IMI WiSe 2019 / Informatik 2

Students : Muhammad Safarov
Alexej Bormatenkow
Pavel Tsvyatkov

Email : s0570690@htw-berlin.de
s0570108@htw-berlin.de
s0559632@htw-berlin.de

Date: 14.10.2019

Instructor: Prof. Dr. Debora Weber Wulff

Laboratory Report

Exercise 0: Dealing with objects.

Lab-plan

First of all, after a short introduction, the professor explained the main idea of the exercise and what we are required to do. The whole exercise consist of 4 tasks. We have decided not to pay attention at how much time we are going to spent solving each task for a simple reason that we had to think carefully to not to miss anything important.

Question 1 & 2:

“*The cinema booking system should store seat bookings for multiple theaters. Each theater has seats arranged in rows. There can be a different number of seats in every row. Customers can reserve seats, and are given a row number and a seat number. They may request bookings of several adjoining seats. Each booking is for a particular show (that is, the screening of a given movie at a certain time). Shows are at an assigned date, time, and price, and are scheduled in a theater where they are screened. The system stores the customer's name and telephone number. The customer is told what the booking will cost when the tickets are picked up.The first step is to discover some candidate classes and methods. Use the Booch method to determine candidate classes and methods, and write them down.*

Make CRC cards for each of your candidate classes. Only put down the class names for now.”

First we discussed on how a typical process of getting seat reservations would look like and isolated all the key roles in it.[1] We decided on having a class for the *customer* which we thought might be necessary for storing data like name and phone number for storing the reservation in the cinemas database later.
Our second class is the *employee*: the employee is like an interface between the customer class and the other classes of which our booking system consists. The employee is getting information from the customer about the movie she wants to see, her name and phone number, preferred date and time.
Our actual booking system consists of the following classes: *cinema, ticket, seat, movie and database.*
The cinema class represents a particular theatre and contains information about screened movies, timetables and available seats. We thought of it a little further: what should we do if the theatre hasn't got free seats for a particular show? What if we are dealing with a cinema franchise which has multiple theatres all over the place and the employee could suggest another theatre, in a different location for the desired movie. Hence we thought of a location field which which gives information about the whereabouts of the particular *cinema.*
With the *seat* class we describe a particular seat in a movie theatre with fields for seat number, row and possible extra fees. In the same manner our movie class consists of fields for movie name*,* runtime (for possible extra fees) and collaborates with the cinema class.
The *ticket* class is where all the information about the particular movie, runtime, cinema, location and cost after viable discounts for the customer is gathered.
After the ticket class is finished and all fields are filled with required information, the employee is able to check whether the customer wants to buy this particular ticket or not. If the customer is satisfied, our ticket class becomes the data model for the *database* class. In an abstract way the database class is our interface to a database in the backend of our booking system. The database is accessible by the employee to retrieve former entries of booked tickets.
After determining the needed classes we scribbled a flowchart of relationships between them, to get an overall view of our booking system.[2]
With the help of our flowchart we determined first collaborations between our classes and decided upon additional fields in our classes. Fields seemed necessary for needed data which couldn't be represented as a collaboration with another class.
Fields are marked with an asterisk * and are written in lowercase letters in the right hand column of our cards.

[1] see Appendix I – CRC cards of our classes
[2] see Appendix II – Relationship flowchart of our classes

Question 3 : The first scenario is a simple reservation:

"Jane Doe calls the cinema and want to make a reservation for two seats to watch Tomb Raider at 8 pm. The cinema employee starts using the booking system to find and reserve a seat. Using the CRC cards, play through the scenario."

Before we have started solving the 3rd Assignment we have decided to write down our answer "step by step".

First of all a customer asks the employee about *movie* and the *time* when its being screened. As requested, employee is looking up the *movie* "Tomb Raider". It's important to mention that the *movie* is connected to specific *cinemas* and *locations.*
In order to search for a movie, every *cinema* in our system contains a field **schedule** which consists of available movies in the specific theatre and their screening times.
After the request of the *employee* the system figures out at which cinema the movie "Tomb Raider" is being screened at 8 pm. The next step would be that the employee asks our customer which seats should be reserved at which of the available *cinemas* in certain *locations.* The available *cinemas* and *locations* are presented / suggested by the employee. The employee checks the availability of the seats in cinema "x" in location "y" and gives back the information that seats 13 and 14 from row 12 are available. All the gathered data about *movie, cinema, location* and *seats* is stored in the *ticket* class.
The employee presents the *ticket* to the customer and awaits a confirmation from the customer for buying this specific *ticket.* If the customers confirms the purchase, the employee asks for phone number and name. (For the further verification purposes). The phone number and customer's name are stored in the in the ticket class. Lastly, the employee saves the created ticket in the database of the booking system.

After we were done with describing the process, we went step by step through the text and filled in all the Collaborators and Responsibilities of the different classes. We decided that *Location* doesn't need to be a class anymore and instead we placed it as a *field* in the *Cinema* class.

Question 4 : Choose another scenario and play it through.

For this exercise we discussed that the process for every scenario is similar to Assignment 3, but with small changes to some details about the reservation. While doing this exercise, we didn't find it necessary to add or change anything on our CRC cards, we thought we already had everything we needed for our booking system to work correctly after doing the previous exercise.

1). James already has a booking and needs another two, adjoining seats.

In the first scenario, the process of booking has already been done once and then the customer wants two additional adjoining seats. The employee has to look up the database entry for the already reserved seats by James. Then he has to check if there are available adjoining seats for this particular movie and cinema. Depending on availability of seats the reservation is then either made or cancelled.

2). Mary wants to book 4 seats together, but there are not 4 adjoining seats in one row available.

For the second scenario, we thought that the customer first says what reservation they would like to have. Then the employee has to look up the desired movie and look for cinemas with 4 adjoining seats. After the check, the employee communicates with the customer and says that there aren't 4 adjoining seats. The customer then has the right to cancel the order or change the movie and look for other opportunities.

3). Joseph wants to book, but there are no seats available.

In the third scenario the customer wants to make a reservation for a certain movie. However, after the employee looks up the database for the particular movie, he says that there are no seats available and the reservation is then cancelled.

4). Anna has a booking she wants to cancel.

All group members agreed that in the fourth scenario the process from exercise 3 has already been done, but the customer now needs to call for cancellation. The employer checks the database for the booked tickets by Anna. He confirms with the customer and cancels the reservation. The further process depends on the refund policy of the cinema.

Personal reflection of the process

Pavel: This exercise involved a lot of thinking, sharing ideas and listening to other group members. It was nice that while making the CRC cards everyone shared their thoughts and we were coming up with different ideas. From this assignment I learned how to think about the process as a whole first and then figuring out the small details about what each class is responsible for. I think the CRC cards really help the process of understanding how the system works, because we can clearly see which classes collaborate with one another and move them around while explaining our thoughts and ideas.

Muhammad: As I expected it was a very interesting exercise because I really enjoy thinking not just like programmer but also as a customer or the employee. Sure it was not as easy as it seems because we have used a LOT of paper figuring out which classes we need and what is a complete nonsense, but it gave me an idea how important it is to know the exact responsibilities, methods etc. of every class. I think it's one of the most important skills in object oriented programming.

Alexej: During the exercise I encountered several topics my group members and I had to tackle. At first establishing a concise communication about the system we want to design and all the parts in it. Does every member of the group understand what is meant with a certain CRC card or the class which is described by it? Some time was needed to getting the terms right and the overall practical approach of thinking and acting on this abstract level. Another topic was working with the analogue medium of index cards and especially the CRC method. I had the impression that we didn’t grasp the simplicity of the CRC model at first and thought a lot in terms of an actual implementation in an OOP language like Java. That’s why we established the *“*fields”* on our CRC Cards or thought a lot about matters of data visibility / encapsulation and how to take note of it on the cards. We lost a little bit of time and energy which we could have used on the real focus of the CRC model which is a fast prototyping tool for class interdependencies and relationships. Furthermore I caught myself thinking a lot in ontological terms about the classes and their place in the domain of our booking system – what is *really* an entity and what is just an attribute or feature of another entity?
Learnings: read the manual, twice. Come to terms with your teammates as soon as possible about what you have *really* understood. Don’t overthink it, it’s a booking system for tickets and not a philosophical problem.

Appendix I: CRC Cards of our classes

Ticket	Customer Movie Cinema Seat Database
Holds Information about Movie, Cinema, Seat and Customer *Models the Database entry type*	

Customer	Employee Ticket *name *phonenr
Chooses movie and time. *Gives personal information (name,phonenr)* *Gives information "seats"* *Confirms yes / no questions from Employee*	

Seat	Cinema *row *seatNum
Gives back information about available seats in specified rows to the Cinema class	

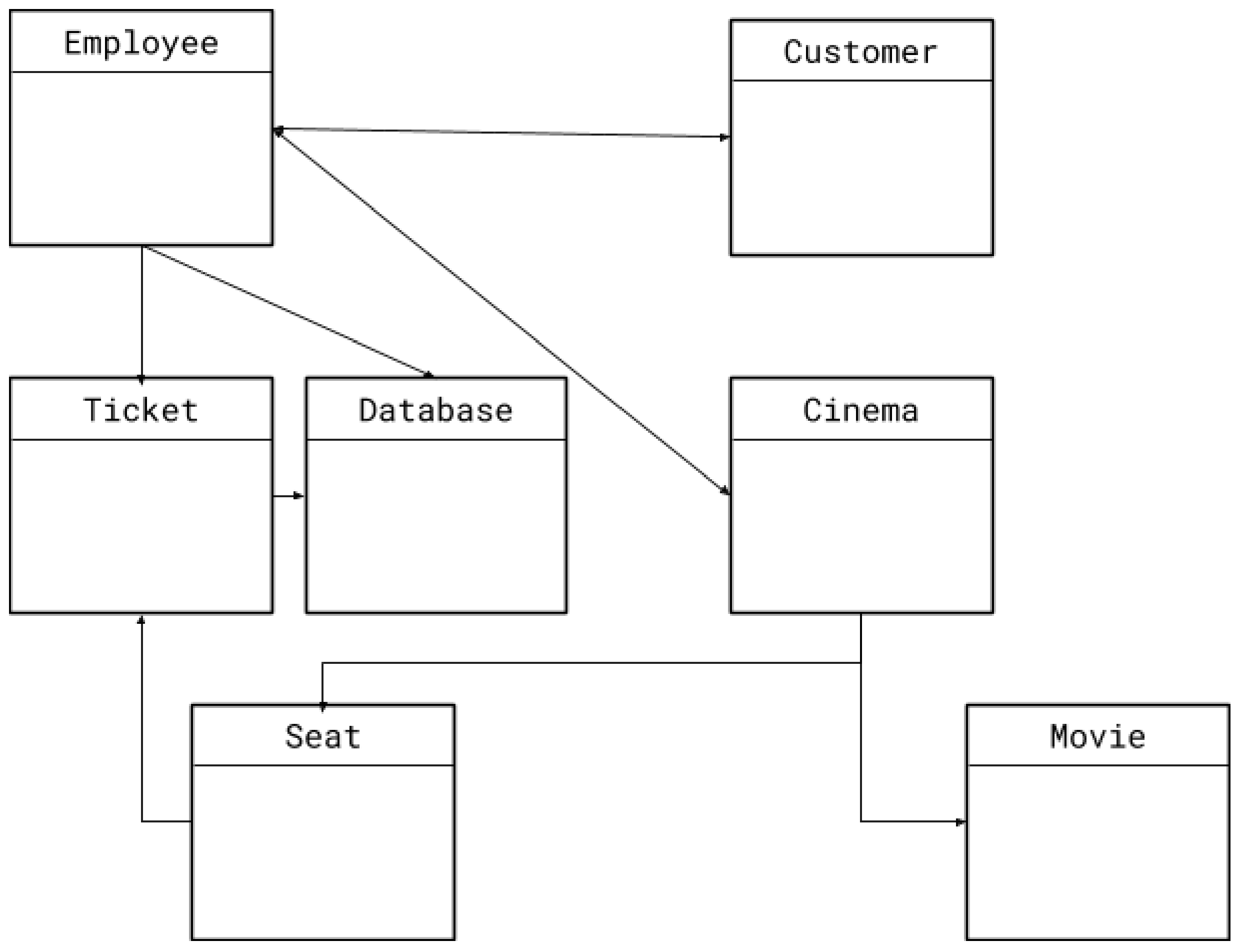
Cinema	Seat Movie *schedule *location
Checks up available seats in Seat class *Presenting a schedule of available movies* *Being located somewhere*	

Employee	Customer Movie Database Cinema
Gathers information and confirms information *Reads, Updates and Deletes reservations in the Database* *Cancels the reservation process*	

Database	Employee Ticket
Provides entries to Employee *Stores confirmed tickets from the Employee*	

Movie	Employee Cinema
Looks up its availability and schedule in cinema class	

Appendix II: Relationship flowchart of our classes



Apendix III: PreLab

P1. How do you obtain tickets to go see a movie? Write down the steps that you take, in order.

1. I go online on my laptop and open an official website of a local cinema.
2. I'm searching for a movie that I would like to watch.
3. I'm clicking a button “Order tickets”
4. I'm choosing a seat where I would like to sit.
5. I'm reserving a seat and going through payment procedure.
6. I'm giving all the required information about me and my credit card to complete the payment procedure.
7. Im receiving my Online tickets on my Email.

P2. If you have tickets and have to cancel, what do you have to do? Write down the steps, in order.

1. The information about canceling a ticket is mostly written somewhere on the ticket, but in case if I didnt find anything I go online on the official website of the cinema and look for a support hotline number.
2. After reaching a support team worker I will claim that I want to cancel my ticket.
3. I will give all the required information to confirm my identity.
4. I will wait for my money to be transferred back on my bank account.