Seminar 1

Object-Oriented Design, IV 1350

Lucas Larsson
lulars@kth.se

31-03-2021

Contents

1 Introduction

Seminar 1 has basically two tasks that need to be completed, to get a passing grade.
Task 1: Create a Domain-Model in UML.
Task 2: Create a System sequence diagram also in UML.

The above-mentioned tasks are the core of the Seminar, through doing these tasks the participants are expected to get aquatinted with UML modeling tools, and practice creating domain model and system sequence diagram which will be referred to as “DM” and “SSD” respectively.

Collaborations:

Dennis Hadzialic

Other students during “*Handlednings Pass*”.
Other students in Discord Chanel.

2 Method

The main method used throughout the seminar tasks are inspired by instructions from the book (A First Course in Object Oriented Development, Leif Lindbäck), specifically from the chapter 4.2 Domain Model and 4.3 System Sequence Diagram.

The first step was to identify necessary classes, and this was achieved through reading and analyzing the seminar instructions that are supposed to be adapted in the domain model, more simply phrased, through identifying all the nouns and writing them down as potential classes, this step is also known as "Noun Identification".

Step two was to look for more class candidates using a different method "category list", this step can be described as complementing to step one, it's used to find more potential classes through inspiration as in a category list, it builds on thinking of classes that could possibly be needed but are not mentioned as nouns in the Requirements specification.

After acquiring potential classes through step 1 and 2, step 3 and 4 is to choose which class candidates to keep and eliminate unnecessary classes, and then which classes fits better as attributes. Steps 3 and 4 are achieved through discussing with other classmates and following the guidelines provided by the teacher in the form of lectures and examples in the book.

Step 5 concludes the Analysis Exercise, the last step is clarifying how the different classes are connected and it's achieved through adding associations that shows the purpose of the acquired classes and how they are connected, this step is also implemented through extensive discussions regarding appropriate associations, the association's name and between which classes.

3 Results

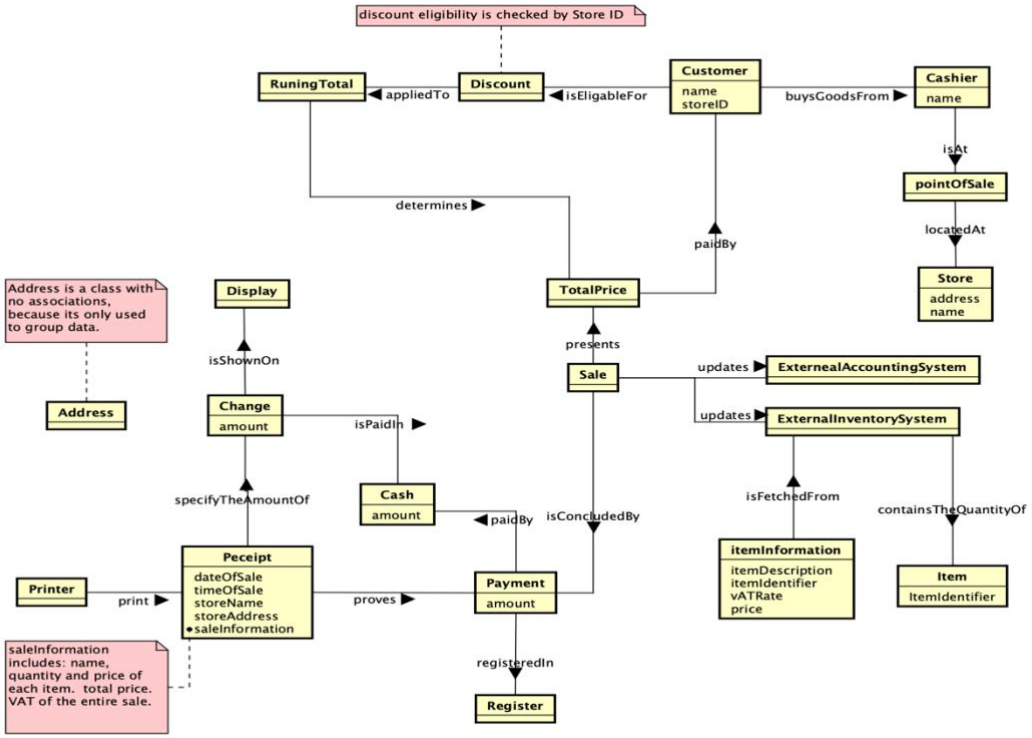

Figure 3.1: A Domain Model Diagram.

Figure 3.1 is one of the potential domain models that can be created according to the seminar requirement specifications. That is, different versions of the model can be created according to the same requirements depending on how it is interpreted by the designer, the main difference being the associations as well as the classes and its content.

The DM describes the workflow of a PointOfSale kind of scenario where a person (customer) is purchasing goods from a retail/grocery store using a UML diagram in the purpose of clarifying the whole process.

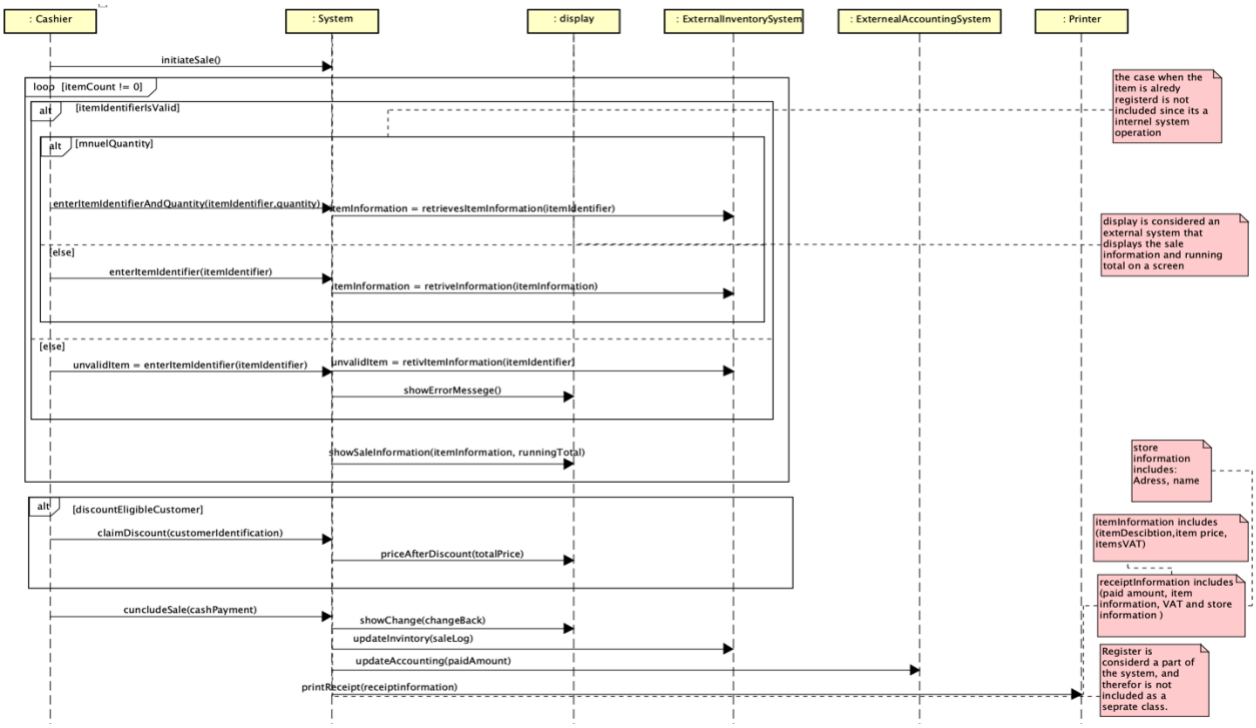

Figure 3.2: A System sequence diagram.

Figure 3.2 is also one of the potential SSDs that could be designed depending on the requirement instructions, it shows the systems different method calls to the different classes with the parameters depending on the function.

The following SSD is the next step after designing the DM to clarify the process, unlike the DM the SSD has a function of time, an example is how the method (initiateSale) is called first to start the process and the (printReceipt) is called last, in addition SSDs describe only the System operations and interaction with users (one user in this case Cashier) and other external systems.

4 Discussion

- Typical mistakes are to create a “programmatic DM” or “naïve DM”. Is the DM either of those?
    - According to me, the results presented are neither programmatic nor Naïve, that is because I got to redo the whole DM, after I received feedback from the responsible teacher regarding the problems that I needed to fix, and some more tips about how to recognize and later prevent such a problem.
- Are there \spider-in-the-web" classes?
    - There are no such classes, most classes have between 1-3 associations, the class with most associations is “Payment” with 4 associations, and this class is not considered a problem since it only represents essential associations without making the DM unclear or hard to read.
- Can you understand the DM? Is it a correct description of the problem domain?
    - Yes, I can, the DM describes the workflow of how a customer buys goods from a cashier with the extend of details such as the payment way, the different item identification etc...
- Does the DM have a reasonable number of classes? Are important classes missing?
    - Yes, the DM contains 20 classes. No, all necessary classes to complete the different operations are included such as discount sale information and receipt.
- Are there irrelevant classes?
    - There are no irrelevant classes. But this is a matter that is up to debate, depending on how others can interpret the instructions.
- Do you agree with the choices between class and attribute?
    - This matter was difficult to decided, a lot of the classes can work as attributes and vice versa, figure 3.1 is the final version and I agree with it, one attribute that I wanted to have as a class is ItemIdentifier, but the teacher gave a compelling argument against why it should be an attribute, and I agree.
- Is there a reasonable number of associations? Do you understand them? Are there classes missing associations? If so, is that a mistake?
    - There’s a reasonable number of associations, almost all classes have associations, all of them are easy to understand since all the associations are named.
    - One class that doesn’t have any associations is the class Address since making it associated with other classes will only serve to complicate the DM, which is exactly the opposite of what an association is meant to do. It is noted in the DM that it’s only used to store and group data that is used in other classes. Another class that can have more associations is the class Display, but no more associations are added with the same motive mentioned above, more associations will only serve to complicate the DM and make it difficult to understand, it is rather obvious that Display shows all the sale information such as the item information, running total and total price after eventual discounts.
- Are naming conventions followed in DM and SSD?
  Yes, they are followed class names always start with Big letters with no space and methods/attributes start with small letters and no space as well, ie camel case.
- Is UML used correctly in DM and SSD?
    - Yes, it is, the DM to clarify the business model and the SSD to show the program interaction and operations.
- Are the correct objects included in the SSD?
    - All the necessary objects are included.
- Are the correct operations and return values included in the SSD?
    - All the correct operations are included with return values in the cases where return values are needed.

- Is the method and result explained in the report? Is there a discussion? Is the discussion relevant?
    - There is not a discussion in the sense of discussion of only one topic, I do not think that it is supposed to be either. The method part of the report explains the method used to achieve the results, and the result part is just a brief explanation of what are the results, the discussion part is rather an answer to the questions provided by the teacher, some answers have a more comprehensive answer than others depending on the relevant information chosen by me, the author.