BDSA - Assignment 02

September 2022

Introduction

This .pdf contains Mandatory Assignment-02 for Analysis, Design and Software Architecture made by Lukas André Rasmussen (lanr) & Weihao Chen Nyholm-Andersen (weny). The document is split in two parts, in which the first part contains C# and second part contains the Software Engineering exercises.

Git link

assignment-02

1 C#

1.1 Classes

We would ideally implemented the solution using null type values for the dates, making it easy to check if a student has ended or is graduated, but since we are not sure if that is allowed, we did not go with that implementation.

1.2 Records

assignment-02/blob/main/Assignment2/Student.cs

1.3 Types

The key differences between a class and a Struct, is that a Struct is a value type, whereas a class is a reference type, meaning if you want to assign a Struct to a variable, the Struct is copied and not referenced.

A Struct can not be abstract, nor contain abstract or virtual functions, as well as not being able to inherit from other classes, or be a base class.

Records are mainly used, whenever you want to use reference variables with the possibility of value-based equality and comparison, opposed to classes.

1.4 Extension Methods

```
return xs.SelectMany(a => a)
          .Select(a => a);

return ys.Where(a => a % 7 == 0)
          .Where(a => a > 42)
          .ToArray();

return ys.Where(a => a % 4 == 0)
          .Where(a => a % 100 == 0 ? a % 400 == 0 : true)
          .ToArray();
```

1.5 Delegates / Anonymous methods

assignment-02/blob/main/Assignment2/Delegates
assignment-02/blob/main/Assignment2.Tests/DelegatesTests

```
// delegate 1:
Func<string, string> reverseFunc = (a) =>
                                string.Concat(a.Reverse());

// delegate 2:
Func<decimal, decimal, decimal>
                        decimalProduct = (a, b) => a * b;

// delegate 3:
Func<string, int, bool> isEqual = (str, n) =>
                                int.Parse(str) == n;
```

1.6 LINQ

assignment-02/blob/main/Assignment2/Queries.cs
assignment-02/blob/main/Assignment2/Wizards.csv
assignment-02/blob/main/Assignment2/Wizard.cs
assignment-02/blob/main/Assignment2.Tests/WizardTests.cs

2 Software Engineering

2.1 Exercise 1

Scenario

Scenario is a way to propose how a specific system or application is used. Scenarios is viewed from the outside, this means that a scenario has the perspective of an user with given user examples.

An example on how a scenario could pan out is that you have a **Purpose** this purpose is often described as "what is the purpose of this scenario". Thereafter you would have the **Individual** that is going to follow through the scenario. Step 3. is what **Equipment** is needed for following through the scenario in forms of browser, hardware etc.. Lastly **Steps** on the scenario, like a movie scene, that describes how the scenario is played out.

Use Case

Use Case is a method that allows developers to fully understand the requirements that is needed for a specific system. Use Cases are often built on modelling which deepens the understanding of the system that is in discussion.

An example of how Use Cases can be used is that you would have a **Name** for the Use Case that summarizes what the purpose is for the Use Case. Thereafter we define who the **Actor(s)** are in the Use Case. Actor is a role that can be given to anyone that meets the requirements to use the system i.e. a BookBorrower - borrows books or GoogleUser - uses Google. We then note what the Actor must go through in a **Step-by-Step** list in which it specifies what actions the Actor must take at specific points in time during the use of a specific system. Lastly the **Condition(s)** that makes the actor The Actor, i.e., BookBorrower need to have an account that makes them accessible to borrowing books or GoogleUser must have a Google Account in order for them to be a GoogleUser.

Difference between Scenario and Use Case

The big difference between Scenarios and User Case is that Scenarios is like a discussion that you discuss with your/a client about what the system/product must be able to do. Whereas Use Cases are more specific in which it takes the Scenarios and models them to fully understand the system requirements and make them more precise.

2.2 Exercise 2

Computer-Based System - User Requirement

One requirement that may appear in a Computer-Based System is **User Requirement**. Since let us say that we are developing a new software, in this case we need User Requirement since depending on what the user needs the software will drastically change. I.e. lets say we are developing a new RPG game, then we need to know what services and requirements RPG players need in order to make an RPG game that is aimed for RPG players.

Computer-Based System - System Requirement

System Requirement is needed in order to get a more specific description on the functionalities of a software. System Requirement is very specific on what is being developed and implemented in a software. I.e. the RPG game we are developing needs to have a combat system, in which we can define as a System Requirement, since an RPG game is nothing without a combat system. We can go even further into specifics and say that in the combat system there needs to be a health system, mana system, stats etc.

Computer-Based System - Functional Requirement

Functional Requirement are statements that a specific system or service should provide. Depending on the input of the system should react in a particular way or given any situation the system should be able to behave depending on the given situation. Functional Requirement is a way to tell users and/or developers how a specific function should work, their input and output or even exceptions.

Computer-Based System - Non-Functional Requirement

Non-Functional Requirement are often more critical than Functional Requirement. Non-Functional Requirement is more critical in the sense that if a Non-Functional Requirement hits the system or service as a whole, where as Functional Requirement only hits the individual functionalities of the system or service. An example would be if a certain system is not secured for outside attacks then the system would be unusable since it is not safe for users to use the system.

2.3 Exercise 3

Below is Figure 1 which shows our Use Case Diagram for a ticket distributor for a train System. The system contains mainly of two actors (Traveler and Central Computer System) where the Use Cases included are: Buy OneWay Ticket, Buy Weekly Card, Buy Monthly Card, Update Tariff including some exceptional cases: Time-Out, Transaction Aborted, Distributor Out of Change and Distributor Out of Paper.

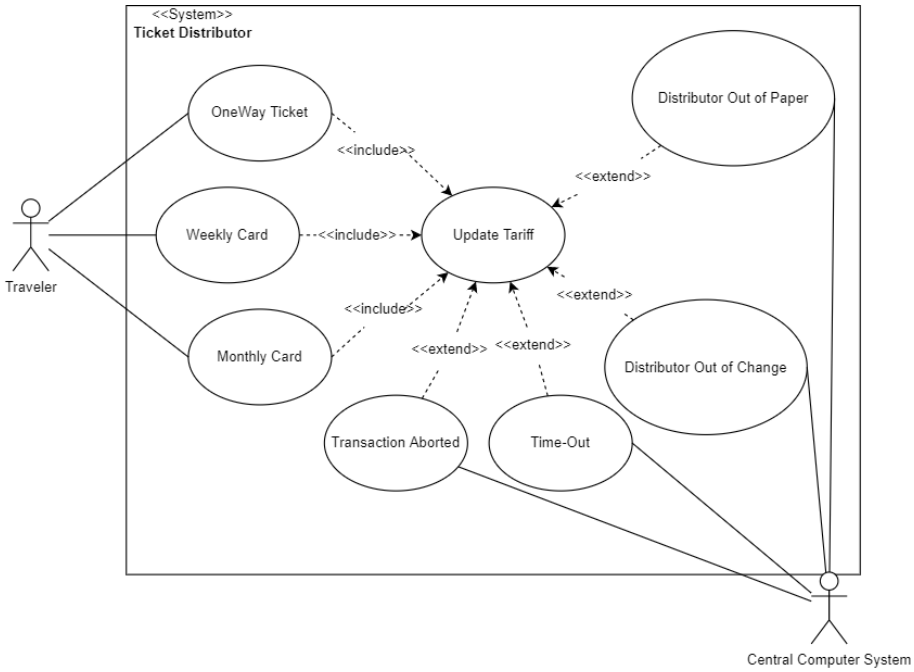

Figure 1: Use Case Diagram

2.4 Exercise 4

Discover formulations in the requirement that are ambiguous

- "Medarbejdere skal digitalt understøttes i udførelsen af deres kerneydelser" - this requirement is a bit vague in definition. The reason being that digital support can come in various types and forms, and therefore should be more specific according to what the workers want as digital support. Besides that the workers as a whole is a bit vague as we do not know what type of workers it is and therefore ambiguous.

- In the following requirement it says "Det gælder både i forhold til at effektivisere tunge arbejds gange via genbrug af data, deling af data [...]" - this requirement is vague in the sense that we do not know what kind of data we are handling. Depending on the type of data a system needs to be built according to that specified data that needs to be manipulated or shared.

Is there any information missing in the requirement?

- From what we could conduct the requirements for this system are a bit vague - in the sense that we have no idea what the Functional Requirements are for this application/system. Everything that is stated is how the application should help workers and how it needs to support them, but we have no idea how the application should function as a system.

- Another view-point we could say is that this information that is given is more like a scenario on how the application should function. If we were to make an application from the given information then we would need to make this scenario into a use case that is more concrete on what functions the application should include. By doing so it is easier as developer to create such application that satisfies the requirements that they want.

The requirement specifies a set of non-functional requirements. What is problematic about there formulation?

- The requirement that is stated has a set of non-functional requirements that simply states that they need a Product in a set Environment that is Dependable. Now this is very vague in the sense that we do not know what kind of dependable it needs to be. Like does the system require any specific functions like a Wi-Fi or does the application need an account? Must the application only work for mobile or can it be used on other platforms as well? There are just too many non-specific requirements that needs to be specified in order to create something that will satisfy the client.

Rewrite the requirement according to what you identified as problematic in the three bullet points above

In the following, the bold text are the rewritten parts:

5.1 Brugervenlighed

Medarbejdere, **såsom socialrådgiver**, skal digitalt understøttes i udførelsen af deres kerneydelser, og derved skal den digitale understøttelse være medvirkende til, at kommunen er et attraktivt sted at arbejde.

Kommunen har forskellige faglige målgrupper, som møder borgerne forskellige steder, og som arbejder, hvor borgerne er. Det kan være på gaden, i borgerens eget hjem, men også på dag- og døgninstitutioner uden for kontoret. Derfor er det vigtigt, at store dele af løsningen kan afvikles på mobile enheder, **det vil sige at produktet vil primært være i form af app som kan blive hentet ned på telefon, tablet og eventuelt computer**

Det er altafgørende, at medarbejderne føler sig godt understøttet af den nye digitale løsning. Det gælder både i forhold til at effektivisere tunge arbejdsgange via genbrug af data **til hjælp for socialrådgiveren til at simplificere deres arbejde**, **mulighed for deling af dette data til andre socialrådgiver** og ikke mindst, at løsningen er medvirkende til god lovmedholdelighed i sagsbehandlingen, hvilket tilsammen kan understøtte en faglig stolthed.

Et andet og vigtigt parameter er, at kommunen gerne vil fastholde sine dygtige medarbejdere med en løsning, der understøtter og guider dem i deres daglige opgaveløsning. **At socialrådgiverene har adgang til de ønsket data og værktøjer de har brug for i løbet af en arbejdsdag.** En stabil og driftssikker løsning, **som ikke bliver påvirket af internetbrud eller strømafbrydelse**, er en medvirkende faktor til større tilfredshed med ansættelsen i kommunen.

2.5 Exercise 5

Identify actors that interact with a music tracker software system

There are many actors that comes to mind during the video, one of them actor roles could be **music composers for games**. Since a game already has a large amount of bytes they need to simplify and compact the music so that it would be able to loop - or at least that was the case in the past. Other than **music composers** for games there are also of course the normal music composers that uses these software in their production of music.
A rare but still viable actor could also be **students** that is learning about music production. As the video mentions the software now a days are easy to use and to learn, therefore many schools implement these types of software or even hardware for educational purposes.

Formulate three use cases in structured language that a software music tracker system has to support

For simplicity purposes all the actors are the same in order to uphold the same concept throughout the exercise

Use Case #1 - Creating and saving a track

Main Actor: Video Game Music Composer

Main Scenarios:

1. The composer selects 'Create Project'
2. The software creates a blank project for the composer to use
3. Composer selects 'Save' or 'Save as...' in order to save the project they have created.
4. 'Save' saves the project to an already existing save file
5. 'Save as...' saves the project with a given name and a filepath

Use Case #2 - Import of a project

Main Actor: Video Game Music Composer

Main Scenarios:

1. The composer selects 'Import Project'
2. The software shows a pop up where the composer must choose the project file they wish to import
3. After selecting the project file, the software will read and render all the sub-files that is contained in the given file and thereafter loads it in, into the software
4. The composer can modify the project however they wish

Use Case #3 - Export of a project to an audio file format

Main Actor: Video Game Music Composer

Main Scenarios:

1. The composer selects 'Export Project'
2. The software shows a pop up in which the composer must choose an audio file type they wish to extract the project as (in forms of MP3, WAV etc...)
3. After selecting the software will compress it down into the desired file type and save it into a given filepath

Express three non-functional requirements for a music tracker software system

The **color-coding** of the music tracker software is unique in a way that it distinguishes the different element the composer is working on. This can be an essential way of the software since having a monotone of colors would be difficult for the user to organize what they are working on. This applies both for the software and also the hardware in itself.

Another essential part of the music tracker software is that it can **preview** a selected part of the music. This makes it easier for the composer to listen if the selected part is the right part or it needs adjustments.

Lastly a unique part of this hardware is that it shows the track in a **vertical viewpoint** and not a horizontal like other mainstream software. This, as he said in the video, gave him a new perspective on how to view the track and made him view the flow of it in a different way.

2.6 Exercise 6

Canteen Payment-System

Main Actor: Customer
Step-by-Step:

1. Customer enters the canteen and grabs their desired item (food or drink)
2. Depending on their item they either scan or weight their product
3. When the customer have scanned their items they checkout (or the cashier checks them out depending on if the customer chooses self-service or counter)

NOTE: *if the customer uses Grab'n'Go App they pay via MobilePay and only scan the QR code at the counter (does not work with weighted foods)*

Requirements

- Self-Service is a the core of the Canteen Payment-System. In Self-Service you simply weight or scan and checkout.
  This is a **Functional Requirement**, since the Self-Service requires an input before it can give an output. The input in this case would be the products the customer(s) want to buy and the output would be the total price for the item(s).
- If the customer were to choose the counter it is required for a someone to stand behind the counter in order for the customer to continue with their payment.
  This would be a **Non-Functional Requirement** since if there is no one behind the counter that would mean the whole Payment-System is not set in action and you are stuck - in other words the Payment-System is unavailable.
- If the customer desires to use the APP it is a requirement that the customer has MobilePay. Therefore if the customer does not have MobilePay they are unable to pay for their product via the APP.
  This is a **Non-Functional Requirement** since MobilePay is essential in this requirement, thereby not having MobilePay the whole Payment System is unavailable.