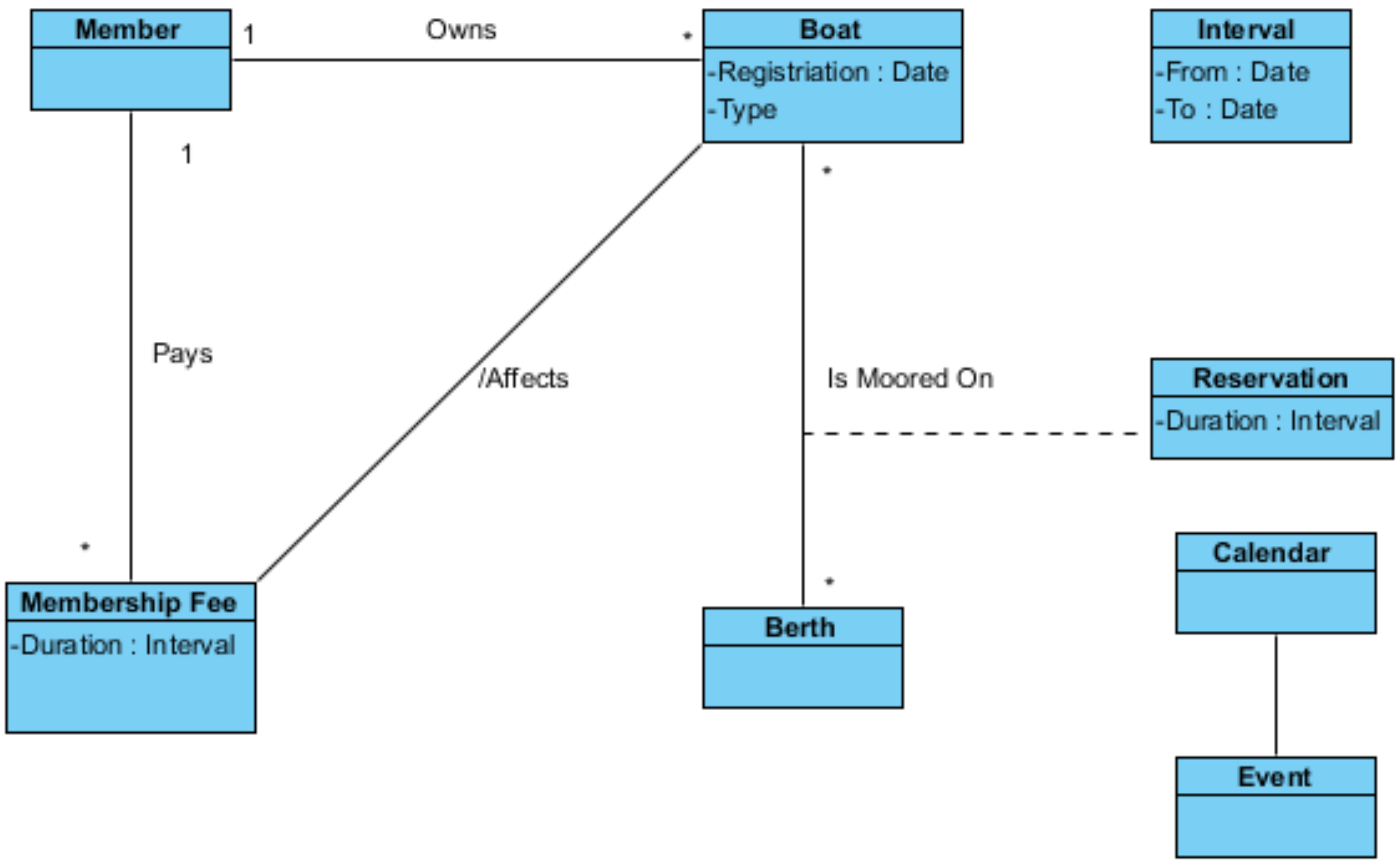
Objektorienterad analys och design med UML

Workshop 1 – Suggested solution

Senast ändrad 2019-09-27 10:27 av ***Tobias Ohlsson***

Below is my proposed solution for workshop 1, passing grade You must be registered for the course and logged in to take part of this. This solution is not intended as an answer key, but rather a basis for comparison with your own model.

The solution will be opened AFTER the final workshop deadline has passed and submissions have been examined.



Discussion

Three key conceptual classes are Member, Boat, and Berth. Most of the problem revolves around these three classes and to sort out the relationship between these classes is critical to understanding how the boat club really works. It is important to scope the domain model to the problematic area under investigation, i.e. the domain model is not meant as a complete "database design" (though DB-design does have a lot in common with domain modeling), and that if it is not in the domain model it is "not there". Finding a good balance between requirements – domain model – "data dictionary" (for more detailed data representations) is essential and quite hard.

To deal with the seemingly cumbersome procedure of the registration of a boat it is really only required that we know the registration date, and we can then determine how the boat should be handled regarding offseason and preseason. The rule is explained relatively clearly in the requirement itself. I also want to clearly show that the membership fee is affected by which boats a member owns, although this obviously is a derived association.

I choose to deal with the different types of boats using a simple attribute. Perhaps this should be elaborated into a class of their own at a later stage (eg if different types of boats must be handled in different ways and/or to have different attributes), however there is nothing in the requirements that indicate that the different types of boats should have some special properties or be handled in any special way. It rather seems as if the boat club is only interested in the information itself and to then build a generalization/specialization hierarchy feels like overkill.

To connect a Boat with a Berth an association class, Reservation, is used. This class connects a specific Boat with a certain Berth in a given time interval. This offers the opportunity to handle history around the Berths and Boats a certain Member have had. Association Classes are a powerful tool that allows us to clearly show how two concepts are interrelated and simultaneously add information associated with each instance of the relationship. Unfortunately, not many UML tools support association classes.

The Membership Fee also got a period added to know what year that the membership fee has been applied for. The requirements concerning the membership fee are quite peripheral and it really only mentioned that the membership fee is affected when a member registers a boat. Therefore, I have chosen not to elaborate that part too much at this stage.

Calendar and Events are very simply described, and it would probably have been enough to have one class "Calendar Event" as Calendar could be considered a report class (just a visualization of a number of events). Possibly, one could imagine different types of Events (competition, etc.), but I chose to keep it simple for now. In this situation asking a domain expert had been appropriate.

Multiplicities

Multiplicities for associations in domain modeling are quite context-dependent. Larman states “The multiplicity value communicates how many instances can be validly associated with another, at a particular moment, rather than over a span of time.” It is easy to interpret this as you should always only model things in a snapshot way of thinking. However, Larman also states “The multiplicity value is dependant on our interest as a modeler and software developer”. In the case of Boat and Berth, we are more interested in being able to maintain a history of Boat-Berth relations (as we need this information to automatically schedule Boats to Berths.), even if one Boat can only occupy one Berth at a time. If we were interested in the current situation only multiplicities would be different and possibly also the name of the association would be “Currently Moored On” or something like that.

Looking at the multiplicity between Member and Boat we have the opposite interest. A boat can be owned by one and only one Member, even if it is quite likely that the same boat could actually change owners over time. In this case, we are not interested in maintaining a history of ownership (as this is not part of the requirements) but rather the actual situation and the domain rule that there should be no “loose” boats.

Authentication

Authentication is omitted because it did not really belong to the problem area of “boat clubs”. The mix of authentication in the domain model confuses rather than facilitates understanding of the real problem area. The requirements regarding authentication are also clear, and this problem area is well explored by others. A scenario to consider is that a domain expert should recognize the domain model; an expert on boat clubs would probably not understand the authentication part.

If authentication needs to be sorted out that’s best done in a separate domain model focusing authentication. One can then show that for an expert on authentication.

It is also interesting that some of you happily threw yourself over this as you may recognize it (authentication) and in some cases complicated models designed for just that, and tried to link this into the actual domain model.

Secretary and Treasurer

I also believe that it is unnecessary to introduce Secretary and Treasurer in the domain model at this stage. There is nothing in the requirement that indicates that the boat club really treats any information about the Secretary or Treasurer. To then take the Secretary and Treasurer, just to show that “they are doing something” is pretty pointless. What someone (an actor) do is best viewed in a Use Case Diagram. It had been a different matter if the requirements pointed out that it was, for example, important to know which Secretary made a reservation. Again, in real life, this is more an assumption that should be brought back to the domain experts and ask them.