Seminar 1

Object-Oriented Design, IV1350

Jonathan Nilsson Cullgert jonnc@kth.se

24 Mars 2025

Project Members:
[Jonathan Nilsson Cullgert, jonnc@kth.se]
[Joel Friis, joefri@kth.se]
[Malte Berg, malte@malte.nu]

Declaration:

By submitting this assignment, it is hereby declared that all group members listed above have contributed to the solution. It is also declared that all project members fully understand all parts of the final solution and can explain it upon request.

It is furthermore declared that no part of the solution has been copied from any other source (except for resources presented in the Canvas page for the course IV1350), and that no part of the solution has been provided by someone not listed as a project member above.

1 Introduction

In this assignment a domain model and a system sequence diagram is meant to be modeled after a retail store that has a basic and alternative flow for how a transaction occurs in the store.

The process of doing a sale in the store is as follows. The Cashier starts the sale when a customer arrives. The cashier scans the product after which the system retrieves the VAT, price and item description from an external inventory system. This is repeated for all items that the customer has. If the id the for an item is invalid, the cashier is notified and if multiples of the same item is scanned, they are added as a grouped together quantity.

The cashier will then ask the customer if they are done when all items have been scanned. The customer can answer yes or no but in this task the answer will always be yes. The sale is then ended by the cashier. In the alternative flow the customer can

say that they are eligible for a discount. The customer will then give the cashier their customer identification which the cashier will register in the system. The system will contact a discount database where the customer will be given a discount if:

- The customer identification is connected to a discount.
- The customer buys items that are connected to a discount.
- If the sale is above a certain amount which triggers a discount.

When or if discounts have been applied, the register will display the total to pay aswell as the VAT. The customer pays a certain amount, the cashier enters this into the register. The system will update the amount of money in the register, send an update to an external accounting system and tell the cashier how much change to give to the customer. The change is given to the customer together with a receipt that contains: the date and time of the sale, name quantity and price of each item, total price of the sale, VAT for the sale and amount paid & change.

2 Method

When solving the two tasks in this assignment astah was used to create both the domain model and the system sequence diagram.

To create the domain model the steps followed were, noun identification, using the category list, finding attributes in classes and from classes, and finally setting up associations between classes.

Noun identification was done by first reading through the text and finding all the nouns in the basic flow of the requirements. We opted to first design the basic flow and then adding the necessary classes, attributes and associations from the alternative flow. After finding all the nouns in the basic flow we used the category list found in the book. When trying to identify classes from the category list discussion was held between group members and this was done to try and find classes that might not be inherently obvious from the requirements.

After the identification all the classes that had been named were gone through one by one and they were discussed whether they should be kept or not and if they might be a better fit as an attribute for another class. The classes got their attributes either from other classes fitting as attributes for them or because it was stated in the description of the requirements and attributes could be found from that. The associations between different classes was about trying to figure out which classes should be associated with other classes and what their association should be. Even in this phase some classes were removed since it became obvious that they served no purpose since they could not be connected to anything else.

The alternative flow was then implemented by following the same steps as for the basic flow, but unnecessary copies were skipped. When the alternative flow was considered some associations were also changed to compensate that flow.

The system sequence diagram was implemented systematically checking for when the cashier is interacting with the system or when the system is interacting with other external systems. The cashier scans items as long as they exist, and the system can flag whether or not they are found in the system. The system will also be in contact with the external inventory system, discount database and a display. Loops and if-statements were found by looking through both basic and alternative flow as well as checking in the requirements when contact is done between the system and the other external systems and the cashier.

3 Result

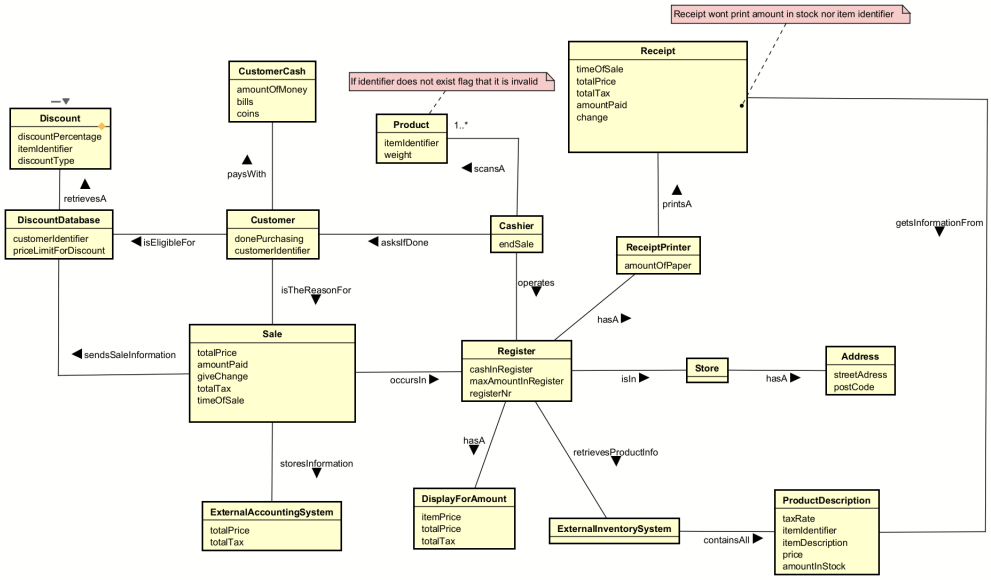

Figure 1: The Domain model that has been created from the requirements of both the basic and alternative flow.

In the domain model seen in 1 the main classes the model is centered around is the sale and register classes. The cashier and customer is associated with these classes and they are the reason for the sale taking place. The customer starts the sale which is done in the register which the cashier operates. The register get product information from an external system which it uses to send information to a display that the customer can look at for the current price and VAT. The register also prints a receipt that is given to the customer, the receipt contains various information, but all the information about the price, name and VAT for individual items is gotten from the same external inventory system.

Both the customer and the sale itself will flag the discount database to check whether a

sale is valid for a discount. The sale will do the check since it contains the sale information (amount of products, and which products), whereas the customer has a personal identifier which could make them eligible for a discount. The database then retrieves a discount from the discount class by sending and using the relevant information.

This domain model is modeled with the store and address classes since we thought it was relevant that information could be found, especially if the store that a transaction is taking place in is part of a larger chain with many locations.

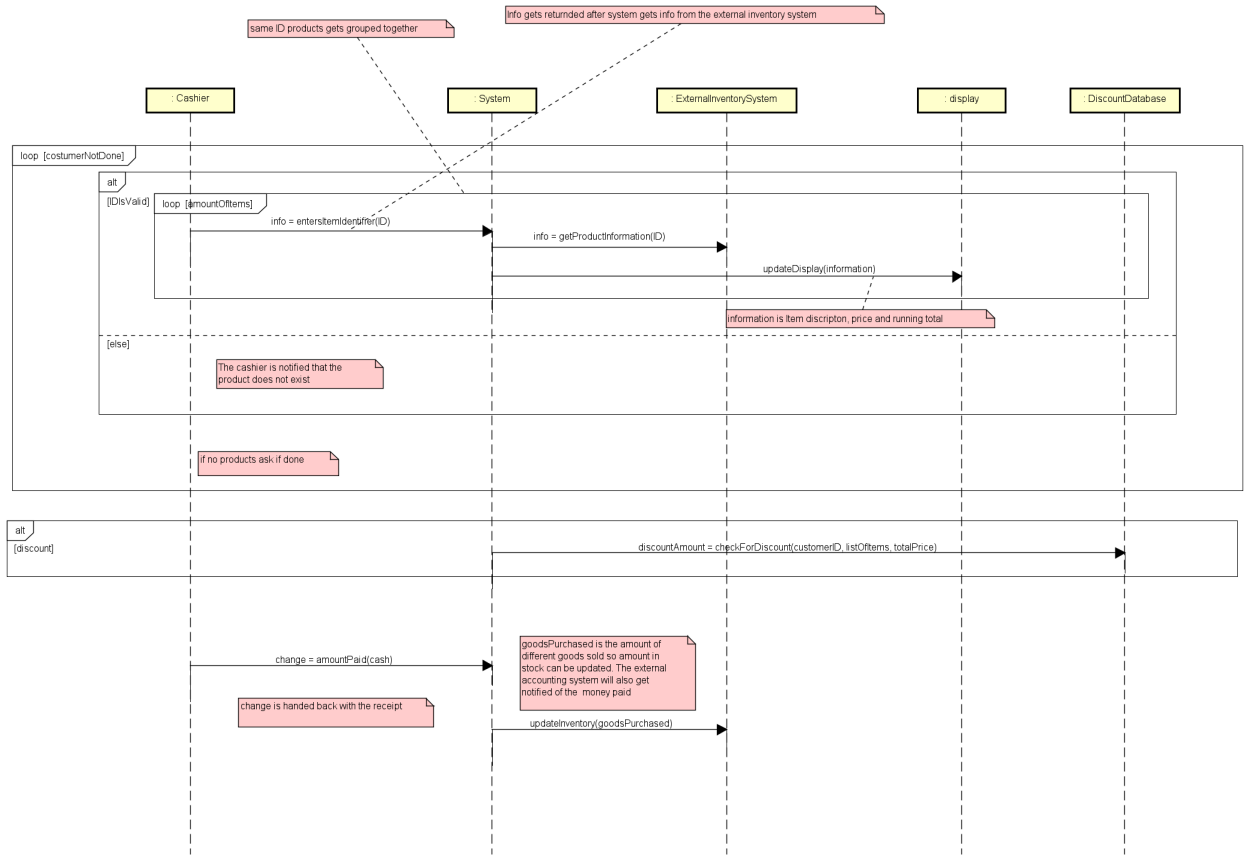

Figure 2: The system sequence diagram that has been modeled after the requirements from the basic and alternative flow as well as our interpretation of the systems.

The system sequence diagram seen in 2 was modeled using 5 ”systems”, the cashier, the system, the external inventory system, the display and the discount database. The interesting interactions are when the main system is interacting to the left of it (cashier), and to the right (external inventory system, display and discount database).

The first thing that happens is that a sale occurs and we enter a loop that continues for as long as there are items to be scanned by the cashier. The cashier then scans an item, and a new check is done to see if the item identifier of the product is valid or

not. If it is valid another loop is entered were multiple products of the same type are scanned in succession. The item id is then sent to the external inventory system where the important information is sent back to the system and the cashier, the system will then also change the display to show the relevant price and name.

If the id was not valid a flag will be sent to the cashier to notify them so that they can take the next necessary action. When all items are scanned, there is a check for discounts, and the system then sends the sale information and customer id to the discount database and get returned a potential discount.

After that the cashier enters the amount paid into the register, so that the change handed back to the customer can be calculated. The system will also send information to the external accounting and inventory system to record the sale and update the inventory.

4 Discussion

The difficult choices when designing the domain model was if we were supposed to keep the external systems (discount, accounting and inventory) and how we would implement them if they were kept. We decided to keep the systems and implement sub classes inside of those systems that contained the information that we are interested in. The reason for doing it this way is that we don't know or care about how those systems are designed, we just know what information we need from them. That way we have classes that are tied to important information for us, as an example a product has an item identifier which is connected to the class product description which has the same id as an attribute. That way we get the relevant information from the database in our design without digging deep into how the inventory system is set up. The same could be done for the accounting system but since we do not retrieve information from it, we decided not to do that.

We tried to avoid getting a programmatic or spider web domain model but since sale and register are so central in the entire model we almost ended up with a spider web. I do believe that it is okay though since we branch out from multiple classes and that there are layers in associations, but it could possibly have been possible to unfold some associations a bit to get a nicer model.

In product we decided to use a multiple identifier and the reason for that was that we wanted to make it super clear that the cashier could scan in multiple items of the same type. It is also true for some associations we have, they might be unnecessary but since we were not one hundred percent certain we kept them in, such as cashier asking the customer if done, and when the customer pays.

The system sequence diagram we decided to keep the display as a separate system that our register system is interacting with. We kept it since we saw it as a part of the register that we might be changing and updating and therefore it is relevant in our SSD to keep it in there.

When writing this report there are some thing I see with our SSD that could have

been changed to improve it. The first thing is that the way it is modeled right now, info would be retrieved from the inventory system multiple times in a row for products with the same ID, this could be changed so that the calls are done outside the loop.

Other than that I can not find any other glaring flaws with our solution. We did the loop the way we did so that multiples of the same items in a row could be scanned once, and an amount entered but with the way it is designed it is not working correctly.

Our SSD flags when something is wrong with the ID, and all the relevant information gets passed to the cashier, register system and the display. Discounts are handled and updates to accounting and inventory are done when the sale is done as the customer receives their change and receipt. Internal changes such as how much money is in the registered is not showed in the SSD since that is just in the nature of this diagram.

References

- The Task description in canvas
- Lindbäck, L. (2024) 'Chapter 4 Analysis', in A First Course in Object-Oriented Development A Hands-On Approach, pp. 20–40. Available at: `https://e.pcloud.link/publink/show?code=XZa6eKZI6J9qagkziJ7TiyNrMf9qQhMDrC7` (Accessed: 18 March 2025).
- Lindbäck, L. (2025) ' Föreläsning 3, Domänmodell, del 1 ', 18 March.
- Lindbäck, L. (2025) ' Föreläsning 3, domänmodell, del 2 ', 18 March.
- Lindbäck, L. (2025) ' SE Föreläsning 4, systemsekvensdiagram ', 21 March.