BDSA
Assignment2
23-09-2022

Name: *eikb, phimo, ssbo*

CSharp - Types

Explain some of the key differences between class, structs, record struct and record class and give examples of where you would use each of them.

Class types have reference semantics which means that a value points to a memory location, so when you try to copy a value it actually copies a memory location.

Structs have reference semantics which means that it copies the actual value and not the memory location. Structs have less memory overhead, so structs can save memory.

Records are good when your data should not change. Records are also convenient when you want your comparisons on the instances to be based on value, since two identical records will be equal if the properties of the two are equal. A class would not equal another class when all its properties were equal.

CSharp - Extension Methods

Flatten the numbers in xs:

```
var flatten = Extensions.xs.SelectMany(c => c);
```

Select numbers in ys which are divisible by 7 and greater than 42:

```
var filtered = Extensions.ys.Where(x => x % 7 == 0 && x > 42);
```

Select numbers in ys which are leap years:

```
var leapYears = Extensions.ys.Where(x => (x % 4 == 0 && x % 100 != 0) ||
(x % 4 == 0 && x % 100 == 0 && x % 400 == 0));
```

CSharp - Delegates / Anonymous Methods

A method which takes a string and prints the content in reverse order (by character):

```
public delegate void Reverse(string s);
Delegates.Reverse r = s => Console.WriteLine(new string(s
    .ToCharArray().Reverse().ToArray()));
```

A method which takes two decimals and returns the product:

```
public delegate double Product(double f1, double f2);
Delegates.Product d = (f1, f2) => f1 * f2;
```

A method which takes a whole number and a string and returns true if they are numerically equal. Note that the string ” 0042” should return true if the number is 42:

```
public delegate bool IsEqual(string s, int i);
Delegates.IsEqual d = delegate(string s, int i) {
            try {
                return int.Parse(s) == i;
            }
            catch {
                return false;
            }
        };
```

Exercise 1

Describe the difference between a scenario and a use case. Describe for each of the two concepts when and for what they are used.

Use cases describes interactions between a user and a system using a graphical model, such as uml, and structured text. Use cases can be used in system design since the people developing the system can find the use cases helpful, where as stakeholders don’t find use cases helpful.

Scenarios is a description of how the system can be used in a specific situation. It is examples of user interaction sessions, written as structured text. Scenarios can be used when interviewing stakeholders to discuss if the system does what it is intended to do. People find it easier to relate to real life examples, rather than a diagram showing all possible interactions.

The difference between use cases and scenarios is that use cases use a graphical model to show the different interactions an actor can have with the system, where as scenarios gives examples in more detail on some of the possible interactions. Some people see each single use case as a low level scenario, others view each use case as a set of scenarios, where each scenario is a single thread through the use case.

Exercise 2

Identify and briefly describe four types of requirements that may be defined for a computer-based system.

1. *Functional* A requirement that describes what the application should do: a feature. For example: the user should be able to send messages to other users in their communications list.

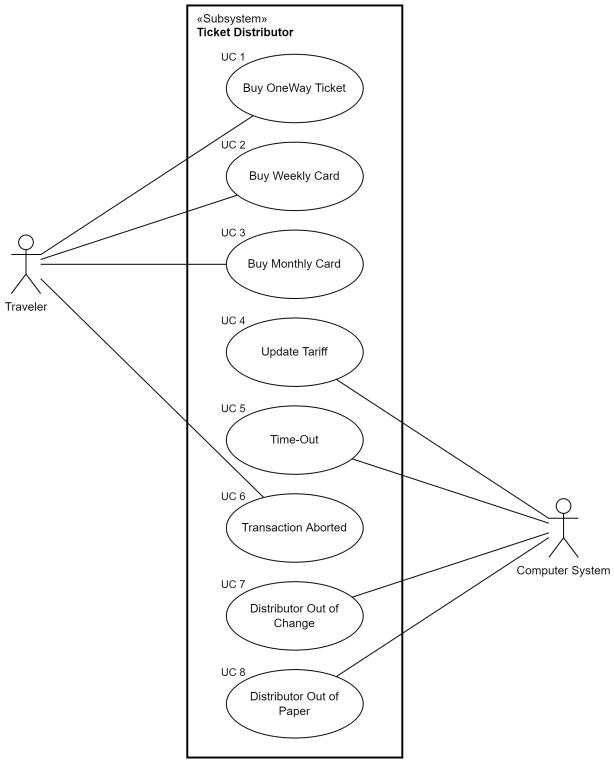
2. *Functional* A requirement that is created from the need of a non-functional requirement. For example a non-functional requirement outlining that the system should be authenticated might create the need for a user login system with a username and password.

3. *Non-Functional* **Efficiency Requirements:** Describes how the system should achieve efficiency in certain areas. For example, the system might be constrained by the hardware, and such the memory usage or processing load might need to be adjusted accordingly.

4. *Non-Functional* **Organizational Requirements:** The programming language or framework the system must use. Maybe the organization has a particular hosting platform or business-to-business agreement that requires the need for a specific language and framework.

Exercise 3

Draw a use case diagram for a ticket distributor for a train system. The system includes two actors: a traveler, who purchases different types of tickets, and a central computer system, which maintains a reference database for the tariff. Use cases should include: Buy OneWay Ticket, Buy Weekly Card, Buy Monthly Card, Update Tariff. Also include the following exceptional cases: Time-Out (i.e., traveler took too long to insert the right amount), Transaction Aborted (i.e., traveler selected a cancel button without completing the transaction), Distributor Out of Change, and Distributor Out of Paper.



Exercise 4

In the following, you find a requirement that is specified for the "samlet socialfagligt it-system" (also called VUM 2.0):

5.1 Brugervenlighed
Medarbejdere 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 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, deling af data 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. En stabil og driftssikker løsning er en medvirkende faktor til større tilfredshed med ansættelsen i kommunen.

Discover formulations in the requirement that are ambiguous.

Is there any information missing in the requirement?

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

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

The following will be answered in danish since the text is in danish.

Det er tvetydigt at skrive at medarbejderne skal digitalt understøttes i udførslen i deres kerneydelser, for hvad betyder det at blive understøttet, i hvilken grad og på hvilken måde skal de understøttes. Det er også lidt uklart at skrive at store dele af løsningen skal kunne afvikles på mobile enheder fordi dette begreb er relativt. Hvem vurderer hvad og hvor meget der er store dele?

Krav 5.1 om brugervenlighed er meget generelle og overordnede krav om, hvilket giver plads til fortolkning hvilket ikke er ønsket i et krav. Der er meget gentagelse, fx skriver de at medarbejderne skal understøttes i deres arbejde flere gange på forskellige måder, uden at komme med noget konkret information om hvordan dette kan testes.

Der mangler speciefik information der kan blive testet, fx er det er svært at teste om kommunen er et mere attraktivt sted at arbejde pga denne løsning. Det er altså problematisk at de skriver kravene på en måde der ikke er testable.

Omskrivning af krav 5.1:
Medarbejdere skal digitalt understøttes i udførelsen af deres kerneydelser og daglige opgaveløsning. Det gælder både i forhold til at effektivisere tunge arbejdsgange via genbrug af data, deling af data og ikke mindst, at løsningen er medvirkende til god lovmedholdelighed i sagsbehandlingen. Efter to timers træning med programmet skal 9/10 medarbejdere rapportere at synes at dette system vil hjælpe dem med at udføre deres kerneydelser og daglige opgaveløsning.

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 80 procent af løsningen kan afvikles på mobile enheder.

Løsningen skal være stabil og driftsikker, hvilket betyder at raten af systemfejl skal være under 1 procent og at tiden det tager at genstarte efter en systemfejl er under 2 timer.

Exercise 5

Watch the video below in which a domain expert explains music trackers. Imagine that the narrator explains only the hardware trackers he demonstrates and that there have never been any music trackers that were implemented as pure software systems.

Identify actors that interact with a music tracker software system.

- Musicians, who use the software to produce professional music.
- Game Developers, who are interested in creating music for their games.
- Students, who use the software for learning.
- Hobbyist who use the software for free time projects and recreational music creation.

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

Use case 1

- Actor: Musicians
- Action: Making Music
- Description: The musician inputs sounds, percussion and other instruments using notes, placing them mindfully on the vertical track as to create a beat, flow and other music-theory

related constructs, that would, after being strung together, constitute a song. The musician then saves the song in a relevant file format and uploads the song to a website of his choice.

Use case 2

- Actor: Hobbyist
- Action: Making sounds
- Description: The hobbyist inputs sounds into the vertical track and uses the inbuilt manipulation features to change the sounds to fit his ideas for his DND roleplaying campaign, so he can bring his groups fireballs and sword clashes to life. He then downloads the sounds onto his computer in an mp3 format, so he can easily move the file around and play it with his group when they are roleplaying.

Use case 3

- Actor: Game Developer
- Action: Creating music and sounds for game development
- Description: The game developer uses the software for creating music and sounds, since his studio did not have the budget for hiring professional musicians. They were recommended tracker software, as it has a lower barrier to entry and could produce proper sounding loops and with some effort from a inspired developer, could produce near-professional sounding music. These files are then downloaded, played and used in the game.

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

- The software should implement a sensible user-input flow using keyboard shortcuts, that mimics the style of pure hardware trackers.
- The software should have enough sounds, instruments and effects as to enable proper music creation and creativity.
- The software should implement a vertical track system, where the user inputs notes and sounds on a vertical-sliding input field, consisting of rows and columns.

Exercise 6

For this exercise, we assume that the canteen at IT University wants to establish a new payment system. As an example for ethnographic work, use ten to 20 minutes when going for lunch to observe how you and other customers of the canteen interact with the current payment system.

During your observations take notes.

Based on your observations and notes, write down a use case in structured language that a canteen payment system has to support. For this use case write down three requirements and categorize them as functional or non-functional.

- Actor: Customer
- Action: Paying for food taken at the buffet
- Description: The User has taken food from the buffet and a drink from the refrigerator and wish to pay for it with their credit card at a customer operated terminal.
- Requirement: The payment system has to support selecting different foods. This has to implemented in a fashion that is both easily expandable to the wishes of the canteen and is easily navigable by the customers. This would be a non-functional requirement
- Requirement: The payment system has to be able to take input from a weight in kg, filtering off the weight of a ordinary canteen plate. This is for the buffet food, where the price is calculated by weight. This would be a functional requirement
- Requirement: The payment system has to support credit cards and be able to interface with credit-card emulating software. This would be a non-functional requirement.