Home Exercise #2 Design and Architecture

Intended learning outcome

After passing this exercise, the student will be able to exemplify some of the most common ways of representing the design and architecture of a system using the UML modeling language.

Problem description

We continue with the "Molvo" system as introduced in Exercise 1 and add the following information found in the analysis:

- The customers of the rides, hereafter called passengers, need to register a membership profile at "Molvo" to start booking rides.
- When booking a ride, the passenger needs to pay in advance to "Molvo", who sets the account with the car owner at the end of each month.

Task

Your tasks for this week are the following:

a) Provide the overall **high-level workflow** of using the Molvo system *using a UML Activity Diagram*. This granularity of the workflow should be on use case level, i.e. the different activities should be the use cases carried out by different actors of the system.

b) Create a **domain model** in the form of a *UML Class diagram* from the Problem description above and in Exercise 1. Your domain model should cover the key *classes*, their most important *attributes*, as well as potential *generalizations* between classes. When a potential generalization arises, you should justify (in 1 sentence) why you opted for a generalization between classes, or why you opted for an alternate modeling solution. Define the key *relationships* (i.e. associations and compositions) between classes and specify *multiplicities* for all relationships. Keep an eye on a meaningful *composition hierarchy*. Provide *meaningful names* for classes, attributes and relationships, but no operations are needed. Your domain model should cover all key information mentioned in the description. *You do not need to include everything in a single diagram*, rather you should split the information into multiple diagrams.

c) Draw a *UML Sequence diagram* for **modeling a scenario** where a passenger is booking one or many rides, starting with the passenger searching for available rides and ends with the passenger paying for the ride(s). Note: the passenger can book many rides in the same session and pay for them all at the end. Don't forget that the car owner needs to be informed about the bookings. You need to use at least one fragment.

d) Define the **stateful behavior** of the class *Ride using a UML StateMachine diagram*. First, you need to define the triggers and the actions used in the state machine.

e) Describe the **overall architecture** of the complete Molvo system. You don't need to use UML; an *informal block diagram* (with hierarchical boxes and lines) is sufficient. Give an example of a quality factor that drives your suggested architecture. Provide a brief (1-sentence) description of each key component. Your architecture does not need to be too detailed, about 7 boxes will be enough.

You may make more assumptions of features of the system than those given in the Problem description, but in that case, you shall explain your assumptions in the solution.

Report

High-level workflow using UML Activity diagrams

Domain model using UML Class diagrams.

Scenario specification using UML Sequence diagrams.

Behavior specification for class Ride using UML StateMachine diagrams.

Architecture description using informal block diagrams together with explaining text.

Grading criteria

General requirements: All diagrams should be *readable,* but they may be hand-written and scanned. The UML diagrams should follow the *correct standard UML syntax* and *semantics*. *Naming conventions* should be followed in all diagrams. The information provided in the various diagrams should be *consistent* with each other. The various diagrams should be *meaningful* for the domain and closely *connected to the textual problem description*.

Extra grading criteria for domain modeling:

Completeness wrt. problem description, complete specification of relations, justified design decisions for the use generalization, appropriate use of relationships vs. attributes, containment hierarchy.