Home Exercise #1 Requirements

Intended learning outcome

After passing this exercise the student will be able to capture, explain and exemplify some of the most common ways of specifying the requirements of a system.

Problem description[1]

A car manufacturer and retailer chain “Molvo” wants to expand the services they are offering their customers and offer a car-ride sharing service. They hired your company to develop the application, but first they need your help to elicit the requirements and write a specification for the project.

The following information has been collected so far from interviews:

- The application should enable car owners to list their rides and available number of places. Interested users can then book/unbook a ride as well as chat with the driver.
- In order to list rides as a car owner, you need to first create a car-owner profile where you are required to submit the copy of your ID card, your vehicle make, model and registration as well as some other information. This information is then verified by “Molvo” and if no issues are found the account is approved. The member data needs to be protected and the management should comply with GDPR.
- It shall be possible for passengers to book rides a month ahead, for instance, for daily commuting from home to work.
- The application should be available both as an app and in a web-browser.
- After a ride, it is possible to rate both the car owner and the passengers. The administrator can follow up if someone is rude and expel them from the system.
- The back-end of the system is run by administrators run on an internal network with ordinary PCs and a database management system.
- It is important that it is impossible to double book a ride and that un-booked rides can be booked by others in a timely manner. For a ride un-booked less than 48 hours in advance 50% of the ride price is retained. (If a ride remains booked, then the full ride price is paid)
- A payment system serves the car owners using services from major credit card vendors or direct bank payment at major banks in the country where the system operates.
- The company will take a 3% commission on all transactions of which 1% will be donated to environmental programs. The sustainability policy must be well-visible on the company homepage.
- In order to differentiate themselves from the competitors “Molvo” wants to connect this application to simulation models they use for car development, to estimate the reduction in CO_2 emissions for each trip when sharing a ride instead of taking your own car. This work is also run on the internal network with a special simulation server.

[1] Original idea: L. Buffoni

Task

Your tasks this week are:

a) Create a *UML Use Case diagram that contains all relevant actors and use cases* mentioned in the specification above. Clearly identify the boundaries of the system to be developed. Give appropriate names for these actors and use cases by following the naming conventions.

b) Write *detailed description for two key use cases using structured scenarios* (option #1 in the lecture). Please remember that such scenarios should the description texts for the use cases shall not just describe a basic function of the system, such as login.

c) Write four (4) *functional requirements* of some part (component) of the system. Don't forget to describe which system part you are writing about. Follow the best practices of writing requirements discussed during the lecture.

d) Write four (4) *non-functional requirements* of the system including at least one design constraint and at least one quality requirements.

e) Create four (4) *user stories* for some part (component) of the system. Don't forget to describe which part of the system you are writing about.

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

Report

Include the UML Use Case diagram on a separate sheet.

Use-case description texts. Requirements texts. User story texts. (1-2 A4 pages)

Grading criteria

Use case diagrams: Completeness of actors and use cases with respect to textual description, consistent UML syntax, adherence to best practices.

Detailed use case descriptions: Meaningful and detailed scenarios, adherence to best practices

Functional requirements: Relevant functional requirements which adhere to best practices taught in the course

Non-functional requirements: Relevant non-functional requirements are provided which adhere to best practices taught in the course

User stories: Relevant user stories are provided which adhere to best practices taught in the course