Cafe_AJOU

Project Elaboration Phase Iteration 1 Report

by TEAM. 학교 가고 싶다

김동현	amijo998@ajou.ac.kr
나용성	nayong2017@ajou.ac.kr
박민지	alswlkku@ajou.ac.kr
신채연	cheayeon33@ajou.ac.kr
전상범	beom115@ajou.ac.kr

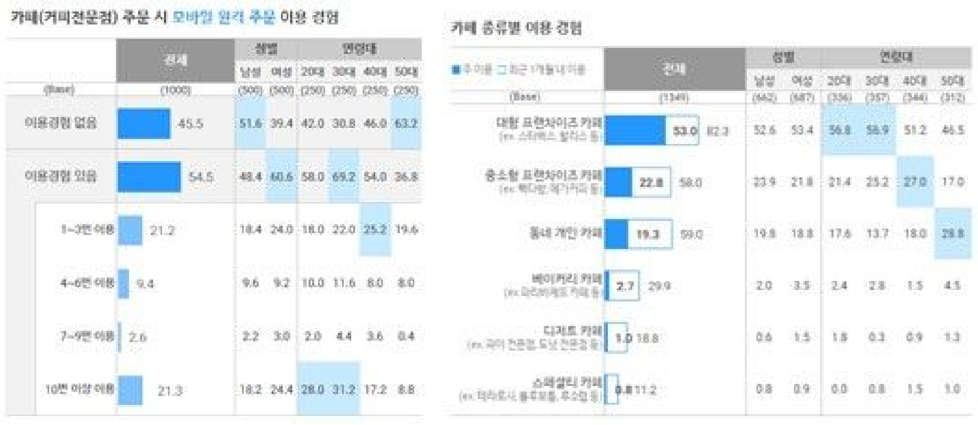
Table of Contents

<List of Tables>

<List of Figures>

1. Vision

1.1. Introduction



카페(커피전문점) 주문 시 모바일 원격 주문 이용 경험

	전체		성별		연령대			
			남성	여성	20대	30대	40대	50대
(Base)	(1000)		(500)	(500)	(250)	(250)	(250)	(250)
이용경험 없음	45.5		51.6	39.4	42.0	30.8	46.0	63.2
이용경험 있음	54.5		48.4	60.6	58.0	69.2	54.0	36.8
1~3번 이용	21.2		18.4	24.0	18.0	22.0	25.2	19.6
4~6번 이용	9.4		9.6	9.2	10.0	11.6	8.0	8.0
7~9번 이용	2.6		2.2	3.0	2.0	4.4	3.6	0.4
10번 이상 이용	21.3		18.2	24.4	28.0	31.2	17.2	8.8

카페 종류별 이용 경험

■주 이용 □최근 1개월내 이용

	전체		성별		연령대			
	■	□	남성	여성	20대	30대	40대	50대
(Base)	(1349)		(662)	(687)	(336)	(357)	(344)	(312)
대형 프랜차이즈 카페 (ex. 스타벅스, 할리스 등)	53.0	82.3	52.6	53.4	56.8	56.9	51.2	46.5
중소형 프랜차이즈 카페 (ex. 빽다방, 메가커피 등)	22.8	58.0	23.9	21.8	21.4	25.2	27.0	17.0
동네 개인 카페	19.3	59.0	19.8	18.8	17.6	13.7	18.0	28.8
베이커리 카페 (ex. 파리바게뜨 카페 등)	2.7	29.9	2.0	3.5	2.4	2.8	1.5	4.5
디저트 카페 (ex. [illegible] 전문점, 도넛 전문점 등)	1.0	18.8	0.6	1.5	1.8	0.3	0.9	1.3
스페셜티 카페 (ex. 테라로사, 블루보틀, [illegible] 등)	0.8	11.2	0.8	0.9	0.0	0.8	1.5	1.0

Figure 1 Cafe Usage Trend Report 2020 by Open survey

According to the “Cafe Usage Trend Report 2020” released by Open Survey, Those in their 20s~30s had a high experience rate with 76.4% of non-face-to-face cafe orders. And People in their 20s tried to use the non-face-to-face ordering method when they wanted to think enough about menu choices by referring to detailed menu images. Currently, many Apps are in operation on the school side to provide convenience to students. However, there is a time to wait for orders and inconvenience of using the cafe between ordinary users and students studying in the cafe has not been resolved. Therefore, we would like to implement the ‘non-face-to-face cafe order’ system mentioned above. This system helps Customer reduce unnecessary time by allowing them to order drinks at any time they want, and also reduces the inconvenience of having to find another cafe because there are no seats available through seat reservation. It also helps store convenience by implementing functions that enable efficient management of the store not only from the customer but also from the store side.

1.2. Positioning

1.2.1. Business Opportunity

Customer can reduce unnecessary time by using a system that sets the time to pick up rather than a system requires them to make orders on their own feet. In addition, it can provide convenience to Customer through a system that allows them to reserve seats together while ordering drinks by supplementing the problem of having to visit person every time to see if there are any seats available or not. This could lead to increased inflow into Customer's store and to increased store profits. Apart from these financial benefits, we thought that providing convenience to the store while providing statics on customer sales using the app and the ability to easily add/delete menu without having to go through complicated procedures could lead to improved quality of service for the customer. Therefore, it was determined that if good results could be achieved in an efficient way, better results could be provided for both sides.

1.2.2. Problem Statement

Due to the nature of the program, which aims to reduce waiting times, the current amount of orders in the cafe(offline orders, not apps) has a significant impact. Without information about these latencies, programs can't operate efficiently. Even if minimized, offline orders would still exist and as much as possible can be linked to the Pos system to provide the expected waiting number of person. But, we thought it would be difficult to provide accurate waiting time considering the development environment and the given development time.

1.2.3. Product Position Statement

Similar programs exist outside the school at the other franchise cafes (like STARBUCKS). However, we have a competitive edge in the 'use of school members' and 'shorT distances' than other cafes outside of school. This competitiveness leads to the need for this program. Store can also reduce the burden of competition from other cafes.

1.3. Stakeholder Description

1.3.1. Stakeholder (Non-User) Summary

- **System Manager :** To maintain the System. System should know errors quickly. so Improve usability by correcting errors in a short time.
- **Pos System:** For overall off-lin sales, menu management and order management.
- **In-app Payment System:** To pay when the reservation is completed.

1.3.2. User Summary

- **Customer** : cafe users can check the status of their seat reservation with the system before visiting the site to reduce the hassle of having to check their seats by visiting the site in person. In addition, customers can reserve the cafe drinks in advance so they can receive their drinks as soon as they arrive at the cafe.

- **Manager** : Through this program, Manager can expect to improve customer convenience and it leads to increase the profits. The menu management function allows managers to quickly add or delete menu in store, which also increases convenience.

1.4. Product Overview

1.4.1 Product Perspective

It is installed within the application. It will be used by the members of Ajou university who want to order a drink and also reserve a seat at the cafe in advance. I will provide services to users, and collaborate with other systems, as indicated in the form shown below.

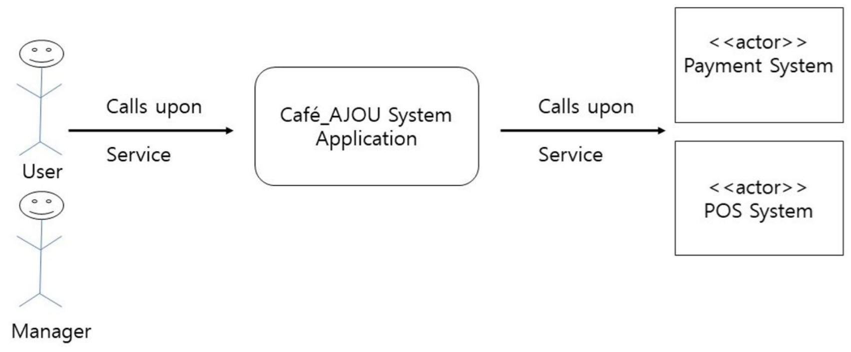


Figure 2 Product Perspective Scheme

1.4.2 Summary of benefits

Supporting Feature	**Stakeholder benefit**
By adding or modifying or deleting menus, the system updates the changes also applied to the POS system.	Manager can manage menu information.
By entering the order/reservation information, the system guarantees a seat thatand pre-order a drink.	User can order a drink and also reserve a seat in advance at the cafe.
The system collects customer purchase data and produces statistics on monthly/weekly basis that the manager wants for each beverage.	Manager can utilize customer data to generate increased revenue and manage inventory effectively.

Table 1 Summary of benefits

1.5. Summary of System Features

- Manager who already has authorization can manage menus including add, delete and modify menus.

- Members of Ajou university can order a drink any time they want and know how many seats are available at the cafe.

- User can pay through In-app payment if the user wants to pay in advance, not on-site payment.

2. Requirement

2.1. use case Diagram

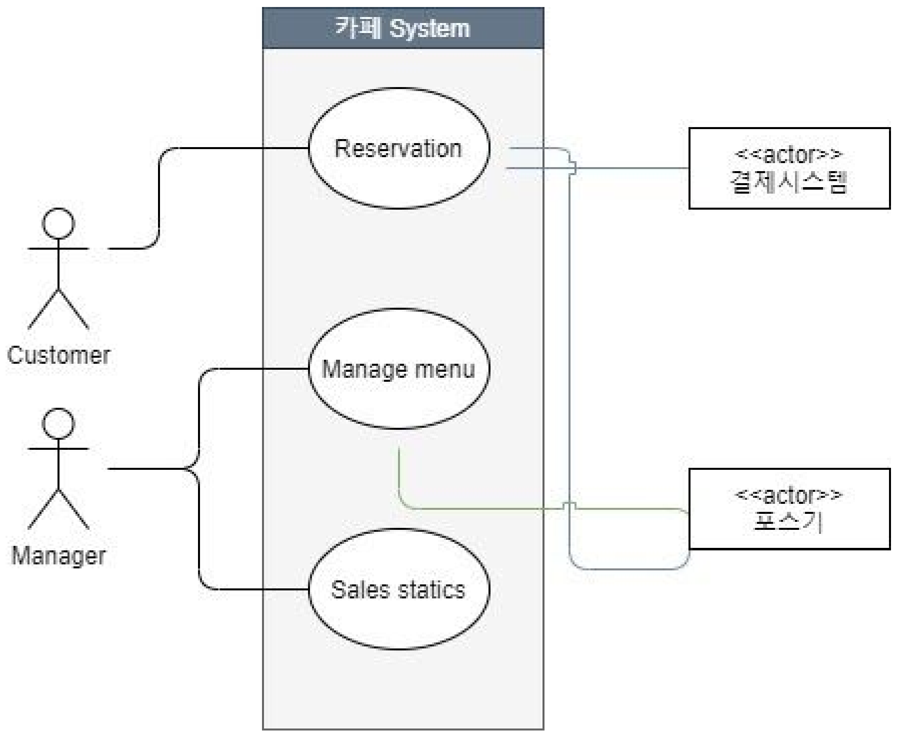


Figure 3 Use Case Diagram

2.2. Use Case Text

1. Reservation

2. Menu management

3. Sales Statistics

Table 2 Use Case List

2.2.1 Functional Requirement

2.2.1.1 Reservation

First user selects one of three optional cafes at Ajou university and also selects takeaway or eat-in. At the next step, the user can designate time when to visit the cafe for the beverage and he can reserve a seat at the cafe with limited time. Maximum 3 times he can extend the seat with the time limit, 1h 30m(this is the course time). If the user doesn't show up at the cafe, he can't reserve a seat for 3 days as a penalty. If he pre-order a drink and a seat but changes may occur so he can modify or cancel his reservation before 30m of the reservation.

2.2.1.2 Menu Management

We need a 'Menu Management System' to make users know the menu information/list of the cafe. Using this system, manager can add new menus, delete discontinued products and modify the information of existing menus and state of menus for indicating out of order for that day.

2.2.1.3 Sales Statics

Based on sales of customer purchase data through the application, manager can see the statistics of sales so it helps for manager to manage inventory effectively like placing a plenty of orders for the stock of popular menus. Also it helps him to generate increased revenue.

2.2.2 Use Case Model

Use Case 1. Reservation

- Level : User-goal

- Primary Actor : Customer

- **Precondition**
  - Customer는 인증이 끝난 상태로 기능에 접근한다.
  - 메뉴에는 카테고리 별 음료 이름, 가격이 저장되어 있다.
  - 시스템에 좌석 별 예약된 시간이 저장되어 있다.

- Scenario Flow :

1. Customer가 예약하기 기능을 시작한다.
2. 이용하고자 하는 매장을 선택한다.
3. 카테고리 별 음료 이름, 가격을 보고 음료를 선택한다.

Customer repeats steps 3 until indicates done.

4. 좌석예약에 대한 여부를 선택한다.
5. 시스템이 모든 지정 예약석을 보여준다.
6. Customer가 이용하기를 원하는 시간을 입력한다.
7. 시스템은 사용자의 입력에 따라 이용 가능한 예약석을 표시한다.
8. Customer가 예약 가능한 좌석을 선택한다.
9. 계산 된 요금의 총액을 알려주고 결제를 요청한다.
10. 시스템이 판매를 완료하고 판매 및 결제 정보를 외부 회계 시스템으로 보낸다.
11. 시스템이 영수증과 주문 내역을 제시한다.
12. 시스템은 store number를 통해 주문 내역을 store에게 전송한다.
13. 시스템이 요청된 시간의 좌석 상태를 예약으로 바꿔주고, 음료 별 Sale에 대한 total 값을 변경시켜준다.

- Extension

1a. 예약수정을 선택한 경우

1. Customer가 예약 수정 기능을 사용한다.
2. 시스템이 사용자의 예약목록을 보여준다.
3. Customer가 수정할 예약을 선택한다.

    2a. 선택한 예약이 '매장에서 먹기'인 경우

    1. 시스템이 예약정보를 보여준다.
    2. Customer가 수정하고자 하는 정보를 선택한다.

        2a. 메뉴를 수정하고자 하는 경우

        1. Customer가 수정하고자 하는 메뉴를 선택한다.

            1a. Customer가 수량의 수정을 원할 경우

            1. Customer가 메뉴의 수량을 조정한다.
        2. 시스템이 선택가능한 메뉴목록과 가격을 보여준다.
        3. Customer가 원하는 다른 메뉴를 선택한다.

        2b. 이용시간, 좌석 변경을 원하는 경우

        1. 시스템이 예약 가능한 시간 현황을 보여준다.
        2. Customer가 원하는 시간대를 선택한다.
        3. 시스템이 선택한 시간대의 예약가능 좌석현황을 보여준다.
        4. Customer가 원하는 좌석을 선택한다.

    2b. 선택한 예약이 '테이크 아웃'인 경우

    1. 시스템이 예약정보를 보여준다.
    2. Customer가 수정하고자 하는 정보를 선택한다.

        2a. 메뉴를 수정하고자 하는 경우

        1. Customer가 수정하고자 하는 메뉴를 선택한다.

            1a. Customer가 수량의 수정을 원할 경우

1. Customer가 메뉴의 수량을 조정한다.
2. 시스템이 선택가능한 메뉴목록과 가격을 보여준다.
3. Customer가 원하는 메뉴를 선택한다.

2b. 수령시간을 수정하고자 하는 경우

1. Customer가 원하는 수령시간을 선택한다.

4. 시스템이 수정된 예약의 총액을 알려주고 재결제를 요청한다.
5. 시스템이 기존의 결제를 취소하고 재결제된 정보를 외부 회계 시스템으로 보낸다.
6. 시스템은 store number를 통해 예약 수정 내역을 store에게 전송한다.
7. 시스템에 수정된 예약내용이 저장된다.

1b. 예약취소를 선택한 경우

1. Customer가 예약 취소 기능을 사용한다.
2. 시스템이 사용자의 예약목록을 보여준다.
3. Customer가 삭제하고자 하는 예약을 선택한다.
4. 시스템에서 선택된 예약의 결제를 취소하고 취소 정보를 외부 회계 시스템이 보낸다.
5. 시스템은 store number를 통해 예약 취소 내역을 store에게 전송한다.
6. 시스템이 저장된 예약내역과 좌석 정보를 삭제한다.

1c. 좌석 예약연장을 선택한 경우

1. 사용자가 예약 연장 기능을 선택한다.
2. 시스템이 이용시간을 연장해주고, 연장된 시간을 좌석 현황에 반영한다.
3. 시스템이 변경된 예약 정보를 사용자에게 보여준다.

5a. 좌석예약을 하지 않는 경우

1. 사용자가 음료를 받고자 하는 시간을 선택한다.
2. 계산 된 요금의 총액을 알려주고 결제를 요청한다.
3. 시스템이 판매를 완료하고 판매 및 결제 정보를 외부 회계 시스템으로 보낸다.
4. 시스템이 영수증과 주문 내역을 제시한다.
5. 시스템은 store number를 통해 주문 내역을 store에게 전송한다.
6. 시스템이 예약내용을 저장한다.

Table 3 [UC1] Reservation(fully dressed format)

Use Case 2. Menu management

- Level : User-goal

- Primary Actor : Manager

- **Precondition**

  - 매니저는 로그인이 완료 된 상태다.
  - 시스템에 기본 카테고리 리스트가 사전에 저장되어 있다.
  - 시스템이 포스 시스템이 연동되어 있다.

- Scenario Flow :

1. Manager가 메뉴 추가기능을 시작한다.
2. 시스템이 선택가능한 카테고리 리스트를 보여준다.
3. Menu가 추가될 카테고리를 선택한다.
4. 추가 할 메뉴의 itemID, name, price, description을 입력한다.
5. 시스템이 입력된 메뉴의 정보를 보여준다.
6. Manager가 추가될 메뉴정보를 확인하고 등록한다.
7. 요청된 사항이 포스시스템에 등록된다.

- Extension

1a. modify Menu일경우

1. Manager가 메뉴 수정기능을 시작한다.
2. Manager는 수정할 메뉴의 ItemID을 입력한다.
3. 시스템이 선택가능한 카테고리 리스트를 보여준다.
4. Menu가 수정될 카테고리를 선택한다.
5. 메뉴의 itemID, name, price, description을 수정한다.
6. 시스템이 수정 메뉴의 정보를 보여준다.
7. Manager가 수정될 메뉴정보를 확인하고 등록한다.
8. 요청된 사항이 포스시스템에 등록된다.

1b. delete Menu일 경우

1. Manager가 메뉴 삭제기능을 시작한다.
2. 삭제할 메뉴의 ItemID을 입력한다.
3. 시스템이 삭제될 메뉴의 정보를 보여준다.
4. Manager가 수정될 메뉴정보를 확인하고 삭제요청을 한다.
5. 요청된 사항이 포스시스템에 등록된다.

1c. add Category일 경우

1. Manager가 카테고리 추가를 시작한다.
2. Manager가 추가할 카테고리의 categoryName과 categoryID를 입력한다.
3. 요청된 사항이 포스시스템에 등록된다.

Table 4 [UC2] Menu management(fully dressed format)

2.2.3 Non-Functional Requirement

Functionality

- **Logging and Error Handling**
  - Users(Customer and Manager) authentication is required.
  - Return to the previous page in case of an error.
- **Security**
  - Users without manager authentication cannot access Store Management.

Usability

- **Human Factor**

  Users should see it on their mobile phone screen because it is aimed at developing Apps.

  - Accurate icon shape and menu names should be used so that customers are not confused.
  - There are not only students in the school but also people who are old, So app help to adjust the size of the letters through the setting.

Reliability

- **Recoverability**
  - all data on orders and menu information should be backed up to the database once every 12 hours.

Performance

- The initial main page should be displayed within 10 seconds of access.
- The message about the user’s error should be sent within 3 seconds.

Supportability

- **Adaptability**

  Order information coming in real time via APP is forwarded to the store.

- **Configurability**

  Consider UX/UI to facilitate access to information.

3. Analysis Modeling

3.1. Domain Model Diagram (Use Case 1 & 2)

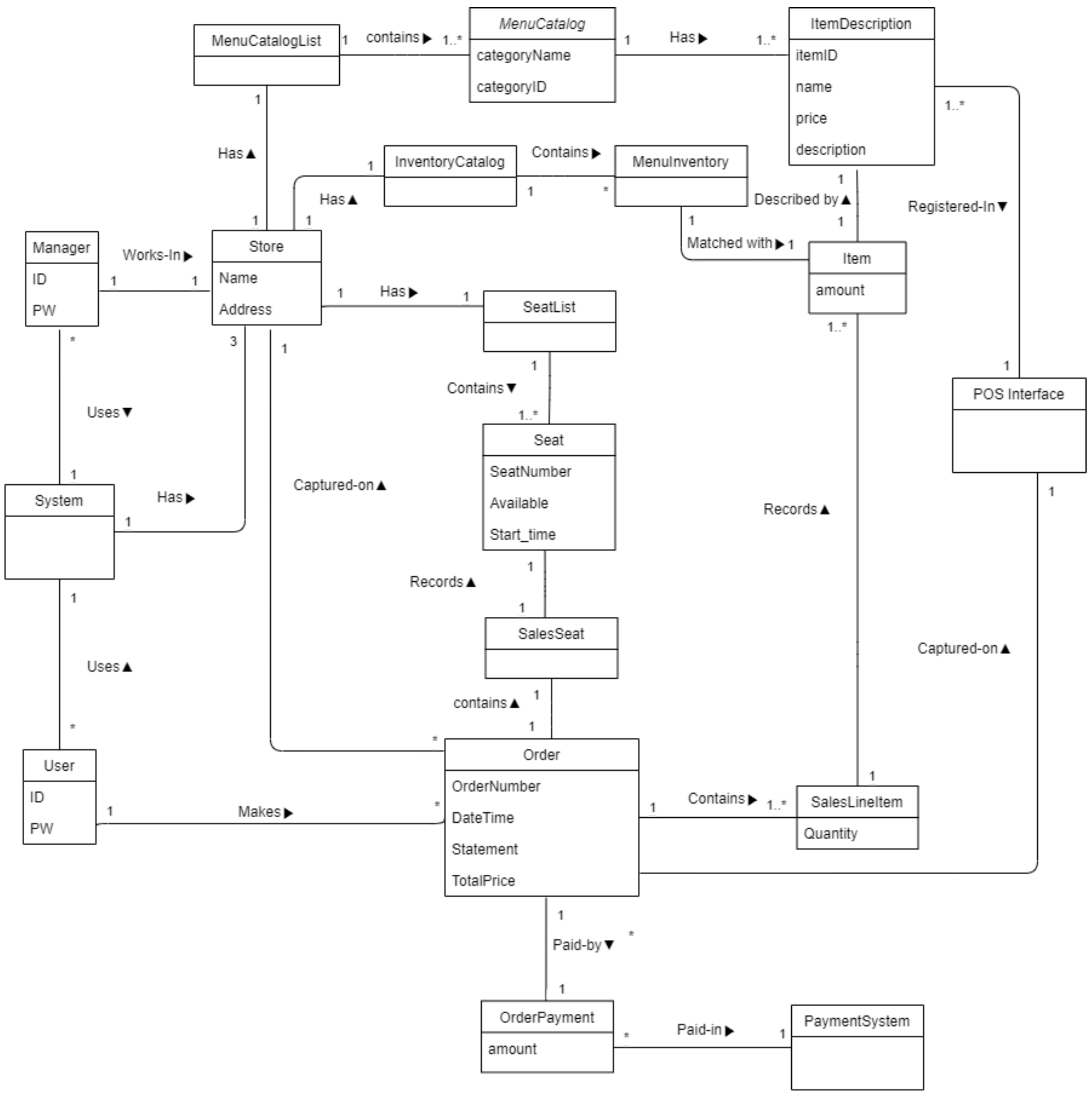


Figure 4 Domain Model Diagram - usecase1_Reservation

3.2. System Sequence Diagram

3.2.1. Use Case 1 : Reservation

3.2.1.1. UC1 Reservation : System Sequence Diagram

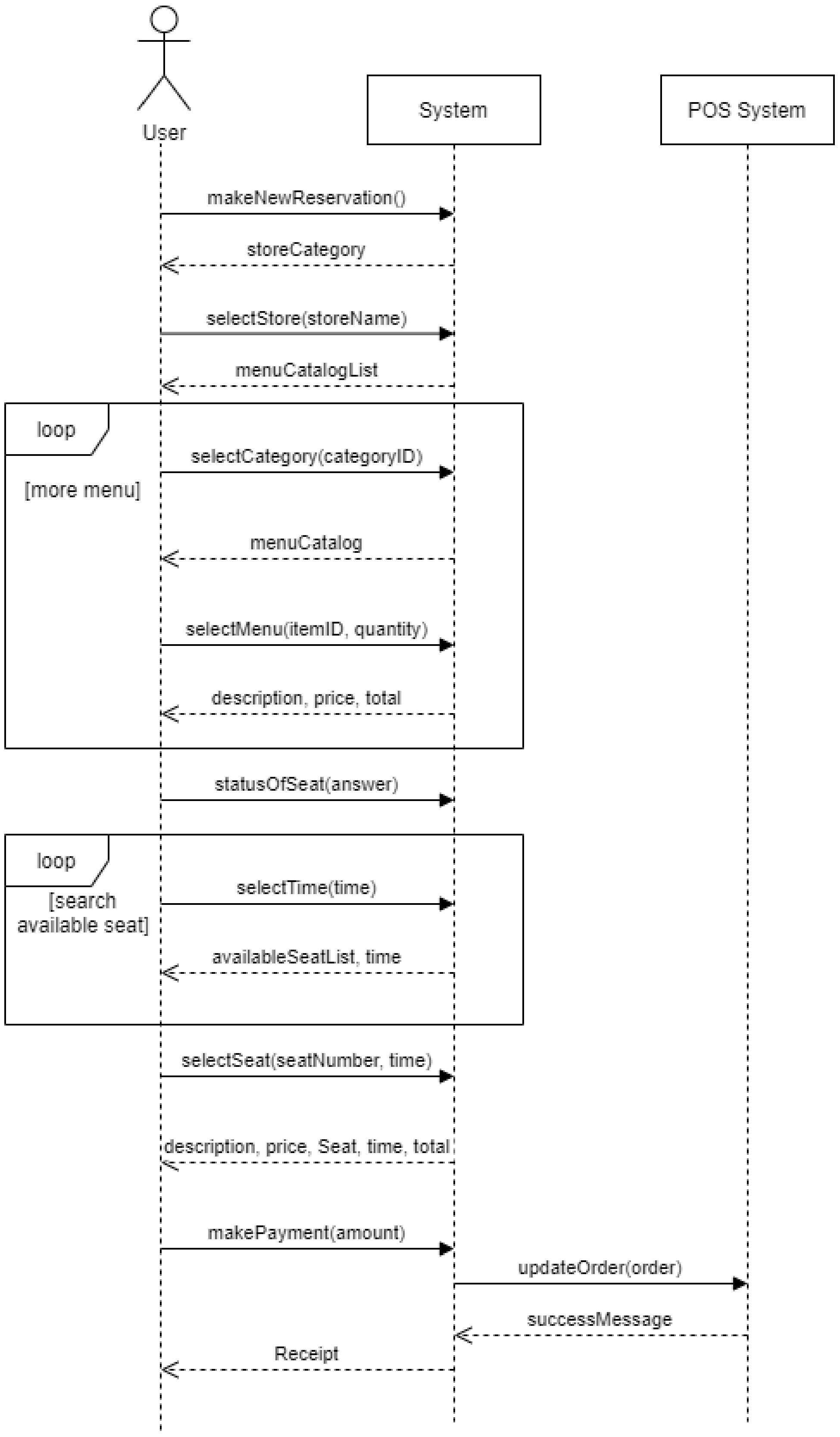


Figure 5. UC1 Reservation : System Sequence Diagram

3.2.1.2. UC1 Reservation : Brief description of system operation

makeNewReservation()	새로운 예약을 생성하는 요청을 시스템에 전송하고, 어디에서 예약을 진행할 수 있는지 시스템 내에 존재하는 store에 대한 정보를 반환 받는다.
selectStore(storeName : String)	Customer는 예약을 원하는 store에 대한 정보를 담아 시스템에 전송한다. 그 정보를 받은 시스템은 메뉴 카타로그 리스트를 반환해준다.
selectCategory(categoryID : int)	시스템을 통해 메뉴 카타로그 리스트를 받은 Customer는 원하는 음료카테고리류를 속성으로 함께 담아 전송한다. 그 속성을 받은 시스템은 그 카타로그 리스트에 맞는 음료 메뉴들을 보여준다.
selectMenu(itemID,quantity)	Customer는 원하는 음료의 정보와 수량을 시스템에 전송해준다. 시스템은 좌석 예약 여부에 대한 것을 묻는다. 좌석 예약하면 (1) 예약하지 않으면 (0)
statusOfSeat(answer : boolean)	Customer는 좌석을 예약할지에 대한 여부를 입력한다. 만약, 좌석예약을 원하지 않는다면 시스템은 픽업을 원하는 시간만 입력받고 좌석예약을 원한다면 시스템은 시간과 좌석을 입력받게 된다.
selectTime(time : int)	원하는 시간대의 예약 좌석 현황을을 요청한다. System은 Customer가 요청한 time을 가지고 그 시간 대의 예약 좌석 현황을 보여준다.
selectSeat(seatNumber,time)	Customer가 원하는 좌석과 확정된 시간을 System에게 전달한다. System은 이 정보를 바탕으로 위의 음료 예약과 합한 price total을 전송한다.
makePayment(amount)	price total을 바탕으로 결제를 수행한다.

Table 5 [UC1] Reservation : Brief description of system operation

3.2.1.3. UC1 Reservation : Operation Contracts

Contract CO1 : makeNewReservation
Operation : makeNewReservation()
Cross reference : Reservation
Precondition : none
Postcondition :

- A order instance o was created
- Attributes of o were initialized
- o was associated with a System.

Table 6 [UC1] - CO1 makeNewReservation

Contract CO2 : selectStore

Operation : selectStore(storeName : String)

Cross reference : Reservation

Precondition :

- There is a reservation underway.
- there is a Order instance o.

Postcondition :

- o was associated with a store

Table 7 [UC1] - CO2 selectStore

Contract CO3 : selectMenu

Operation : selectMenu(ItemId : int, quantity : int)

Cross reference : Reservation

Precondition :

- There is a reservation underway.
- there is a Order instance o.

Postcondition :

- A SalesLineItem instance sli was created.(instance creation)
- sli was associated with o(association formed)
- sli.quantity became quantity. (attribute modification)
- sli was associated with Item, based on itemId.(association formed)
- total value that attribute of ItemDescription added sli.quantity(Find the total for which item by ItemID.)

Table 8 [UC1] - CO3 selectMenu

Contract CO4 : selectSeat

Operation : selectSeat(seatNumber : int, time)

Cross reference : Reservation

Precondition :

- There is a reservation underway.

- There is a Order instance o

Postcondition :

- A SalesSeat instance ss was created.
- ss was associated with Seat, based on seatNumber.
- o was associated with ss.
- Seat.available became False, based on seatNumber.
- Seat.start_time became time, based on seatNumber.
- o.time became time

Table 9 [UC1] - CO4 selectSeat

Contract CO5 : makePayment

Operation : makePayment(amount : int)

Cross reference : Reservation

Precondition :

- There is a reservation underway.

Postcondition :

- A OrderPayment instance p was created. (instance creation)
- p.amount became amount (attribute modification)
- p was associated with the current Order o (association formed)
- o.statement became ‘ready’

Table 10 [UC1] - CO5 makePayment

3.2.2. Use Case 2 : Menu management

3.2.2.1. UC2 Menu management : System Sequence Diagram

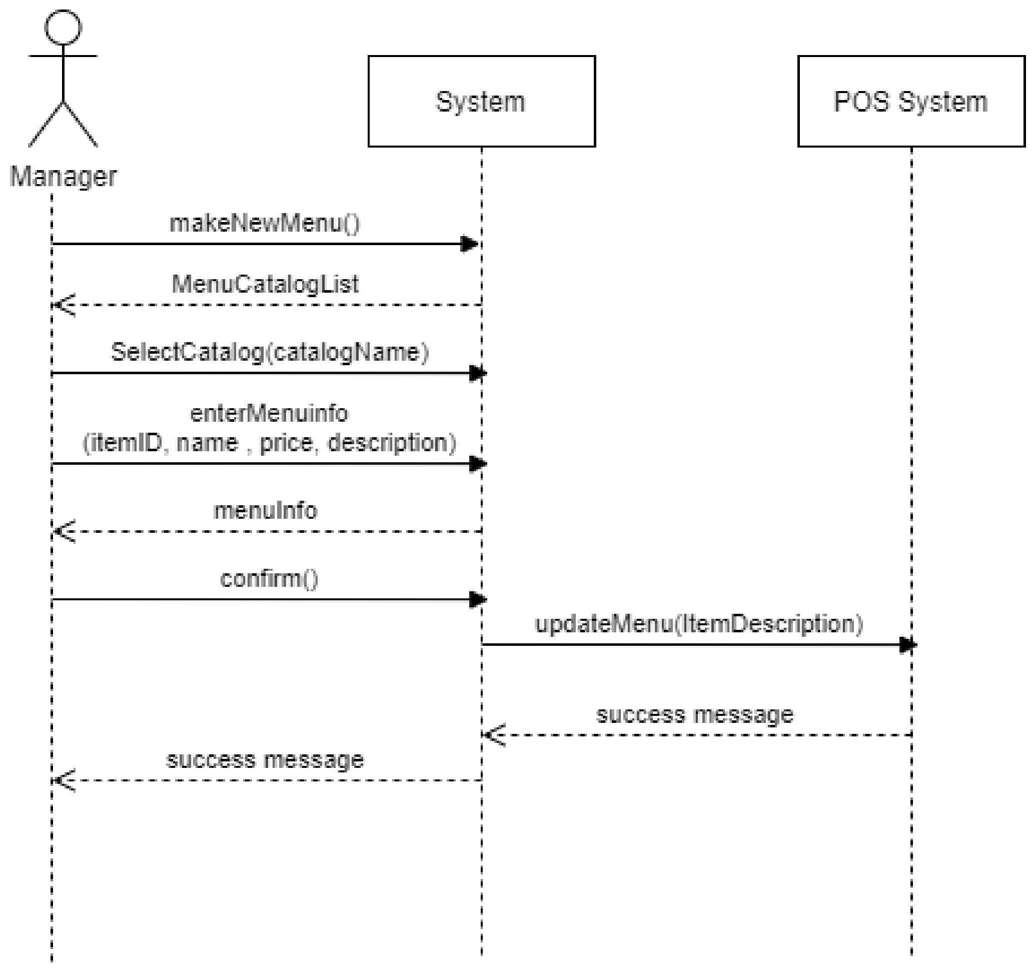


Figure 6. UC2 Menu management : System Sequence Diagram

3.2.2.2. UC2 Menu management: Brief description of system operation

makeNewMenu()	메뉴 추가 기능 요청을 시스템에 전송하고, MenuCatalogList를 반환받는다.
selectCatalog(catalogName)	메뉴의 카탈로그 이름을 입력해서 시스템에 전송한다. (커피,주스,에이드와 같은) 시스템은 manager가 입력한 카탈로그에 추가될 음료의 inputable form 반환한다.
enterMenuinfo(itemID,name,price, description)	manager가 추가하고 싶은 음료의 정보와 가격을 inputable form에 맞춰 시스템에 전송한뒤 시스템은 manager에게 입력된 정보를 확인시킨다.
confirm()	manager는 입력된 정보를 확인한 후 메뉴등록을 확정한다. 시스템은 등록된 메뉴를 POS에 등록한 후 Manager에게 success message를 보낸다.

Table 11 [UC2] Menu management : Brief description of system operation

3.2.2.3. UC2 Menu management : Operation Contracts

Contract CO1 : makeNewMenu

Operation : makeNewMenu()

Cross reference : Menu Management

Precondition :

Postcondition :

- A Item instance ie was created.
- ie was associated with itd
- A ItemDescription instance itd was created.
- itd.itemID, itd.name and itd.price was initialized.

Table 12 [UC2] - CO1 makeNewMenu

Contract CO2 : selectCatalog

Operation : selectCatalog(categoryName : String)

Cross reference : Menu Management

Precondition :

- There is a Menu Management underway.
- There is a MenuCatalog instance mc.

Postcondition :

- itd was associated with mc, based on categoryName

Table 13 [UC2] - CO2 selectCatalog

Contract CO3 : enterMenuinfo

Operation : enterMenuinfo(itemID : int ,name : String, price : int, description : String)

Cross reference : Menu Management

Precondition :

- There is a Menu Management underway.
- there is a ItemDescription instance itd.

Postcondition :

- itd.itemID became itemID, itd.name became name, itd.price became price, itd.description became description.

Table 14 [UC2] - CO3 enterMenuinfo

Contract CO4 : confirm

Operation : confirm()

Cross references : Manage MenuInfo

Preconditions :

- There is a make new Menu underway.

Postconditions :

- itd was associated with POS interface.

Table 15 [UC2] - CO4 confirm

4. Design Modeling

4.1. UC1 Reservation : Realization

4.1.1. Design Sequence Diagrams

상단 Operate Contract 중 selectStore(storeName : String),statusOfSeat(answer), selectCatalog(categoryName : String)에 대한 Operation은 System이 User인 Customer에게 UI로 보여주고, 변수에 값을 입력하는 진행하는 단순한단계이기 때문에 Sequence Diagram 단계에서 생략했다. 이외에는 Use case가 어떻게 구현되는지 인지가 필요한 모든 Operation은 sequence diagram을 작성했다.

4.1.1.1. Operation 1: makeNewReservation()

4.1.1.1.1. Sequence Diagram

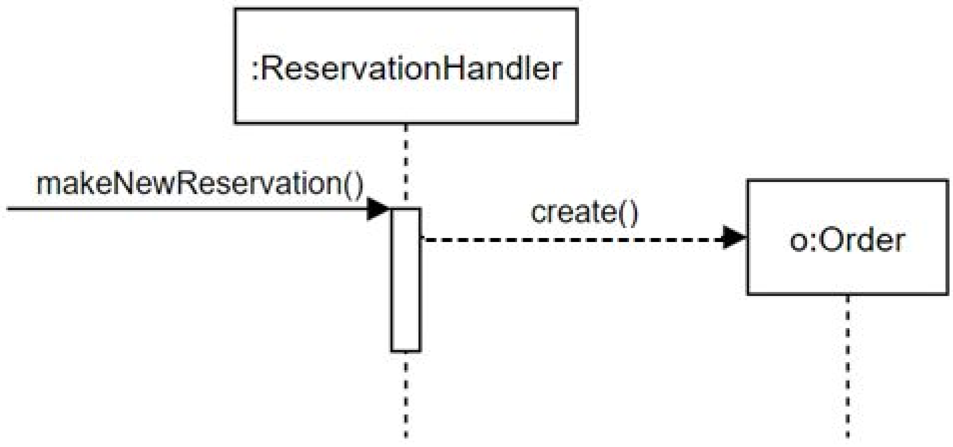


Figure 7. UC1 - Operation 1 - interaction diagram : makeNewReservation()

4.1.1.1.2. GRASP Pattern

Controller	- **ReservationHandler** is the first object beyond the UI layer receives and coordinates a system operation.
Creator	- **ReservationHandler** creates an **Order instance.**
Information expert	NA
Low Coupling	NA
High Cohesion	NA

Table 16 [UC1] - Operation 1 - GRASP Pattern : makeNewReservation()

4.1.1.2. Operation 3 : selectMenu(itemID, quantity)

4.1.1.2.1. Sequence Diagram

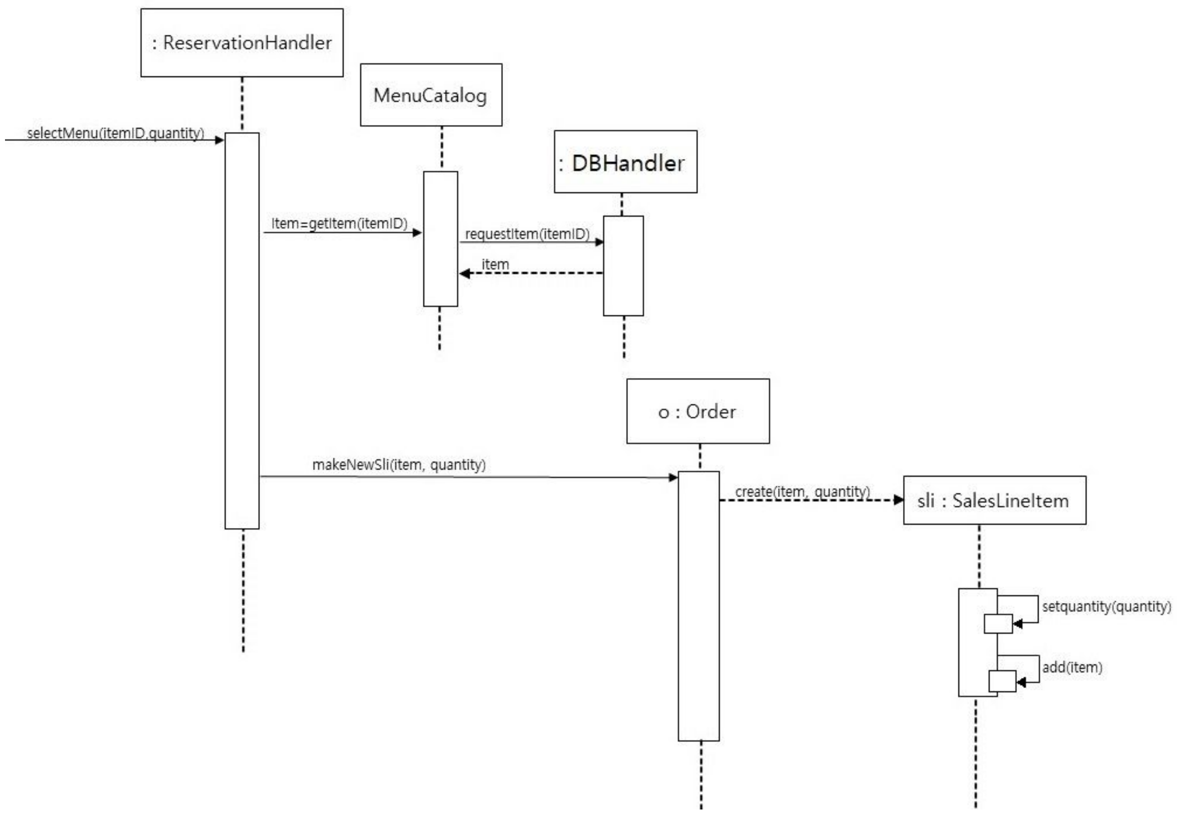


Figure 8. UC1 - Operation 3 - interaction diagram : selectMenu(itemID, quantity)

4.1.1.2.2. GRASP Pattern

Controller	- **ReservationHandler** is the first object beyond the UI layer receives and

	coordinates a system operation.
Creator	**- Order** creates **SalesLineItem.**
Information expert	**- MenuCatalog** knows **item**. **- SalesLineItem** knows **quantity**.
Low Coupling	NA
High Cohesion	NA

Table 17 [UC1] - Operation 3 - GRASP Pattern : selectMenu(itemID, quantity)

4.1.1.3. Operation 4 : selectSeat(seatNumber, time)

4.1.1.3.1. Sequence Diagram

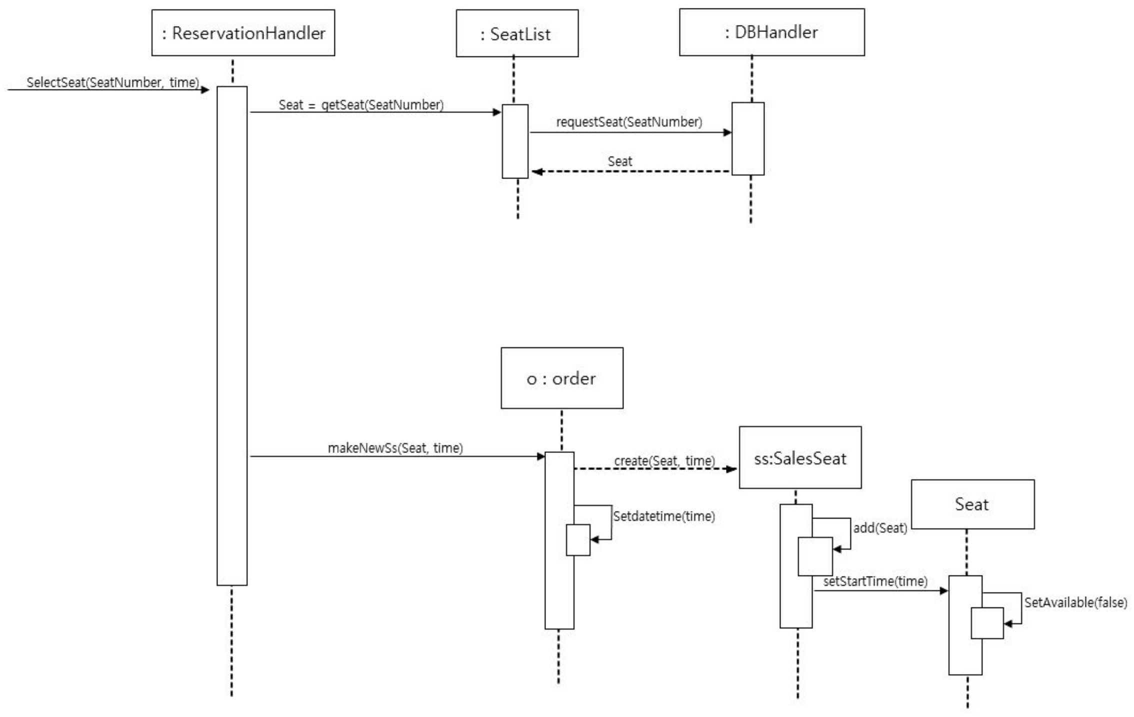


Figure 9. UC1 - Operation 4 - interaction diagram : selectSeat(seatNumber, time)

4.1.1.3.2. GRASP Pattern

Controller	**- ReservationHandler** is first object beyond the UI layer receives and coordinates a system operation.
Creator	**- Order** create **SalesSeat.**

Information expert	**- SeatList** knows **Seat.** **- Order** knows **datetime.** **- Seat** knows **Start_time, Available.**
Low Coupling	**- ReservationHandler** delegates the create **SalesSeat** responsibility to the **Order.** **- ReservationHandler** delegates the requestSeat responsibility to the **SeatList.**
High Cohesion	NA

Table 18 [UC1] - Operation 4 - GRASP Pattern : selectSeat(seatNumber, time)

4.1.1.4. Operation 5 : makePayment(amount)

4.1.1.4.1. Sequence Diagram

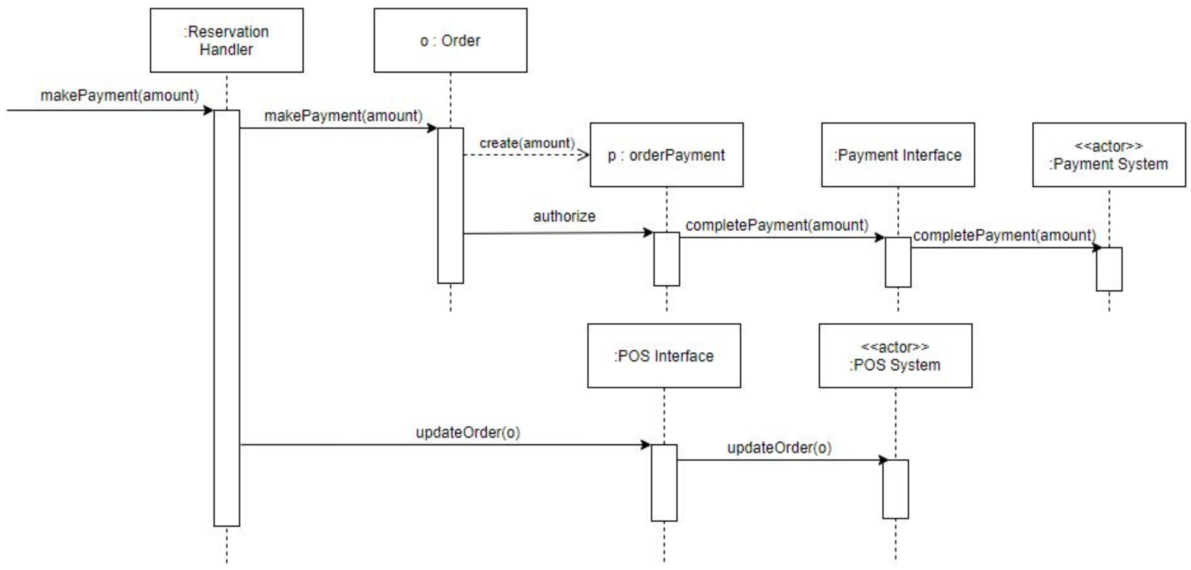


Figure 10. UC1 - Operation 5 - interaction diagram : makePayment(amount)

4.1.1.4.2. GRASP Pattern

Creator	**- Order** creates **orderPayment.**
Controller	**- ReservationHandler** is first object beyond the UI layer receives and coordinates a system operation.
Information expert	**- OrderPayment** knows amount**.** **- ReservationHandler** knows **Order.**
Low Coupling	**- ReservationHandler** delegates the create orderPayment responsibility to the **Order.**

High Cohesion	NA

Table 19 [UC1] - Operation 5 - GRASP Pattern : makePayment(amount)

4.1.2. Combined DCD

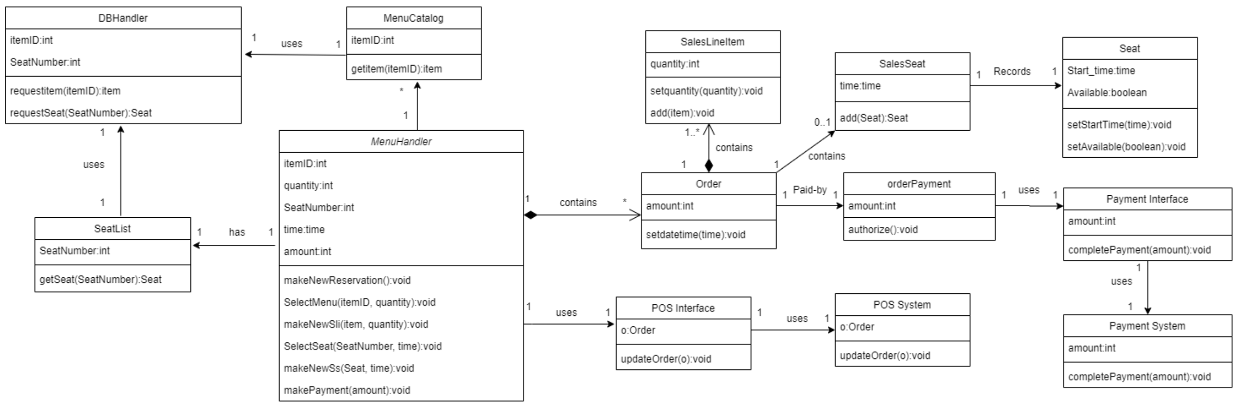


Figure 11. Use Case 1 : Combined Design Class Diagram

4.2. UC2 Menu Management : Realization

4.2.1. Design Sequence Diagrams

UC2인 Menu Management에서는 모든 Operation에 대해 Use case가 어떻게 구현되는지 인지가 필요하기 때문에 sequence Diagram을 작성했다.

4.2.1.1. Operation 1 : makeNewMenu()

4.2.1.1.1. Sequence Diagram

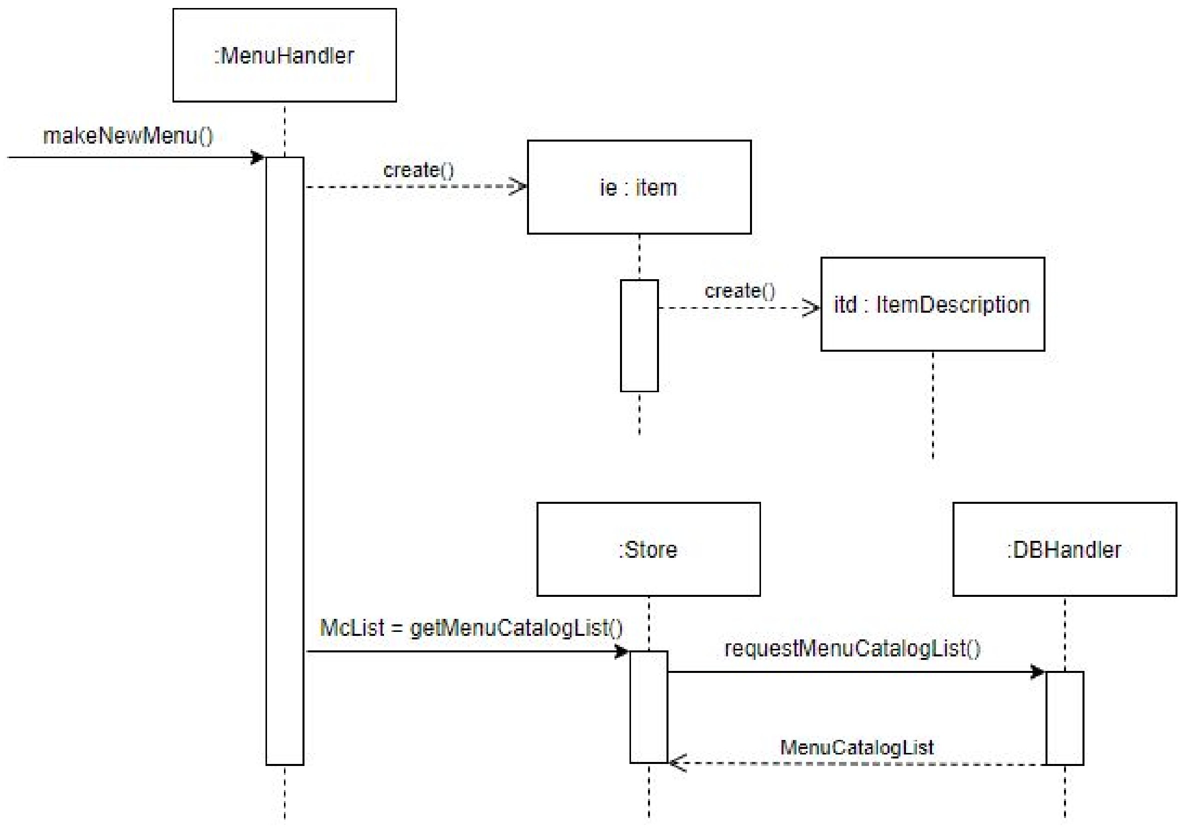


Figure 12. UC 2 - Operation 1 - interaction diagram : makeNewMenu()

4.2.1.1.2. GRASP Pattern

Creator	- **MenuHandler** creates **Item.** - **Item** creates **ItemDescription.**
Controller	- **MenuHandler** is first object beyond the UI layer receives and coordinates a system operation.
Information expert	- **Store** knows **MenuCatalogList.**
Low Coupling	- **MenuHandler** delegates the create ItemDescription responsibility to the **Item.** - **MenuHandler** delegates the requesMenuCatalogList responsibility to the **Store.**
High Cohesion	NA

Table 20 [UC2] - Operation 1 - GRASP Pattern : makeNewMenu()

4.2.1.2. Operation 2 : enterCatalog(categoryID)

4.2.1.2.1. Sequence Diagram

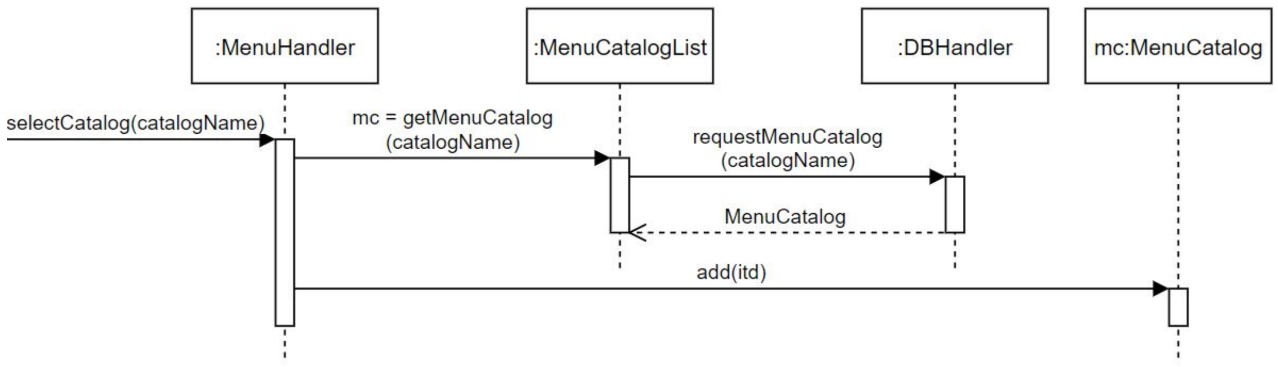

:MenuHandler
:MenuCatalogList
:DBHandler
mc:MenuCatalog
selectCatalog(catalogName)
mc = getMenuCatalog (catalogName)
requestMenuCatalog (catalogName)
MenuCatalog
add(itd)

Figure 13. UC 2 - Operation 2 - interaction diagram : selectCatalog(catalogName)

4.2.1.2.2. GRASP Pattern

Creator	NA
Controller	**- MenuHandler** is first object beyond the UI layer receives and coordinates a system operation
Information expert	**- MenuCatalogList** knows **MenuCatalog**. **- MenuHandler** knows **itd.**
Low Coupling	**- MenuHandler** delegates the requestMenuCatalog responsibility to the **MenuCatalogList** .
High Cohesion	NA

Table 21 [UC2] - Operation 2 - GRASP Pattern : selectCatalog(catalogName)

4.2.1.3. Operation 3 : enterMenuinfo(itemID, name, price, description)

4.2.1.3.1. Sequence Diagram

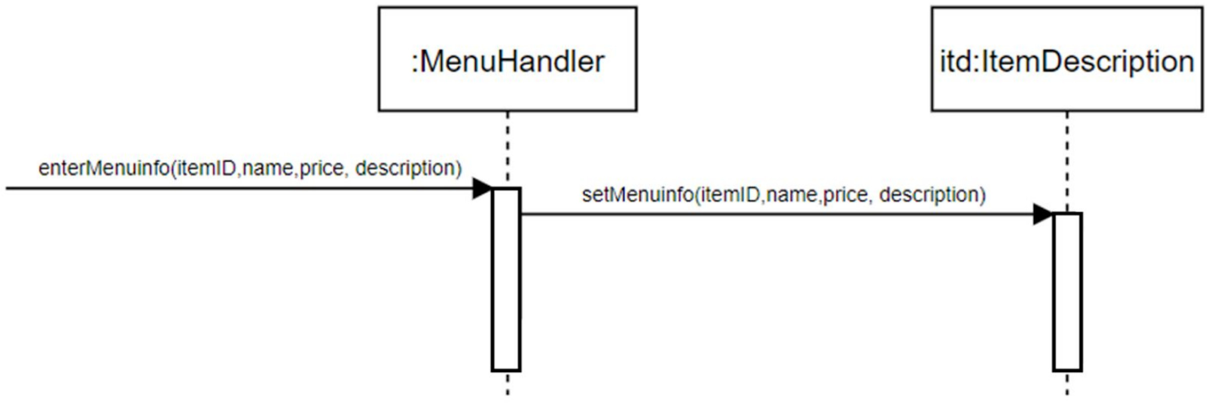

:MenuHandler
itd:ItemDescription
enterMenuinfo(itemID,name,price, description)
setMenuinfo(itemID,name,price, description)

Figure 14. UC 2 - Operation 3 - interaction diagram : enterMenuinfo(itemID, name, price, description)

4.2.1.3.2. GRASP Pattern

Creator	NA
Controller	**-MenuHandler** is first object beyond the UI layer receives and coordinates a system operation
Information expert	**- ItemDescription** knows **itemID, name, price, description**.
Low Coupling	NA
High Cohesion	NA

Table 22 [UC2] - Operation 3 - GRASP Pattern : enterMenuinfo(itemID, name, price, description)

4.2.1.4. Operation 4 : confirm()

4.2.1.4.1. Sequence Diagram

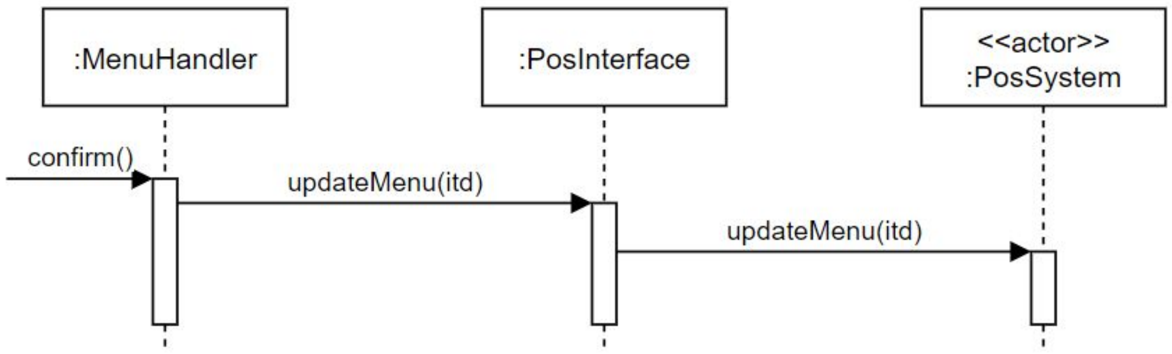


Figure 15. UC 2 - Operation 4 - interaction diagram : confirm()

4.2.1.4.2. GRASP Pattern

Creator	NA
Controller	**- MenuHandler** is first object beyond the UI layer receives and coordinates a system operation
Information expert	**- MenuHandler** knows **itd.**
Low Coupling	NA
High Cohesion	NA

Table 23 [UC2] - Operation 4 - GRASP Pattern : confirm()

4.2.2. Combined DCD

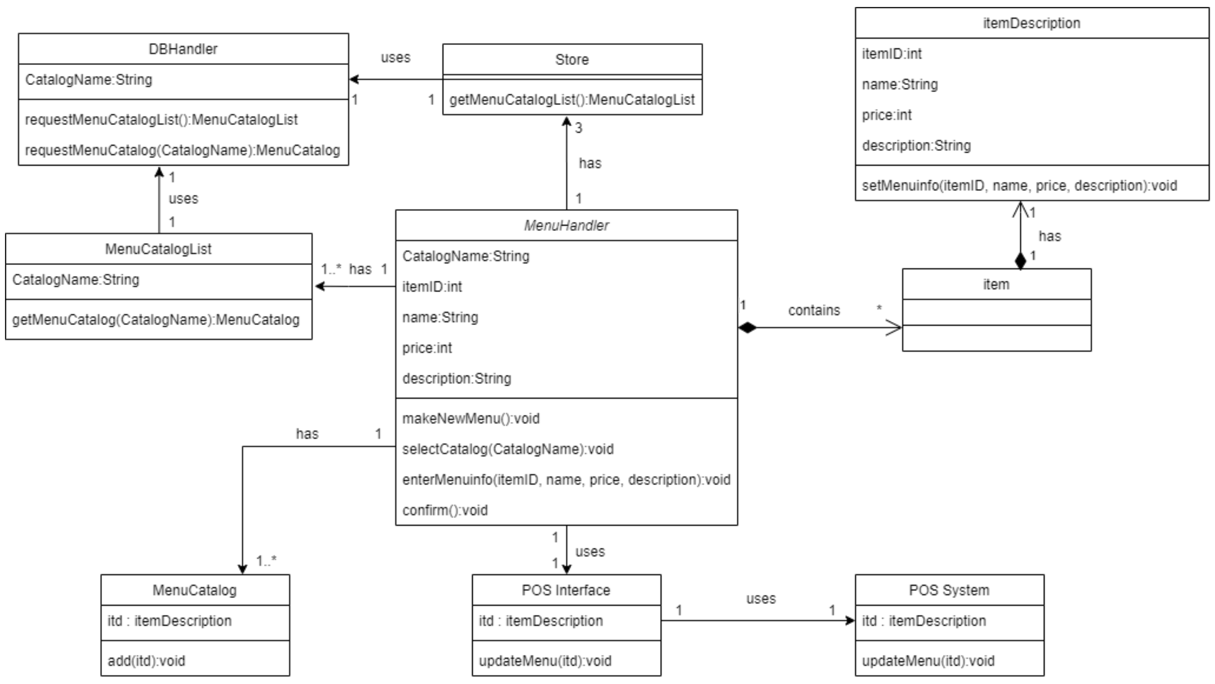


Figure 16. Use Case 2 : Combined Design Class Diagram

5. References

Figure 1 - https://contents.opensurvey.co.kr/form_coffee_2020

6. Appendix (Glossary)

Term	Definition and Information	Note
시스템 운영자	시스템의 유지보수와 에러수정, 편의성 개선등을 하는 사람	
Manager	카페의 운영을 담당하며 시스템의 메뉴관리를 수행할 수 있는 사람	
Customer	예약 시스템을 이용해 음료를 예약하고자 하는 카페 손님	

Reservation	시스템 상에서 원하는 음료, 시간, 자리를 선택해서 예약 시스템에 저장하고 예약한 내용을 수정하거나 취소할 수 있는 기능	
Sales Statistics	매니저가 매출의 통계 등을 볼 수 있는 기능	
Menu management	카페 매니저가 시스템에 메뉴를 추가하거나 제거할 수 있는 기능	
예약 시스템	카페 이용객이 입력한 예약 내용이 저장되는 시스템	
인앱결제 시스템	예약한 음료에 대한 값을 미리 지불하고자 하는 카페 이용객을 위한 인앱 결제를 지원하는 시스템	
메뉴관리 시스템	카페 매니저가 추가한 메뉴가 저장되는 공간으로 카페 이용객이 예약할 때 저장된 메뉴를 보여줌	

Table 24 Appendix

7. Revision History

Version	Date	Description	Author
Inception Phase 1	2020.09.24	First version of the inception phase report	학교가고싶다
Domain modeling (UC2 Reservation)	2020.9.28	Drawing Domain modeling diagram for the complex use case	학교가고싶다
SSD and OC (UC2 Reservation)	2020.10.27	Drawing System sequence diagram and writing operate contract for the most complex use case	학교가고싶다
Elaboration Phase 1.1 (UC1 Authentication &UC2 Reservation)	2020.10.28	Modify Domain model diagram, System sequence diagram and Refresh Use case text	학교가고싶다
Elaboration Phase 1.2	2020.10.30	Add interaction diagrams per certain operation contracts and draw GRASP Pattern	학교가고싶다
Elaboration Phase 1.3	2020.11.02	Draw combined DCD	학교가고싶다

Table 25 Revision history