BDSA Assignment 1

alsk,

September 16, 2021

1 Software Engineering

1.1 Exercise 1

Knowledge acquisition is not sequential means that if you work on a project and you have acquired information which you build the system about, it is then entirely possible to be almost done with the system, and then learn something that invalidates the entire system, meaning you have to start from scratch again. This also means that new knowledge can further proof how old knowledge is no longer valid, and has to be changed as a result.

A real life historical example of this, was before 1975 where car gas was sold with lead because scientists believed that the car would run better and the gas would last longer, however this was later disproven, and it was discovered that the lead had negative health consequences for the people, especially in the cities, where a lot of cars were present. The idea about lead in gas was then scrapped because knowledge acquisition is not sequential.

1.2 Exercise 2

Requirement design:

- "The ticket distributor will use PowerPC processor chips."
- "The ticket distributor provides the traveler with an on-line help."

The first statement is a non-functional requirement, made to describe a specific constraint in the development of the product.

The latter statement was made during requirement design and is a functional requirement, describing the supportability of the product.

System design:

- "The ticket distributor is composed of a user interface subsystem, a subsystem for computing tariff, and a network subsystem managing communication with the central computer."

This statement describes a division of the system, which enables the programmers to have a more efficient workflow, as they can work on individual parts easier.

1.3 Exercise 3

The word account is used on 4 occasions in the text:

"Assume you are developing an online system for managing bank accounts for mobile customers."
In this sentence, "account" is merely used in the application domain, and it is simply a use-case description, ultimately not a part of the solution domain.

"A major design issue is how to provide access to the accounts when the customer cannot establish an online connection."
This sentence describes a specific design issue, Design issues are in general in the solution domain, no

matter what type of design it is. In this example it seems to be used as a system design.

"One proposal is that accounts are made available on the mobile computer, even if the server is not up."
It seems like this example of account is used as a solution domain, which is a part of the system design.

"In this case, the accounts show the amounts from the last connected session."
In this last use of "account", it is refering to the fact that its showing the users information abou the amounts from the last connected session. As it is showing information, it means that it would be classified as an application domain concept.

1.4 Exercise 4

When working with a bridge or an aircraft it is obvious and apparent how the project is coming along. This is because a bridge or an aircraft is a physical object that needs to be built, and we can easily measure how close to completion we are. On the other hand, a software project is not something physical we can measure. We can of course try to measure how far we have gotten by looking at project requirements. However, not every requirement takes the same amount of time to complete and it is very likely to encounter obstacles which were not visible from planning. This could explain why software often is delivered late and over budget.

Also, if we compare a bridge to a software project, the bridge would be more visible and relatable to people in general, as most people have experienced "building" things in their life, either with toys, or in real life as well. However people has most likely not tried programming a project before, meaning that the concept of programming becomes more abstract to more people than the building of a bridge. This can likely also be a reason for late projects that are over budget, especially in cases where the people budgeting and doing deadlines have little to no experience with programming.

1.5 Exercise 5

Functional requirements:

- "The TicketDistributor must enable a traveler to buy weekly passes."
  Describes a service of the system.
- "The TicketDistributor must always be available."
  Describes a service of the system.
- "The TicketDistributor must provide a phone number to call when it fails."
  Describes a service of the system.

Nonfunctional requirements:

- "The TicketDistributor must be written in Java."
  Describes a constraint of the system's development.
- "The TicketDistributor must be easy to use." (Cannot measure "easy" since its subjective)
  Describes a constraint on the system's operation.

1.6 Exercise 6

In terms of software development, modeling helps developers to visualize and understand the program's application domain with objects and connections between these objects. These connections and objects are then supposed to represent real life concepts, illustrating the solution domain. As software development is not physical, it means modeling is an even more important and powerful tool for developers to communicate with each other, and visualize how the program is supposed to behave in specific situations. All in all, modeling is a scientific method and a visual representation of a system that is supposed to help people by answering questions they might have about the system.