

객체지향 개발 방법론

DVM Project OOI Cycle2

#4조



201710240

이해림



201811217

이해인



201711836

송호영

Index

1. Documents Changes
2. System architecture description
3. Real Use Cases and Interaction Diagrams
4. Class Diagram
5. Manual
6. System Testing
7. Testing Traceability Analysis

Documents Changes

수정 전

Num	Term	Description
1	SW	Software의 약자
2	DVM	Distributed Vending Machine의 약자
3	선결제	사용자가 현 DVM에서 구매 불가능한 물품을 타DVM에서 구매하고자 할 때 이용하는 시스템이다. 사용자는 현 DVM에서 미리 결제를 진행한 뒤 인증번호 및 바코드를 출력 받아 해당 자판기에서 결제된 음료를 전달받을 수 있다.
4	C_Number	사용자가 선결제를 한 경우 해당 DVM에서 음료를 전달받기 위한 인증 번호이다. 이외에 관리자는 관리자 메뉴로 접속하기 위해 고유한 인증번호를 갖고있다.
5	Msg	Message의 준말로 DVM이 네트워크에 연결된 다른 DVM과 정보를 주고받는 데이터 단위이다. Message Protocol에 형식이 정의된다.
6	Title	DVM에 등록된 각각의 음료 이름
7	Title ID	DVM에 등록된 각각의 음료 번호
8	item	DVM에 등록된 Title의 수량
9	Address	DVM에 갖고 있는 자신의 위치
10	ID	네트워크 내 DVM의 고유번호
11	Broadcast Msg	DVM네트워크 내 모든 DVM에게 보내는 메시지

SRS-1	4	[1.2] [3.2.2.2.3]에서 인증코드만 출력하고 있지만, 이곳에서는 인증코드/바코드를 입력한다고 되어 있어 통일이 필요.
SRS-2	4	[1.3] C_Number에서 사용자와 관리자가 쓰는 인증번호가 다른 것이라면 명확히 표현이 되는 것이 좋아보임. 인증번호가 몇자리인지도 나타내는 것이 필요해 보임.
SRS-3	4	[1.3] Item이 수량을 나타내는 변수 이름으로 쓰이는 것이 직관성을 떨어뜨리기 때문에 변수 이름을 바꾸는 것이 좋아보임.
SRS-4	4	[1.3] Title, Title ID가 단 하나의 음료의 정보를 나타내는 것이라면 '각각의' 라는 표현을 삭제하는 것이 좋아보임.
SRS-5	4	[1.3] Address의 description 내용 중 'DVM에 갖고 있는'에서 '에'를 '이'로 수정 필요.
SRS-6	4	[1.3] Address 중 '자신의'라는 표현보다는 PFR에서 명시한 대로 물리적 위치임을 나타내는 것이 필요해 보임.
SRS-7	4	[1.3] 에 추가로 간편 결제가 무엇인지 정의하는 것이 필요해 보임.
SRS-6	4	[1.3] Address 중 '자신의'라는 표현보다는 PFR에서 명시한 대로 물리적 위치임을 나타내는 것이 필요해 보임.
SRS-7	4	[1.3] 에 추가로 간편 결제가 무엇인지 정의하는 것이 필요해 보임.

Documents Changes

수정 후

Num	Term	Description
1	SW	Software의 약자
2	DVM	Distributed Vending Machine의 약자
3	선결제	사용자가 현 DVM에서 구매 불가능한 물품을 타DVM에서 구매하고자 할 때 이용하는 시스템이다. 사용자는 현 DVM에서 미리 결제를 진행한 뒤 인증번호를 출력 받아 해당 자판기에서 결제된 음료를 전달받을 수 있다.
4	C_Number	사용자가 선결제를 한 경우 해당 DVM에서 음료를 전달받기 위한 6자리 인증 번호이다.
5	M_Number	관리자가 관리자 메뉴로 접속하기 위해 사용되는 고유한 인증 번호이다.
6	Msg	Message의 준말로 DVM이 네트워크에 연결된 다른 DVM과 정보를 주고받는 데이터 단위이다. Message Protocol에 형식이 정의된다.
7	Title	DVM에 등록된 각각의 음료 이름
8	Title Index	DVM에서 판매하는 음료 리스트내 특정 음료의 인덱스
9	item	DVM에 등록된 Title의 각 재고
10	Address	DVM에 저장된 해당 DVM의 물리적 위치
11	DVM ID	네트워크 내 DVM의 고유번호
12	Broadcast Msg	DVM네트워크 내 모든 DVM에게 보내는 메시지
13	간편 결제	화면에 출력된 QR코드를 통해 결제를 진행하는 방법.

Documents Changes

수정 전

수정 후

Constraints

1. 다중 동시 터치 입력은 허용하지 않는다
2. 타 자판기의 위치는 오직 텍스트 형식으로 표현한다.
3. UI는 오로지 한글로 제공된다
4. 환불은 불가능하다.
5. 일시불 카드결제만 가능하다.
6. 프로세스는 항상 한번에 하나만 처리된다.
7. 한 item에 대해 여러 DVM에서 동시에 결제를 진행하는 경우는 없다고 가정한다.

Constraints

1. 다중 동시 터치 입력은 허용하지 않는다
2. 타 자판기의 위치는 오직 텍스트 형식으로 표현한다.
3. UI는 한국어 버전만을 제공한다
4. 환불은 불가능하다.
5. 일시불 결제만 가능하다.
6. 동작을 처리하는 프로세스는 항상 한번에 하나만 처리된다.
7. 특정 DVM의 동일한 Title에 대해 여러 DVM에서 동시에 결제를 진행하는 경우는 없다고 가정한다.
8. 특정 DVM의 동일한 Title에 대해 관리자의 재고 관리와 다른 DVM에서의 선결제가 동시에 이루어지는 경우는 없다고 가정한다.
9. 오류/알림 메시지는 10초 동안 출력된다.
10. 결제/오류/알림 화면을 제외한 모든 화면은 3분간 입력이 없을 시 대기화면으로 전환된다.
11. 결제화면은 30초간 입력이 없을 시 메인 화면으로 전환된다.
12. 관리자 인증번호는 '111111'로 정한다.
13. 한 Title의 item은 최대 30개 까지로 제한한다

SRS-11	6	[2.4] 일시불 카드 결제만 가능하다는 조건은 간편 결제 기능이 존재한다는 것과 충돌하는 조건으로 보임.
SRS-12	6	[2.4] 프로세스에 대한 설명에서 어떤 프로세스인지 구체적으로 설명하는 것이 필요해 보임. (모든 Task를 일컫는 말인지, 동작을 처리하는 프로세스인지.)
SRS-13	6	[2.4] item을 수량이라고 정의하였기 때문에 한 item에 대해 라는 표현이 적절하지 못함.

System architecture description

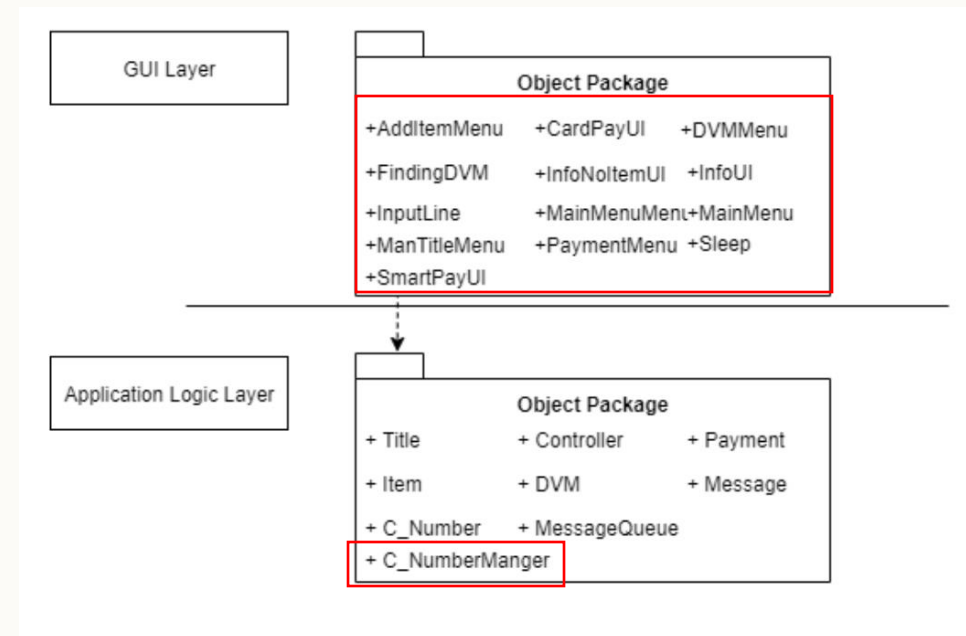
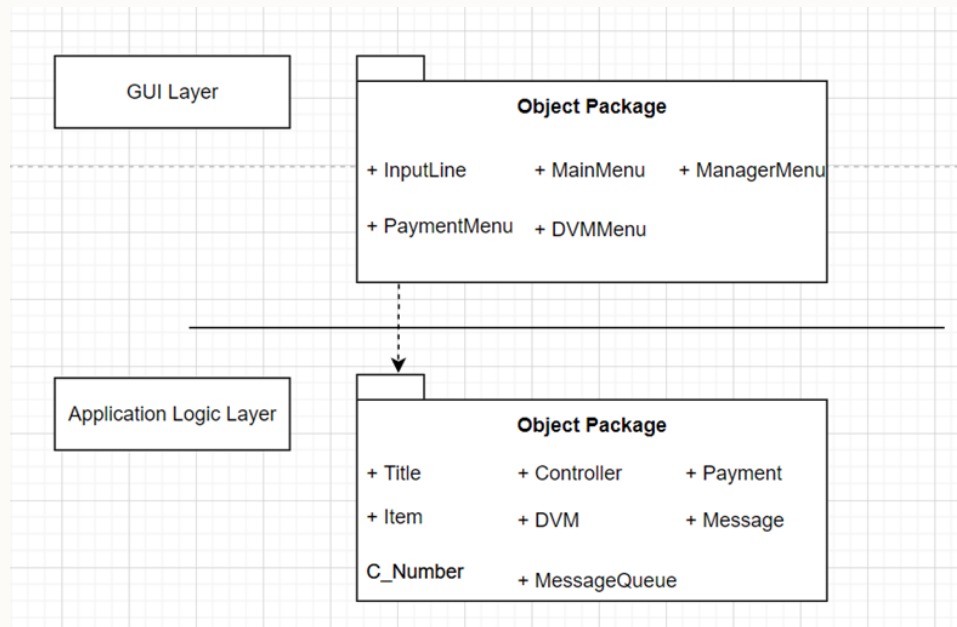
수정 전

수정 후

SDS-2

4

System architecture description에서 C_Number 앞에 +가 없음



Real Use Cases and Interaction Diagrams

SDS-2	5	<p>현재 작성된 Sequence Diagram을 보면 Customer라는 Actor가 ShowMainMenu()라는 Operation을 가지고 있는 것으로 보이는데, 이는 실제와 다름.</p> <p>아래의 모든 Sequence Diagram이 이렇게 표현되어 있는데 수정 권장.</p>
-------	---	---

->위의 내용에 따라 이하 모든 시퀀스 다이어그램을 수정하였음

Real Use Cases and Interaction Diagrams

수정 전

수정 후

SDS-3 | 5 | [3.1.1] Pre-Requisites의 "종료"는 적절하지 않은 표현임

Use Case	1.음료 리스트화면 출력
Actor	Customer
Purpose	사용자에게 메뉴를 표시
Overview	사용자가 자판기에 어떠한 입력을 가했을 경우 디스플레이 화면이 켜지면서 메뉴가 출력됨.
Type	Evident
Cross Reference	N/A
Pre-Requisites	3분간 어떠한 입력도 받지 않아 종료 상태 여야 함
Typical Courses of Events	(A) : Actor (S) : System 1. (A) : 사용자가DVM 스크린을 터치 2. (S) : 시스템이 이를 인식 후 메뉴 화면을 출력함
Alternative Courses of Events	N/A
Exceptional Courses of Events	N/A

Use Case	1.음료 리스트화면 출력
Actor	Customer
Purpose	사용자에게 메뉴를 표시
Overview	사용자가 자판기에 어떠한 입력을 가했을 경우 디스플레이 화면이 켜지면서 메뉴가 출력됨.
Type	Evident
Cross Reference	N/A
Pre-Requisites	3분간 어떠한 입력도 받지 않아 대기 상태 여야 함
Typical Courses of Events	(A) : Actor (S) : System 1. (A) : 사용자가DVM 스크린을 터치 2. (S) : 시스템이 이를 인식 후 메뉴 화면을 출력함
Alternative Courses of Events	N/A
Exceptional Courses of Events	N/A

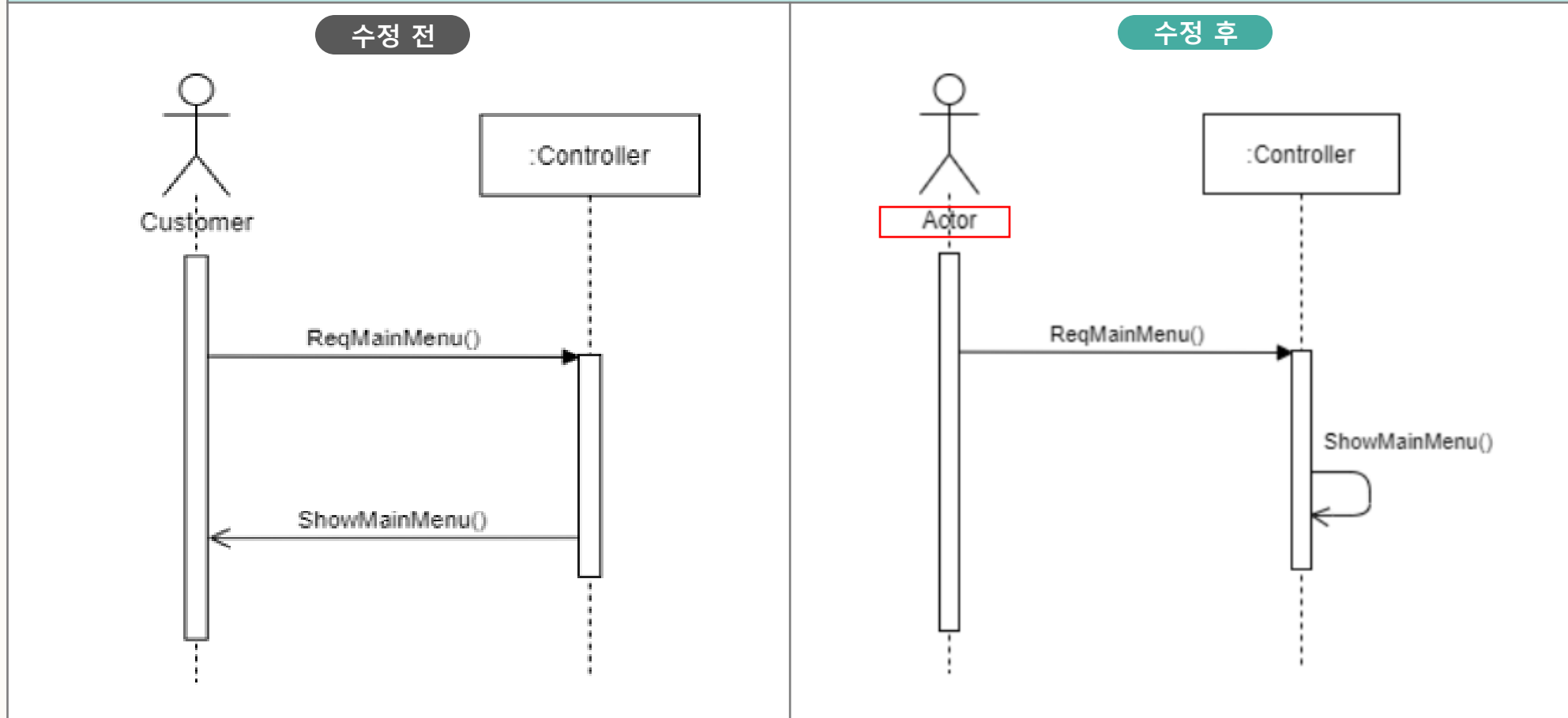
Real Use Cases and Interaction Diagrams

SDS-5

5

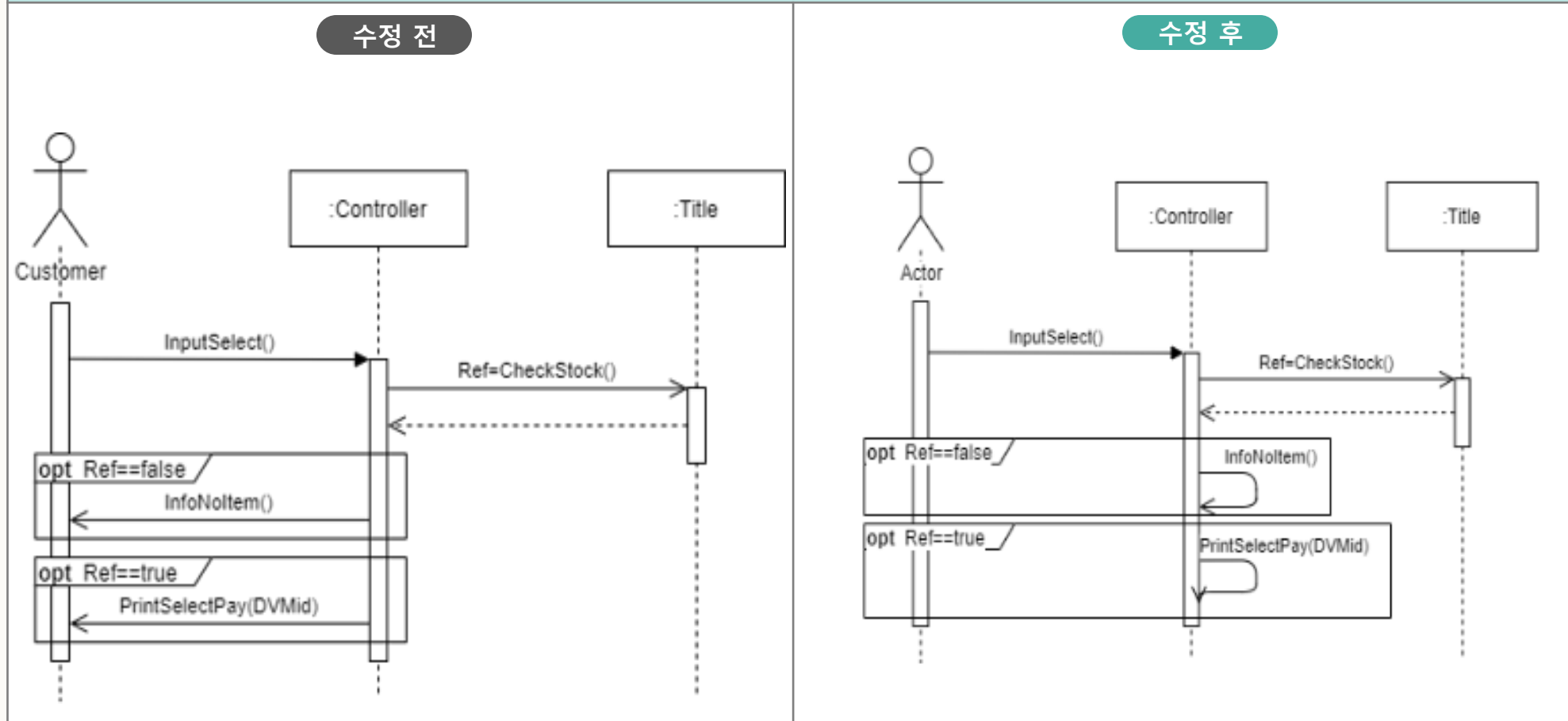
[3.1.1] Sequence Diagram에서는 주체가 Actor로 표현되어 있고 다른 모든 Sequence Diagram에는 주체가 Customer로 표현되어 있음. 표현을 통일해야함

Sequence Diagram 1.음료 리스트화면 출력



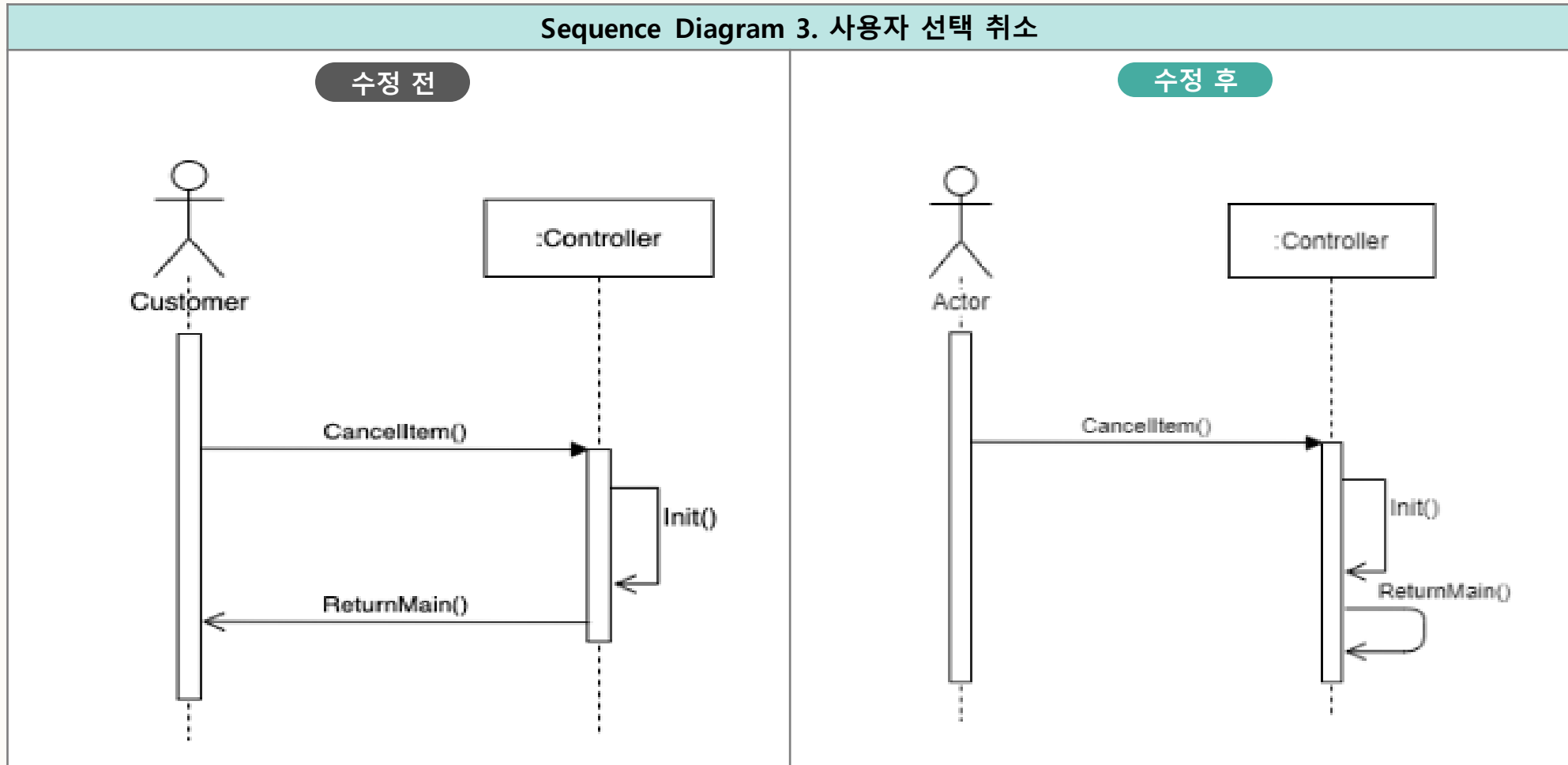
Real Use Cases and Interaction Diagrams

Sequence Diagram 2. 구매할 음료 입력



Real Use Cases and Interaction Diagrams

Sequence Diagram 3. 사용자 선택 취소



Real Use Cases and Interaction Diagrams

수정 전

SDS-7

7

[3.1.4] Typical Courses of Events 중 5번 과정이 어떤 식으로 이루어지는지 표현되지 않음

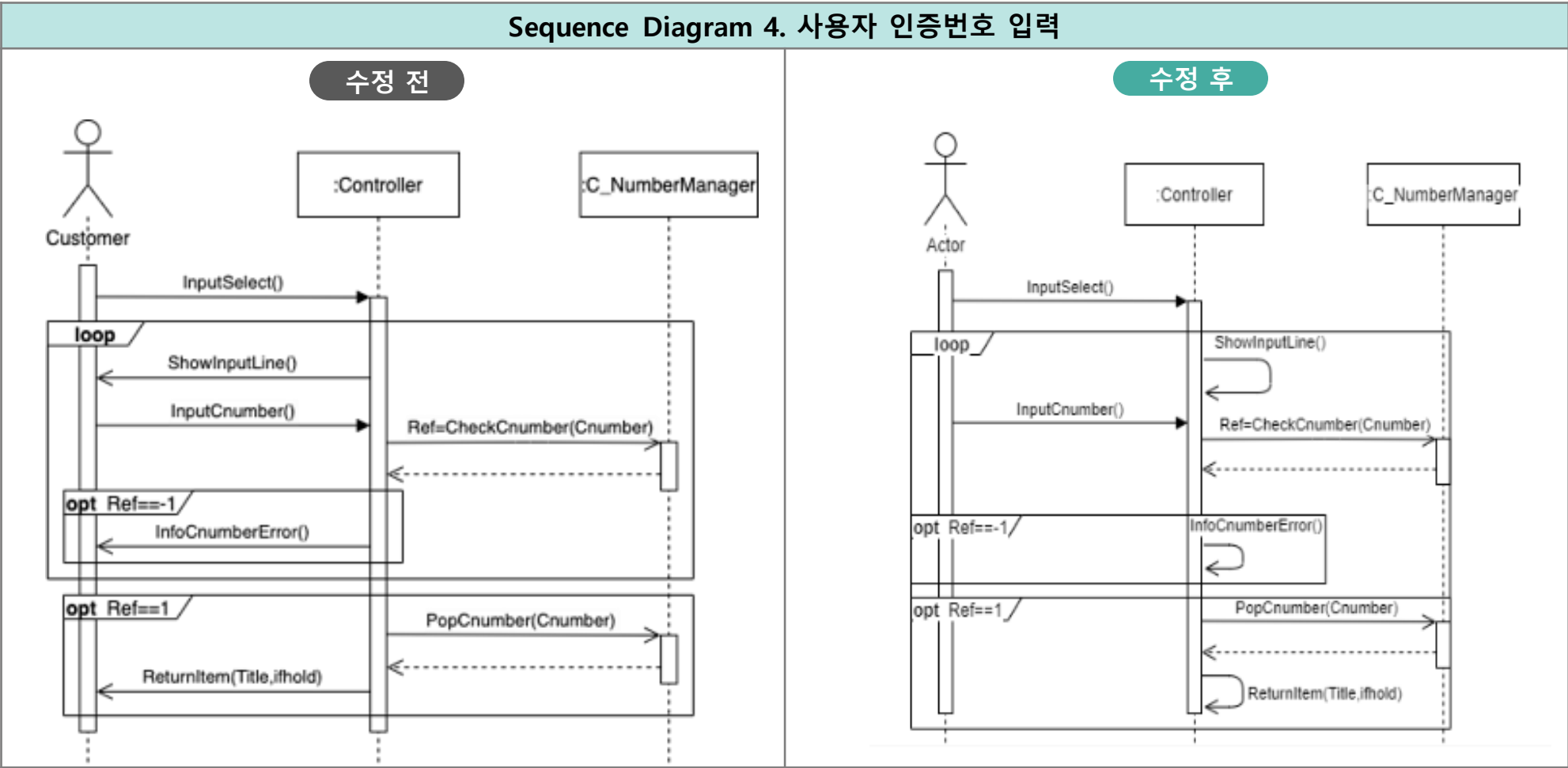
수정 후

Use Case	4. 사용자 인증번호/바코드 입력
Actor	Customer
Purpose	선결제한 사용자가 제품을 전달받기 위해 인증번호를 입력
Overview	사용자가 적합한 인증번호 입력 시 시스템이 해당 제품 전달
Type	Evident
Cross Reference	Functions: R3.1.1, R3.1.2, R3.2
Pre-Requisites	사용자가 타 DVM에서 인증번호를 출력 받은 상태 여야 함
Typical Courses of Events	(A) : Actor (S) : System 1. (A) : 사용자가 DVM에 인증번호 입력버튼을 선택 2. (S) : 시스템이 이를 인식 후 입력 화면을 출력 3. (A) : 사용자가 이전에 출력 받은 인증번호를 입력 4. (S) : C_Number Stack에서 해당 인증번호를 검색 5. (S) : 해당 C_Number에 지정된 Title을 확인 6. (S) : 인증번호에 따른 제품을 전달
Alternative Courses of Events	A3.1 사용자(관리자)가 관리자 메뉴에 접근하려는 경우 Ref)19.관리자 메뉴 출력.Typical Courses of Event
Exceptional Courses of Events	E3.1 사용자가 입력한 인증번호를 찾지 못한 경우 1. 시스템이 오류 메시지 출력 2. 이전화면으로 복귀

Use Case	4. 사용자 인증번호 입력
Actor	Customer
Purpose	선결제한 사용자가 제품을 전달받기 위해 인증번호를 입력
Overview	사용자가 적합한 인증번호 입력 시 시스템이 해당 제품 전달
Type	Evident
Cross Reference	Functions: R3.1.1, R3.1.2, R3.2
Pre-Requisites	사용자가 타 DVM에서 인증번호를 출력 받은 상태 여야 함
Typical Courses of Events	(A) : Actor (S) : System 1. (A) : 사용자가 DVM에 인증번호 입력버튼을 선택 2. (S) : 시스템이 이를 인식 후 입력 화면을 출력 3. (A) : 사용자가 이전에 출력 받은 인증번호를 입력 4. (S) : C_Number Stack에서 해당 인증번호를 검색 5. (S) : 해당 C_Number 객체에 저장된 Title을 확인 6. (S) : 인증번호에 따른 제품을 전달
Alternative Courses of Events	A3.1 사용자(관리자)가 관리자 메뉴에 접근하려는 경우 Ref)19.관리자 메뉴 출력.Typical Courses of Event
Exceptional Courses of Events	E3.1 사용자가 입력한 인증번호를 찾지 못한 경우 1. 시스템이 오류 메시지 출력 2. 이전화면으로 복귀

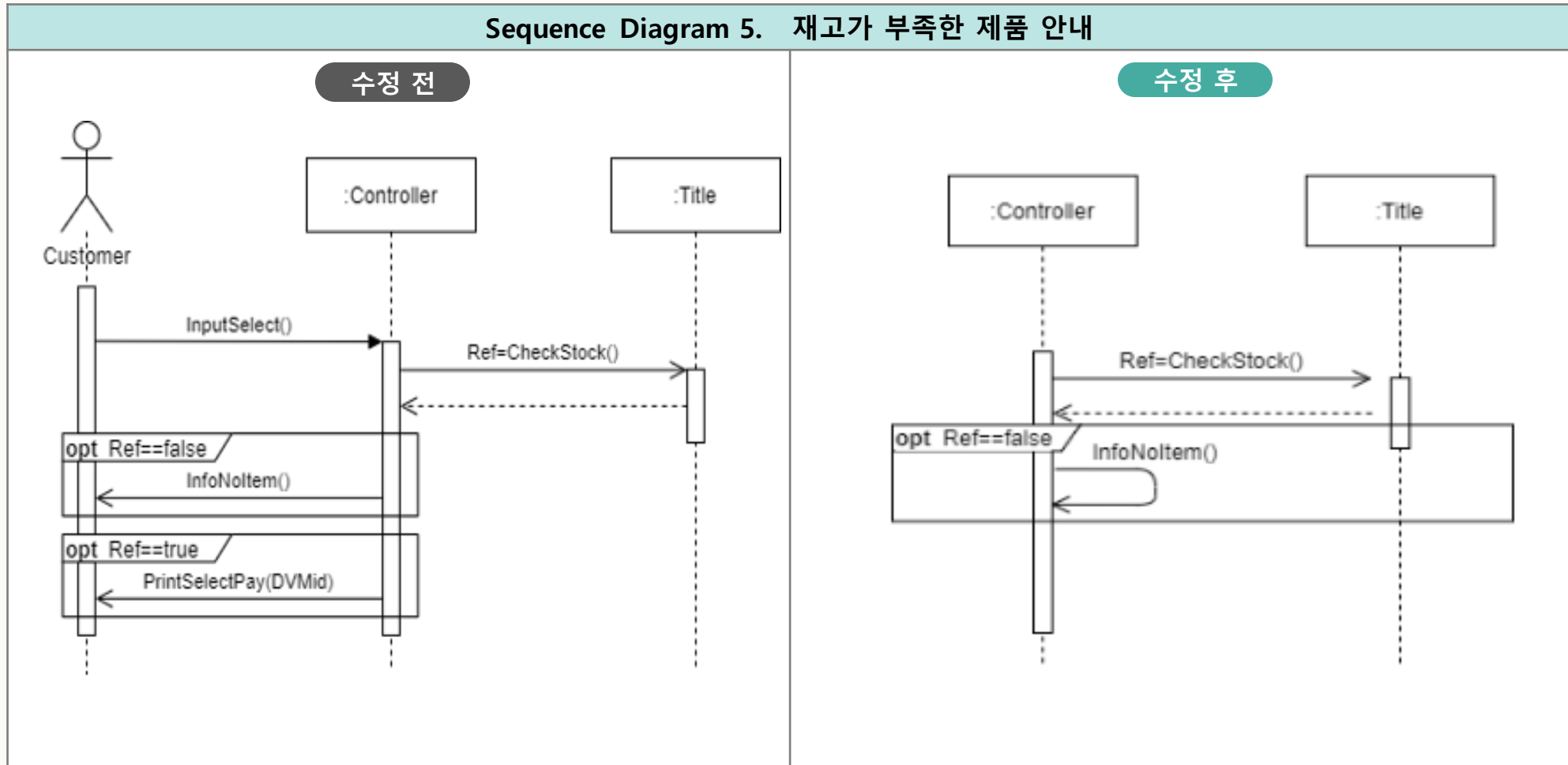
Real Use Cases and Interaction Diagrams

Sequence Diagram 4. 사용자 인증번호 입력



Real Use Cases and Interaction Diagrams

Sequence Diagram 5. 재고가 부족한 제품 안내



Real Use Cases and Interaction Diagrams

수정 전

SDS-15

9

[3.1.6] Typical Courses of Events의 2번에 "Msg protocol"이란 표현이 있는데 "MessageQueue"를 의도한 것이라면 수정해야 함

수정 후

Use Case	6. 구매 가능한 자판기 안내
Actor	None
Purpose	사용자에게 구매 가능한 자판기 안내
Overview	사용자가 선택한 음료의 재고가 있는 타 DVM을 안내한다.
Type	Hidden
Cross Reference	R1.3, R4.5.1
Pre-Requisites	사용자가 재고가 부족한 제품 안내를 받은 상황
Typical Courses of Events	(A) : Actor (S) : System 1. (A) : 사용자가 구매 가능한 자판기 안내 버튼을 입력 2. (S) : 저장된 Title을 Msg protocol에 저장. 3. (S) : Ref)21 메시지 발신을 이용해 broadcast Msg 발신 4. (S) : Ref)22 메시지 수신 5. (S) : Ref)21 Msg queue에서 응답이 True인 DVM에 주소 요청 Msg 발신 6. (S) : Ref)22 메시지 수신 7. (S) : Message Queue에서 수신된 address를 출력
Alternative Courses of Events	N/A
Exceptional Courses of Events	E5.1 선택한 제품을 보유한 자판기가 없음 1. 해당 메시지 출력 후 메인 화면으로 복귀

Use Case	6. 구매 가능한 자판기 안내
Actor	Customer
Purpose	사용자에게 구매 가능한 자판기 안내
Overview	사용자가 선택한 음료의 재고가 있는 타 DVM을 안내한다.
Type	Evident
Cross Reference	R1.3, R4.5.1
Pre-Requisites	사용자가 재고가 부족한 제품 안내를 받은 상황
Typical Courses of Events	(A) : Actor (S) : System 1. (A) : 사용자가 구매 가능한 자판기 안내 버튼을 입력 2. (S) : 저장된 Title을 Message객체에 저장. 3. (S) : Ref)21 메시지 발신을 이용해 broadcast Msg 발신 4. (S) : Ref)22 메시지 수신 5. (S) : Ref)21 Msg queue에서 응답이 True인 DVM에 주소 요청 Msg 발신 6. (S) : Ref)22 메시지 수신 7. (S) : Message Queue에서 수신된 address를 출력
Alternative Courses of Events	N/A
Exceptional Courses of Events	E5.1 선택한 제품을 보유한 자판기가 없음 1. 해당 메시지 출력 후 메인 화면으로 복귀

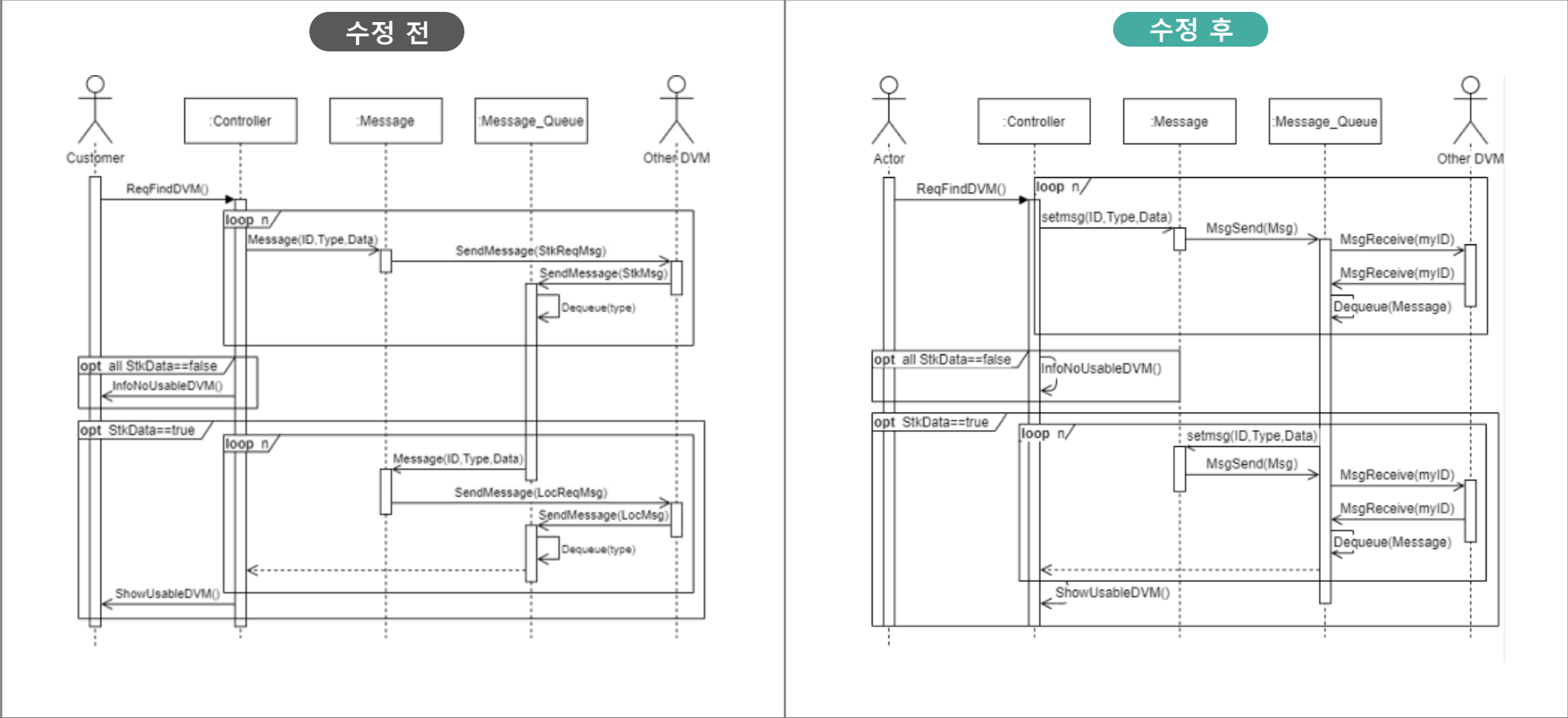
Real Use Cases and Interaction Diagrams

SDS-5

10

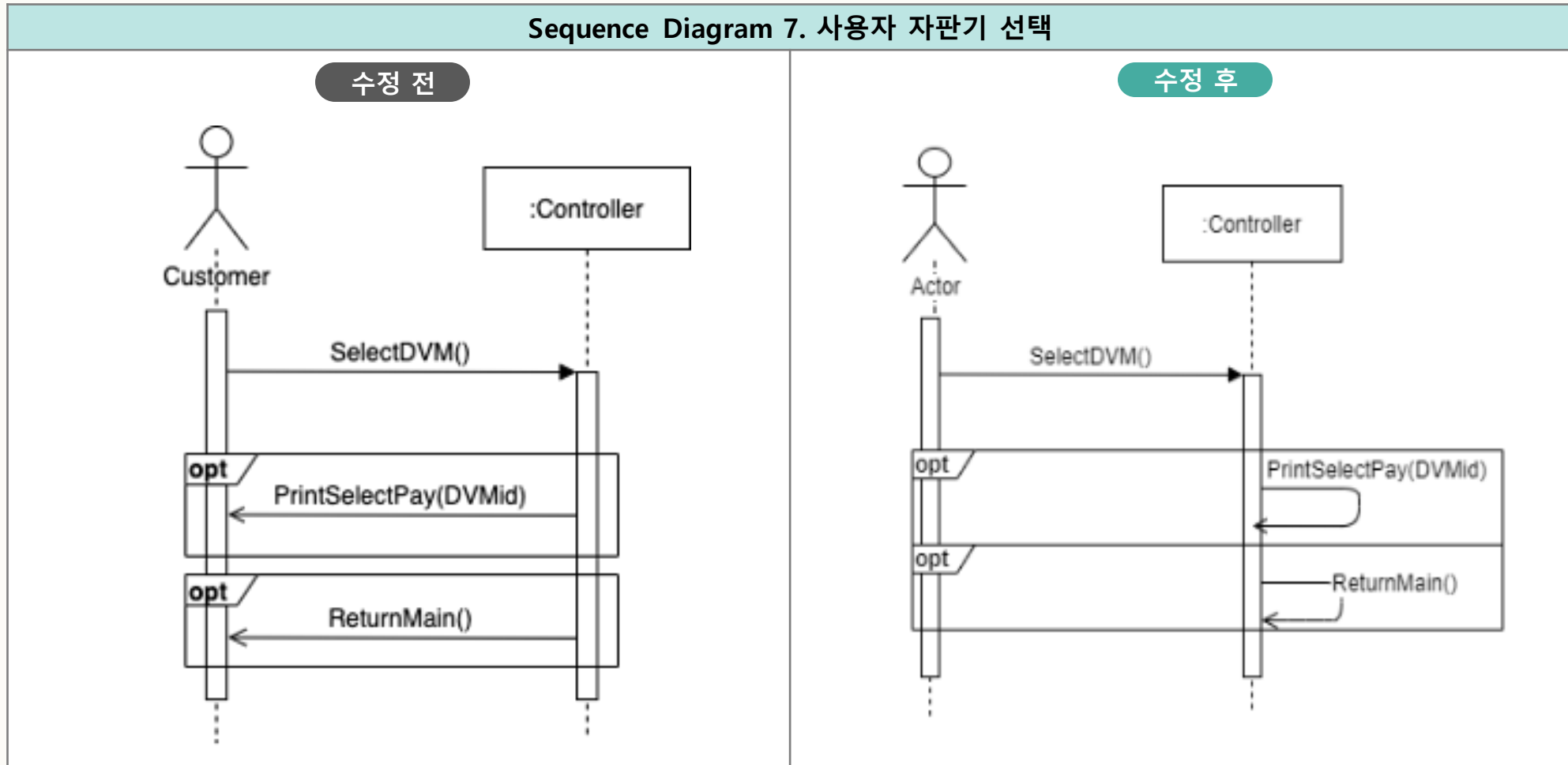
Class Diagram을 보면 SendMessage() operation은 Message Queue가 가지고 있는것으로 되어 있는데, 이는 작성된 Sequence Diagram과 다름.

Sequence Diagram 6. 구매 가능한 자판기 안내

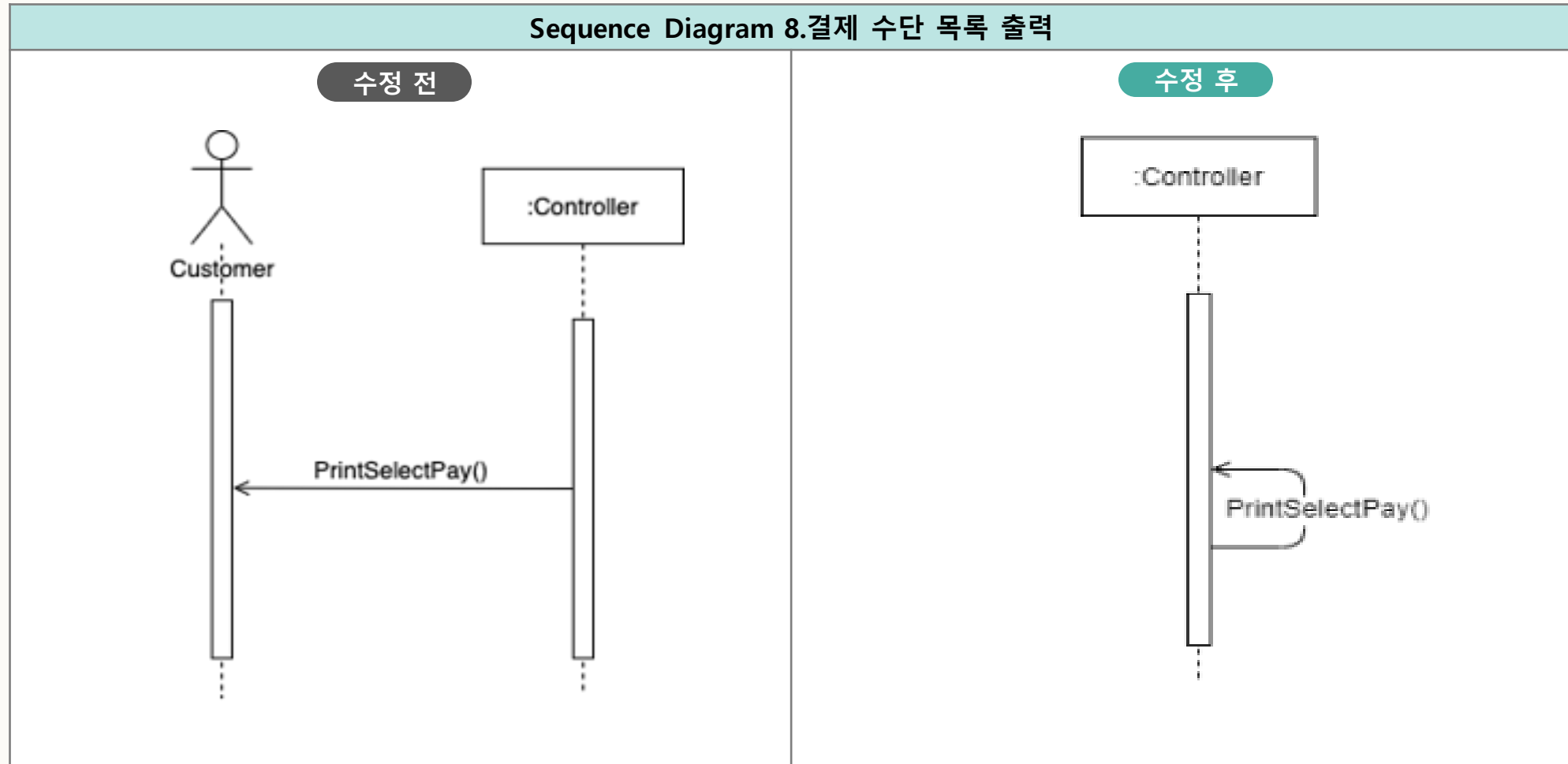


Real Use Cases and Interaction Diagrams

Sequence Diagram 7. 사용자 자판기 선택

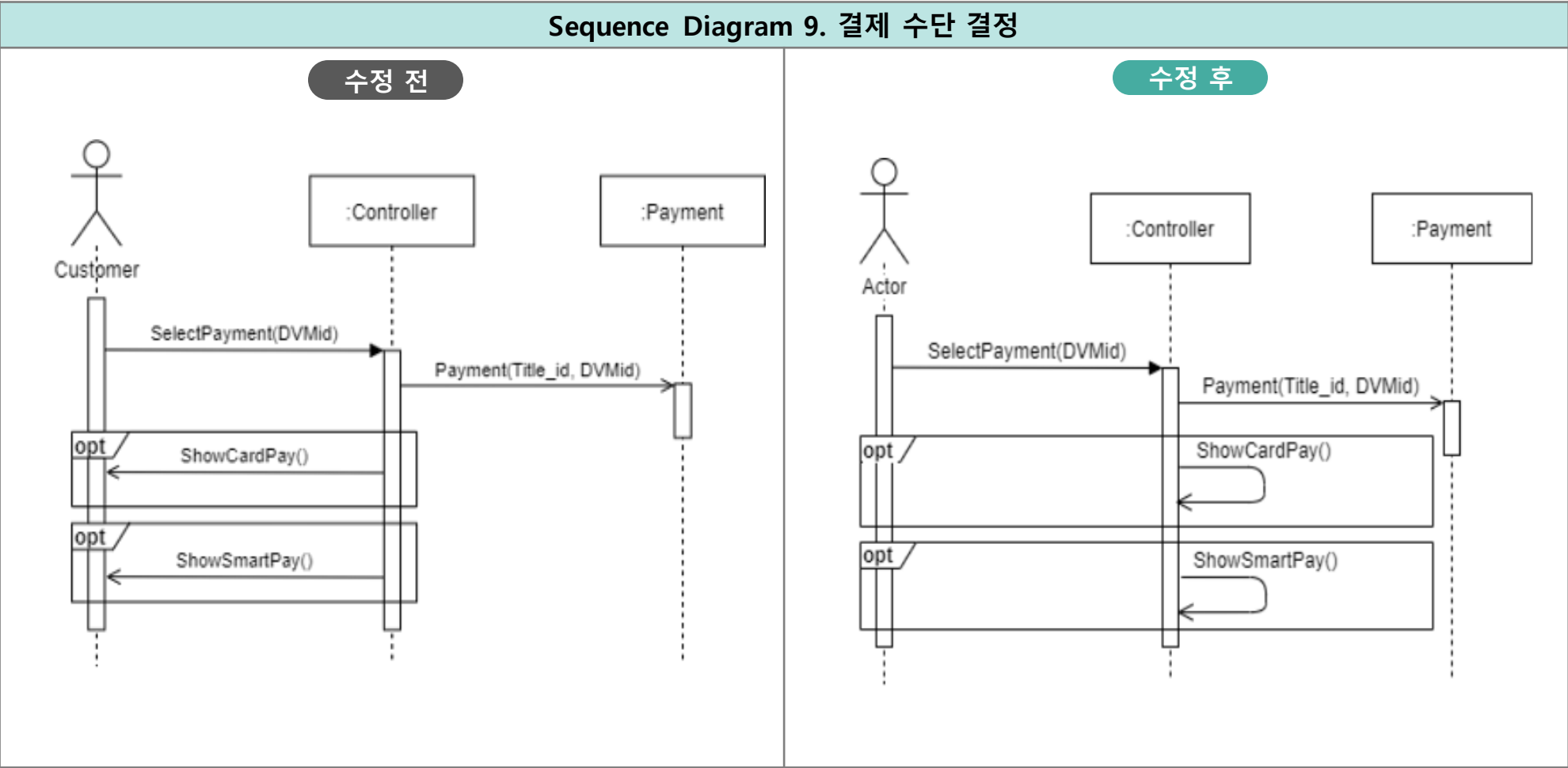


Real Use Cases and Interaction Diagrams



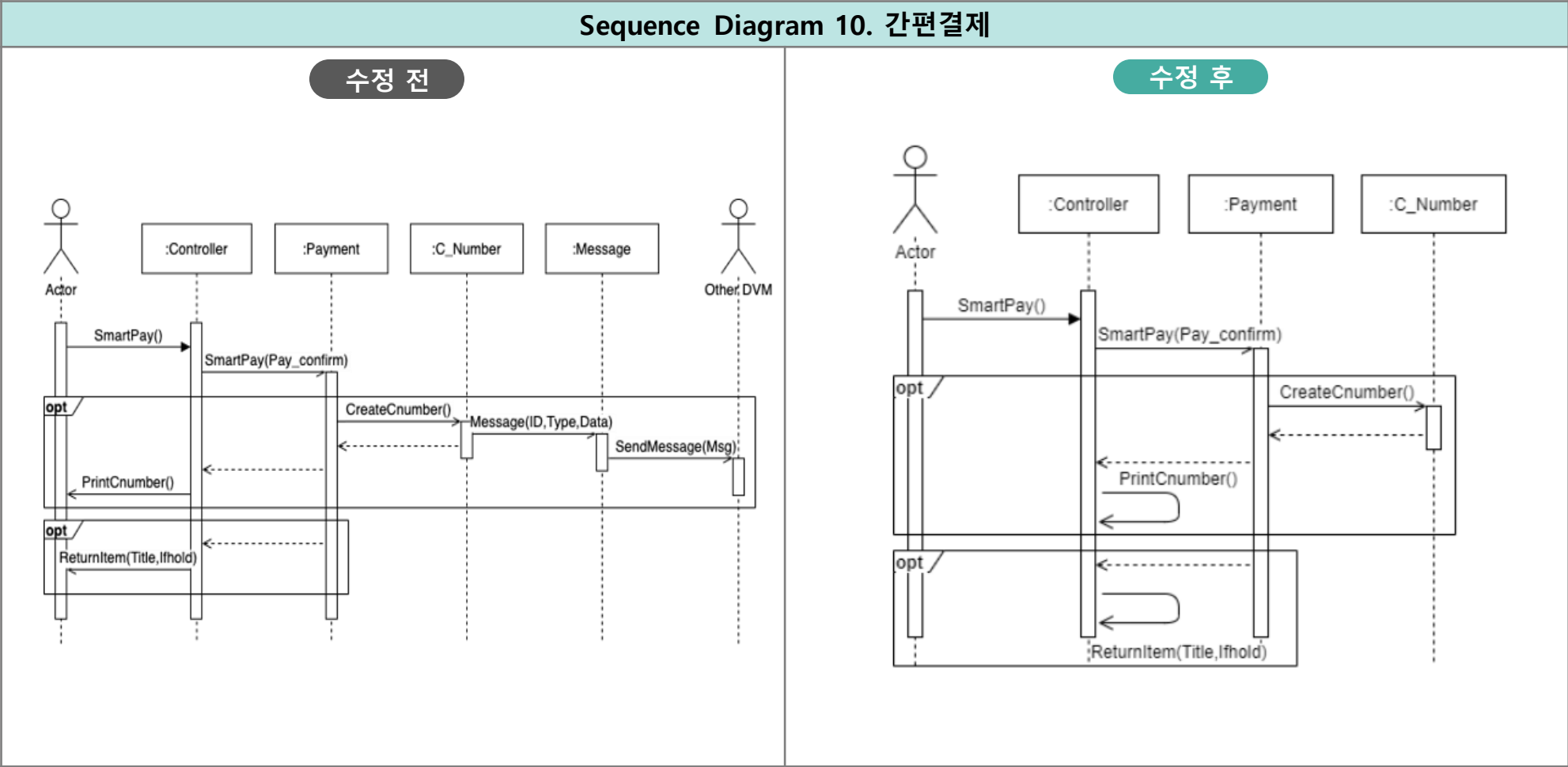
Real Use Cases and Interaction Diagrams

Sequence Diagram 9. 결제 수단 결정



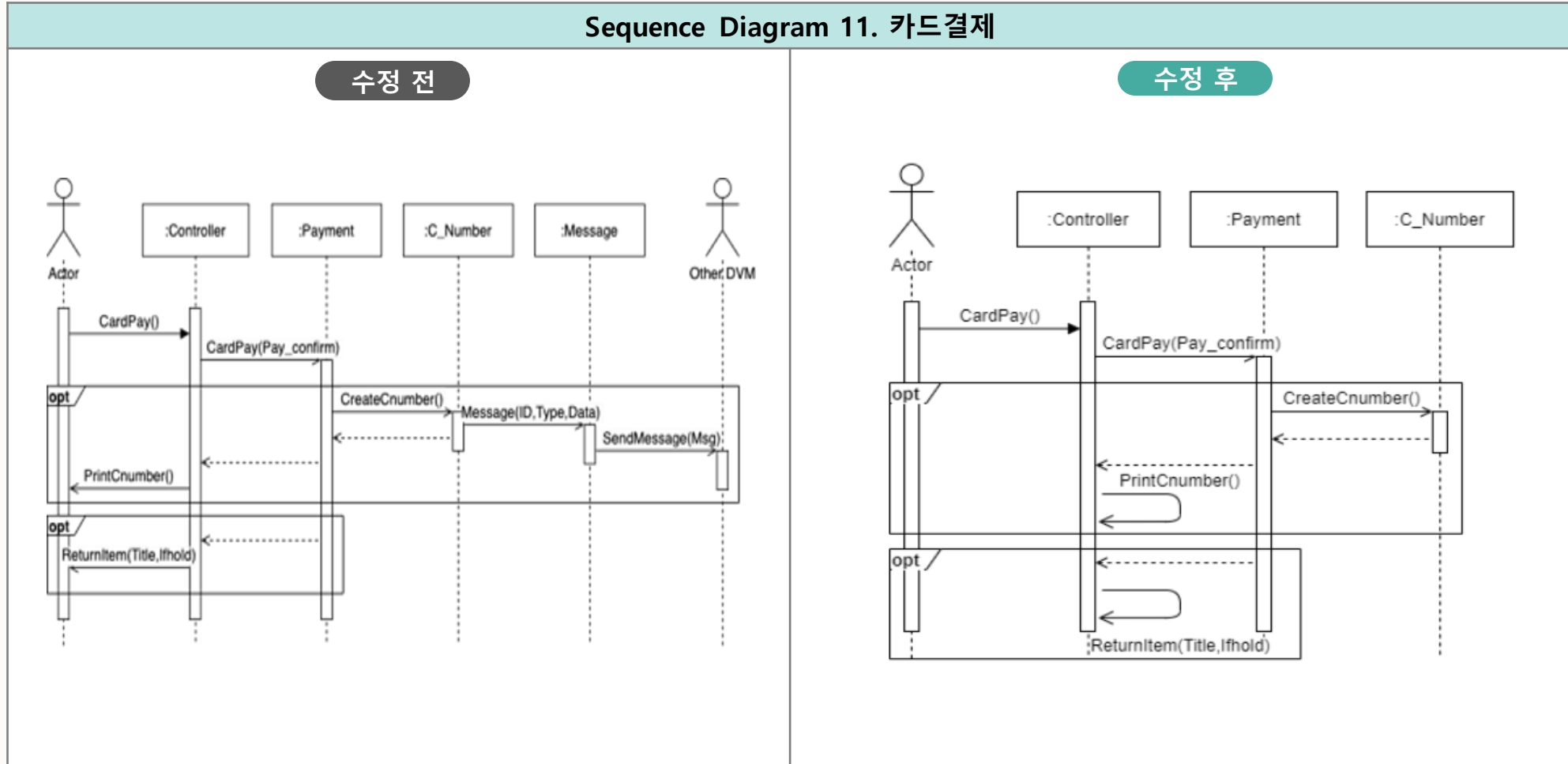
Real Use Cases and Interaction Diagrams

Sequence Diagram 10. 간편결제



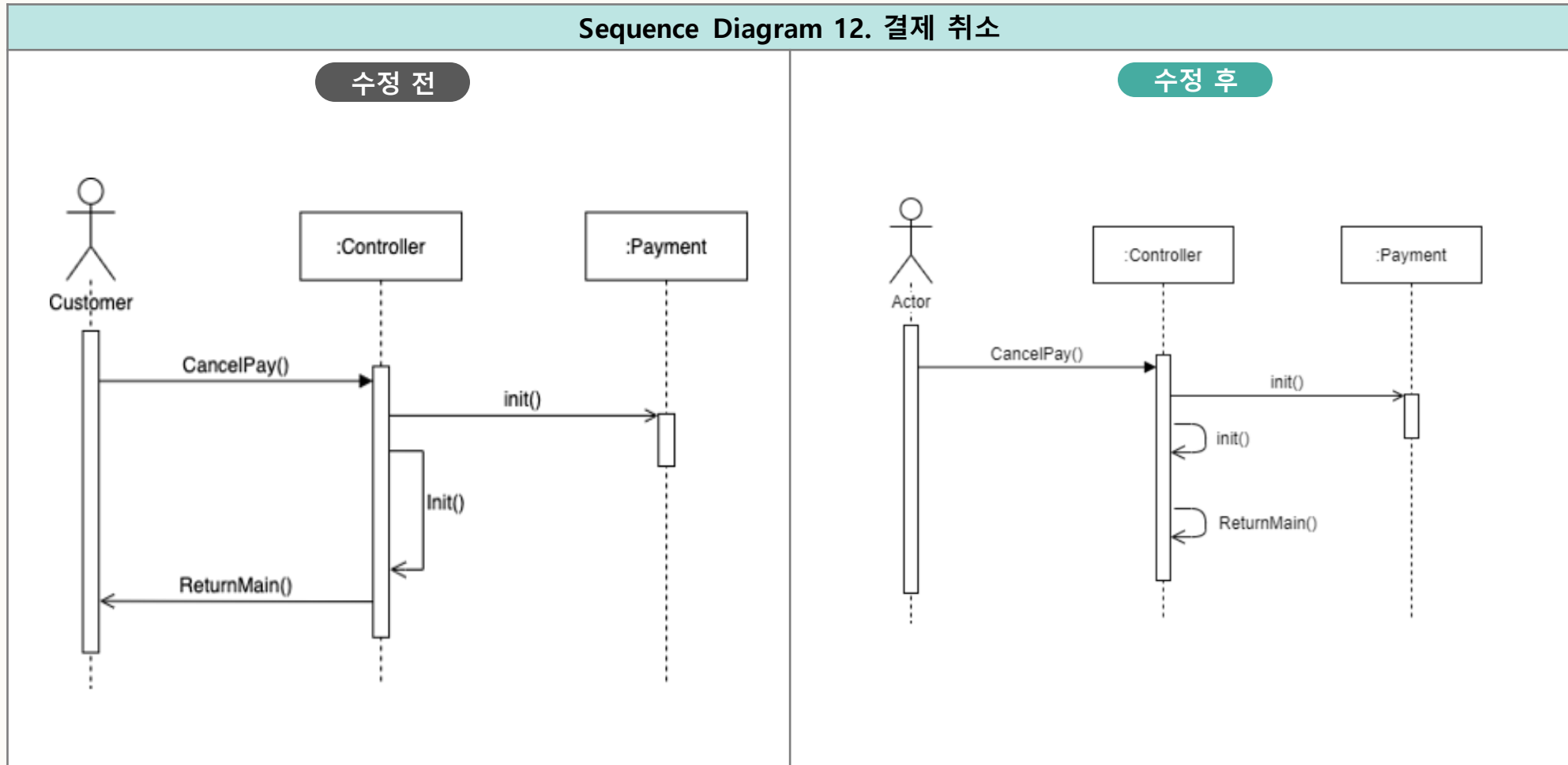
Real Use Cases and Interaction Diagrams

Sequence Diagram 11. 카드결제



Real Use Cases and Interaction Diagrams

Sequence Diagram 12. 결제 취소



Real Use Cases and Interaction Diagrams

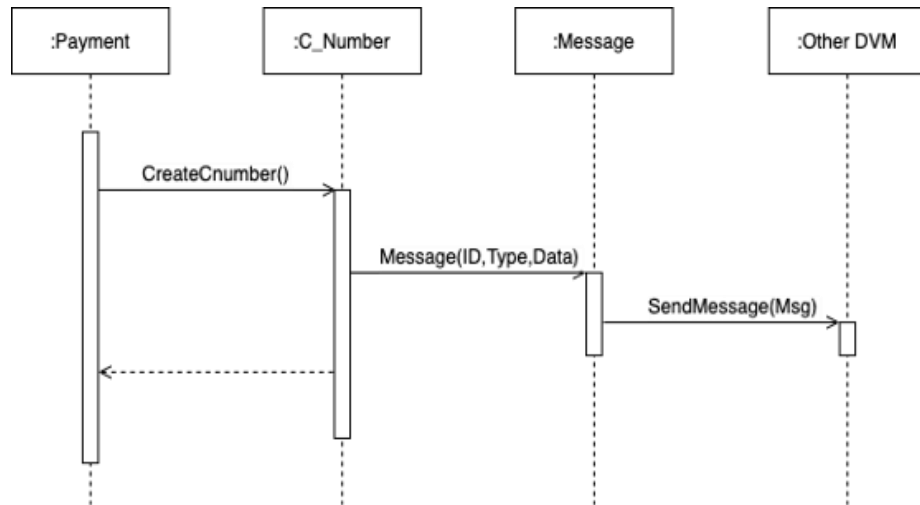
SDS-7

14

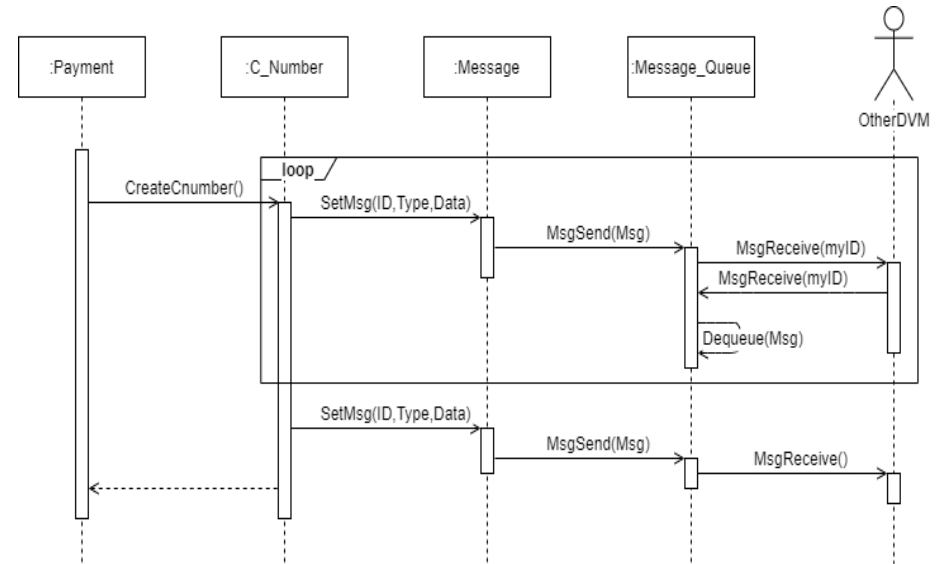
Message Queue Class에 대한 내용이 누락되었음.

Sequence Diagram 13.인증번호 생성

수정 전

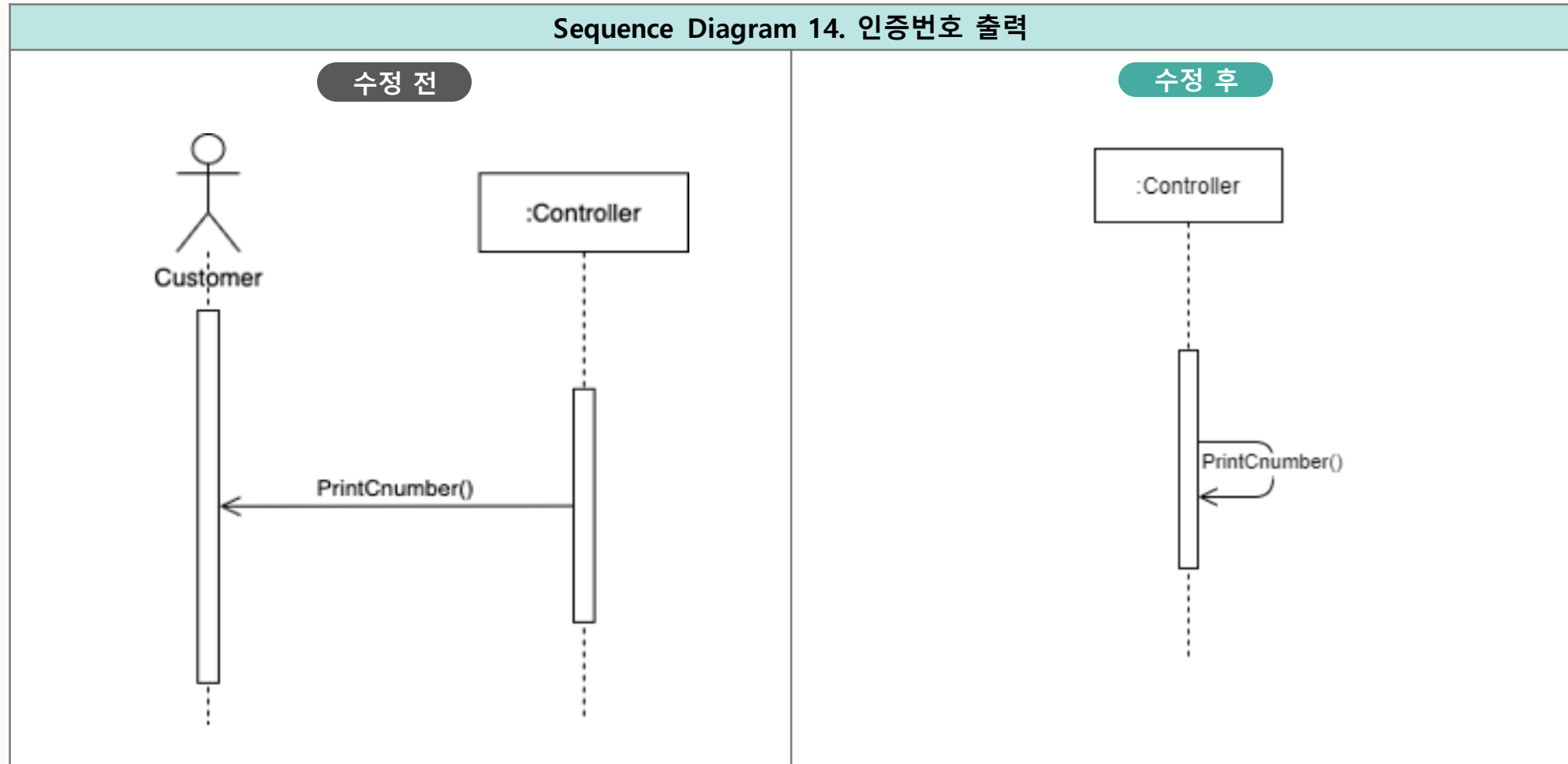


수정 후



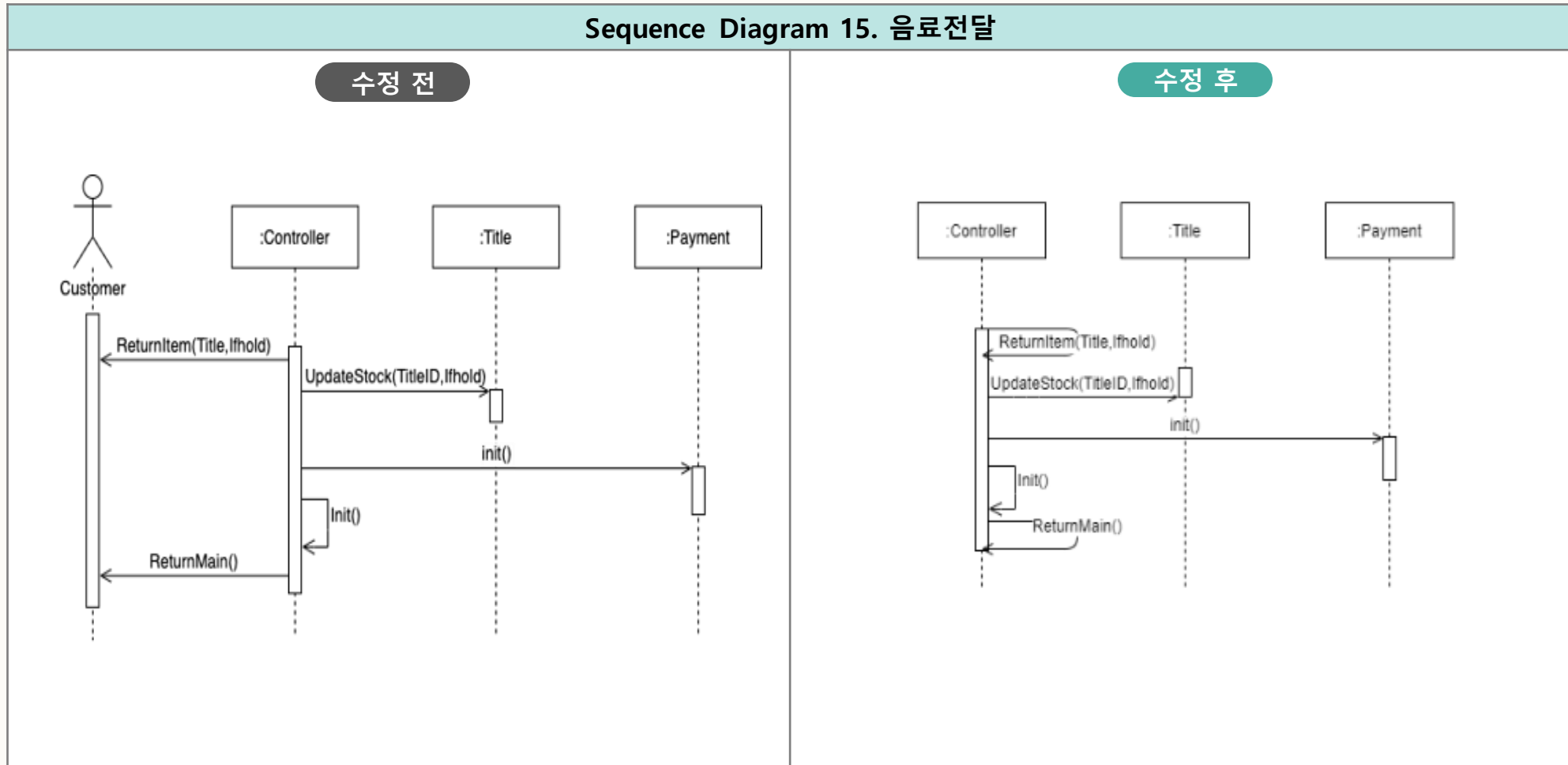
Real Use Cases and Interaction Diagrams

Sequence Diagram 14. 인증번호 출력



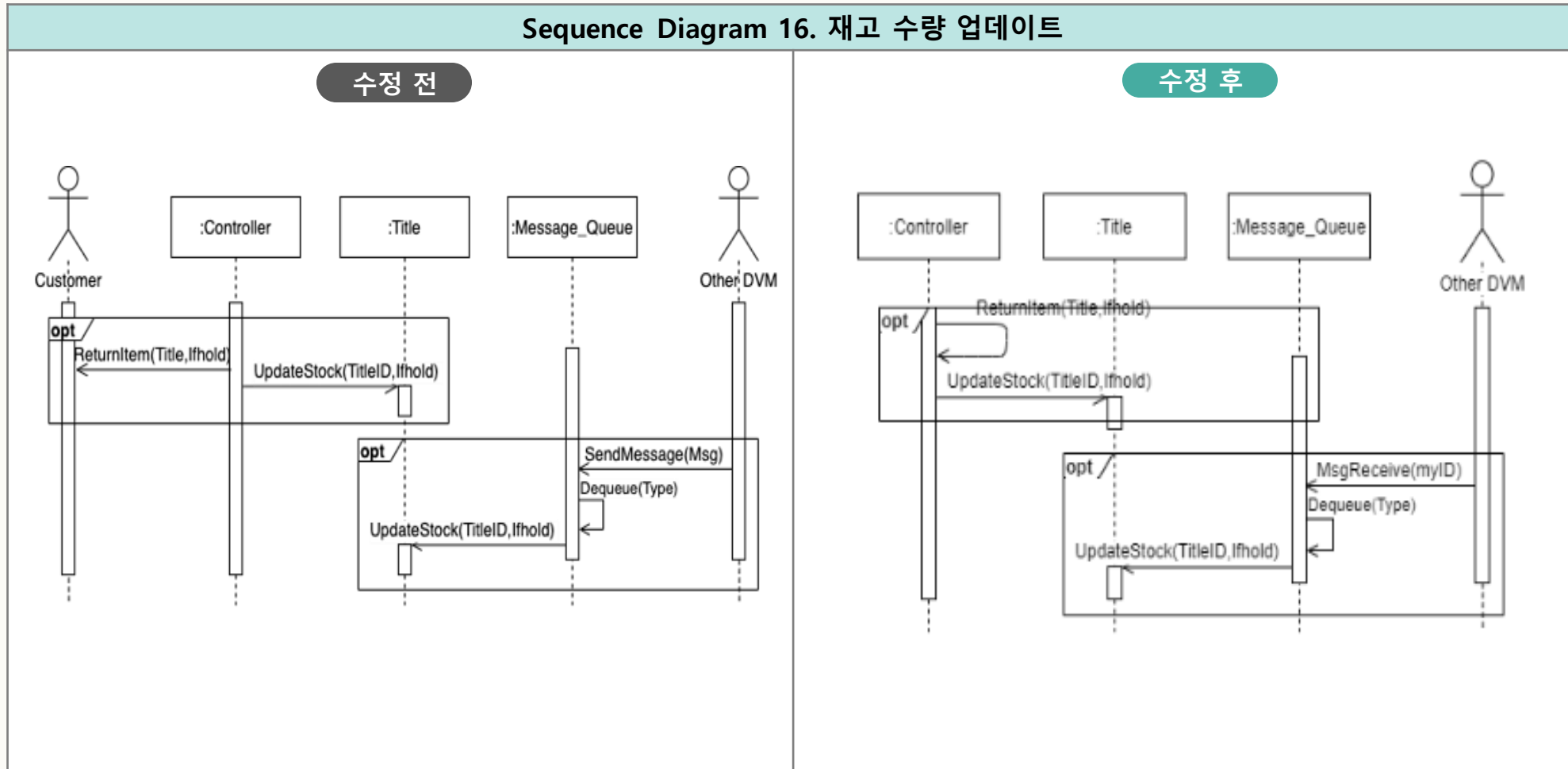
Real Use Cases and Interaction Diagrams

Sequence Diagram 15. 음료전달



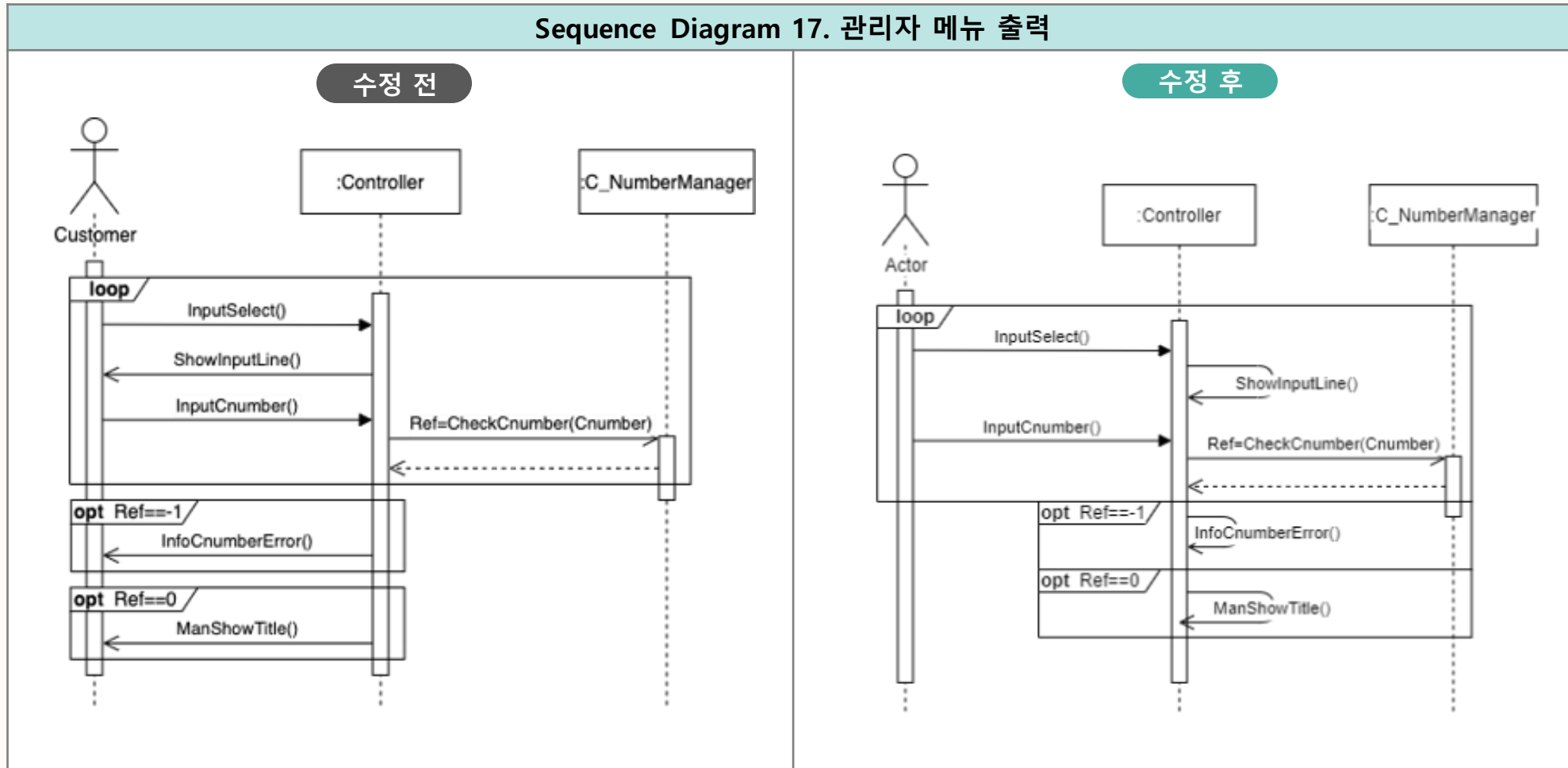
Real Use Cases and Interaction Diagrams

Sequence Diagram 16. 재고 수량 업데이트



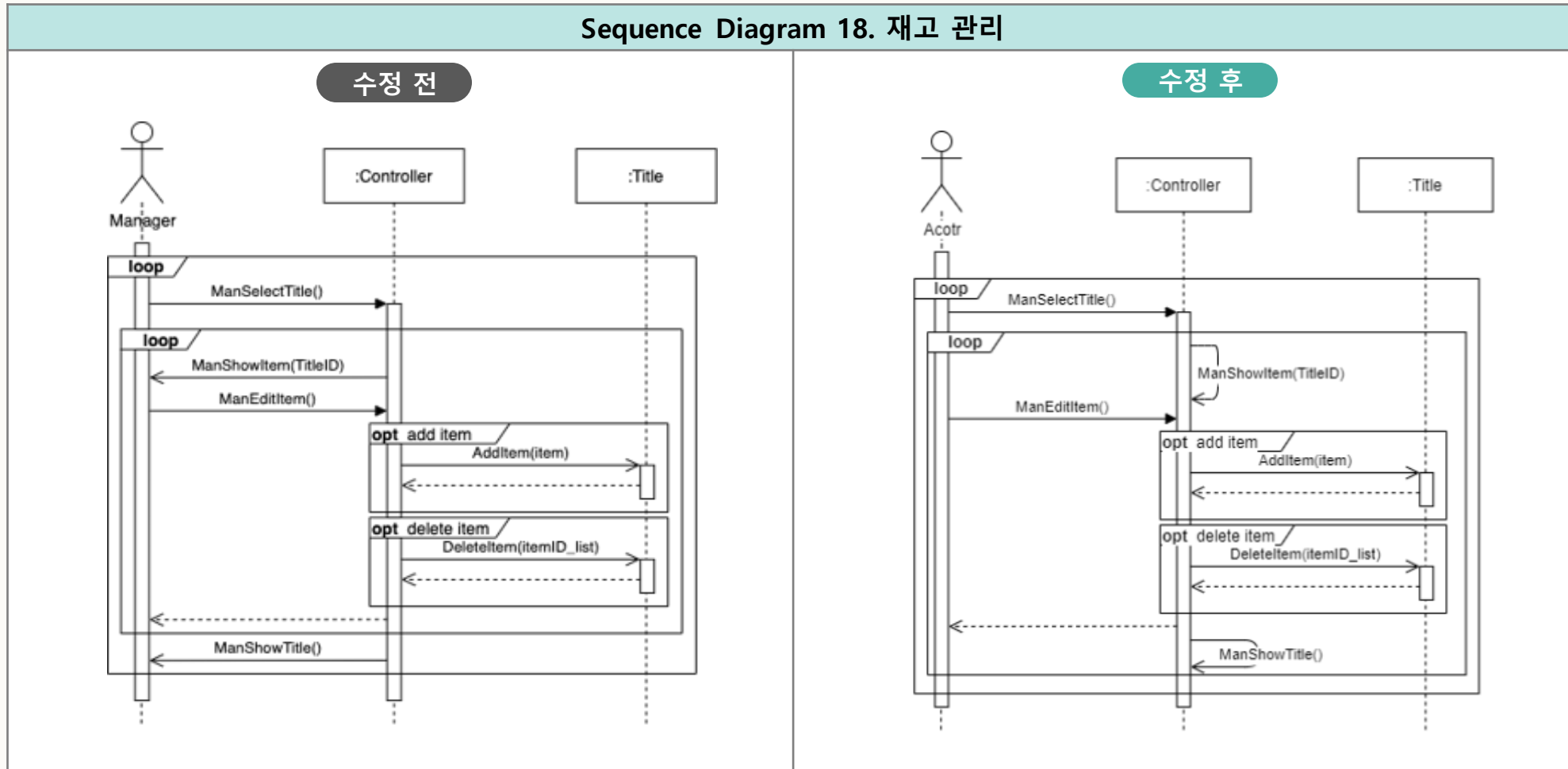
Real Use Cases and Interaction Diagrams

Sequence Diagram 17. 관리자 메뉴 출력



Real Use Cases and Interaction Diagrams

Sequence Diagram 18. 재고 관리



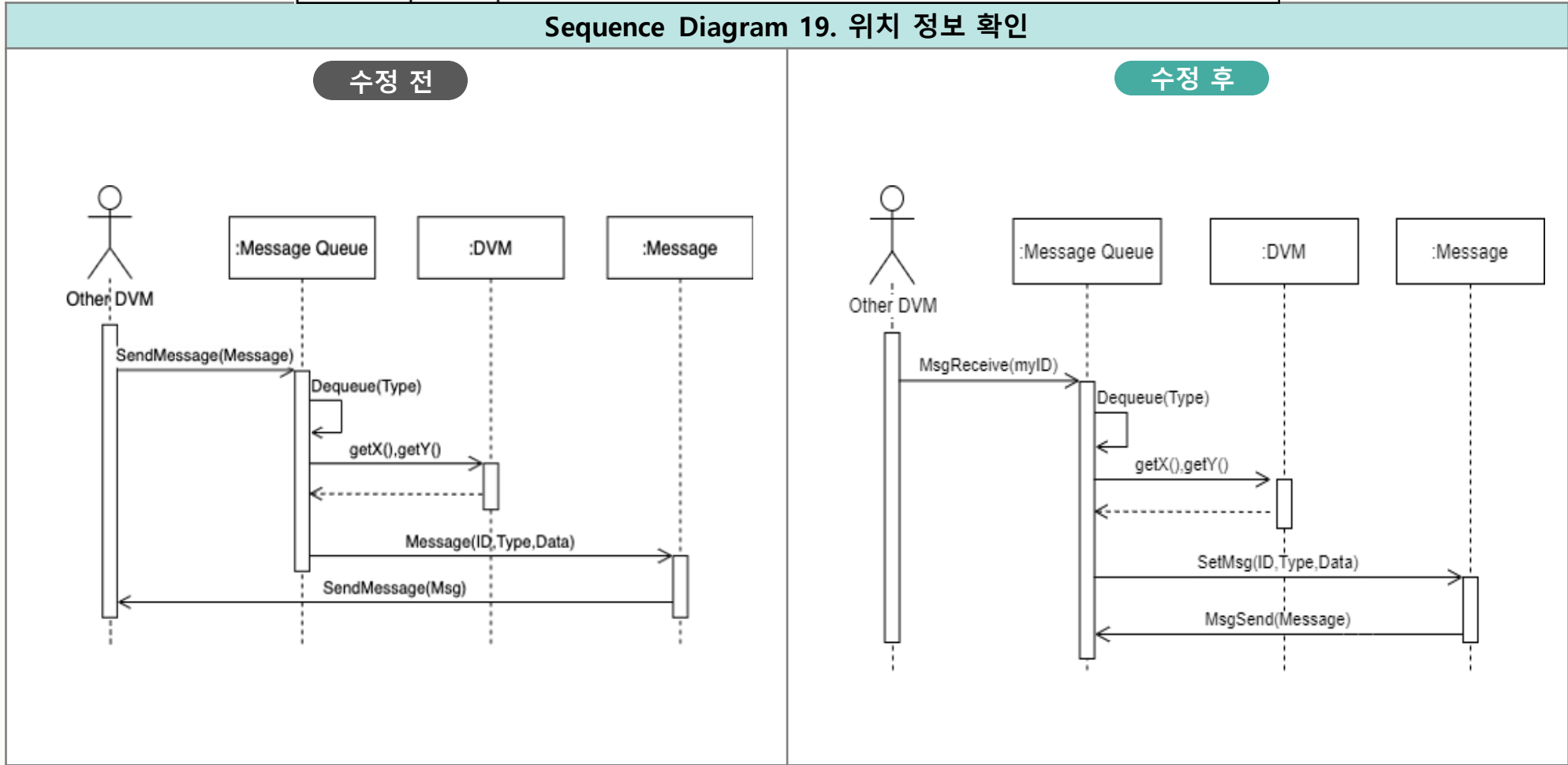
Real Use Cases and Interaction Diagrams

SDS-13

27

Class Diagram에서 Message Class의 operation이 실제 구현과 다름.

Sequence Diagram 19. 위치 정보 확인



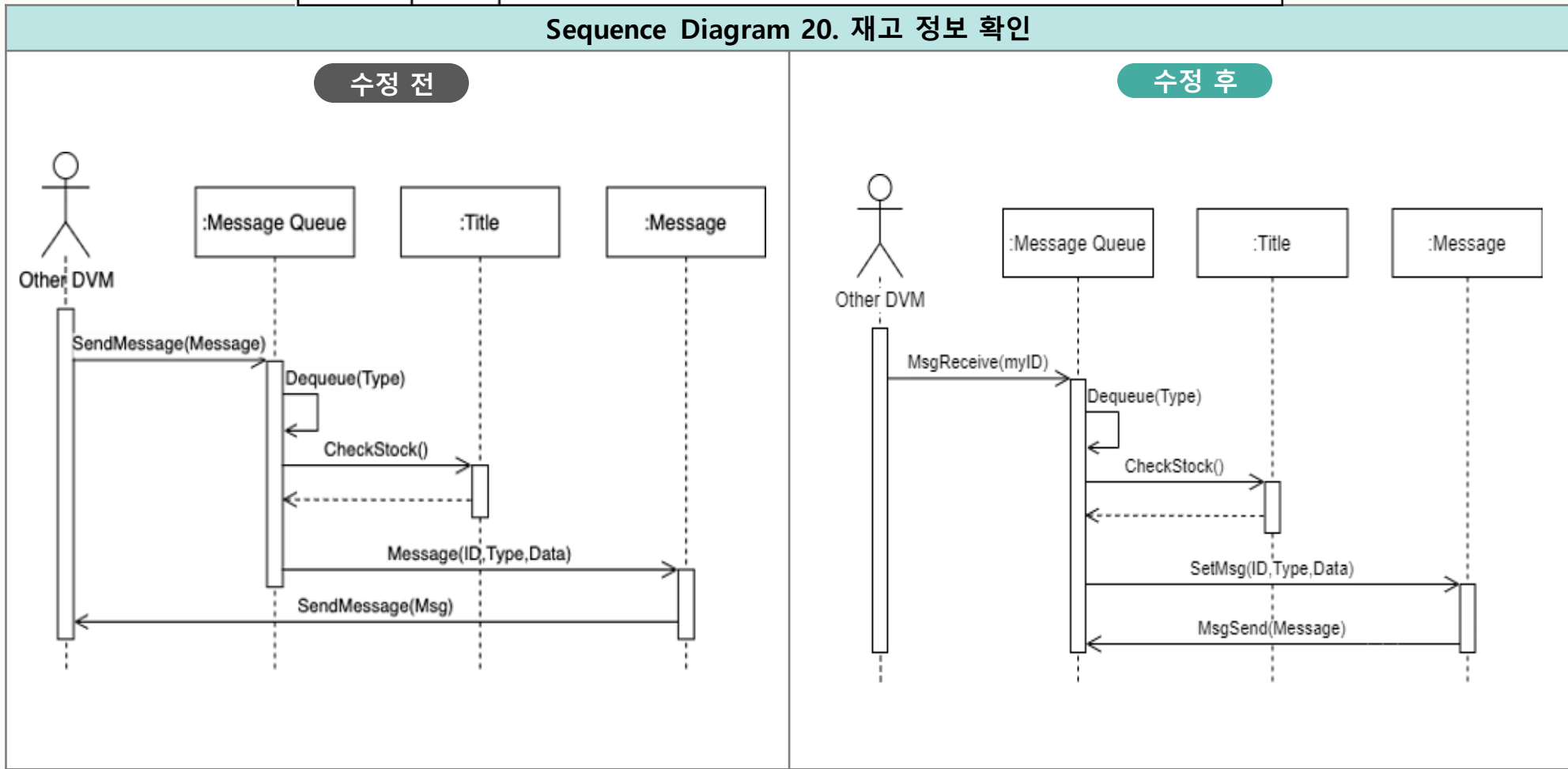
Real Use Cases and Interaction Diagrams

SDS-13

27

Class Diagram에서 Message Class의 operation이 실제 구현과 다름.

Sequence Diagram 20. 재고 정보 확인



Real Use Cases and Interaction Diagrams

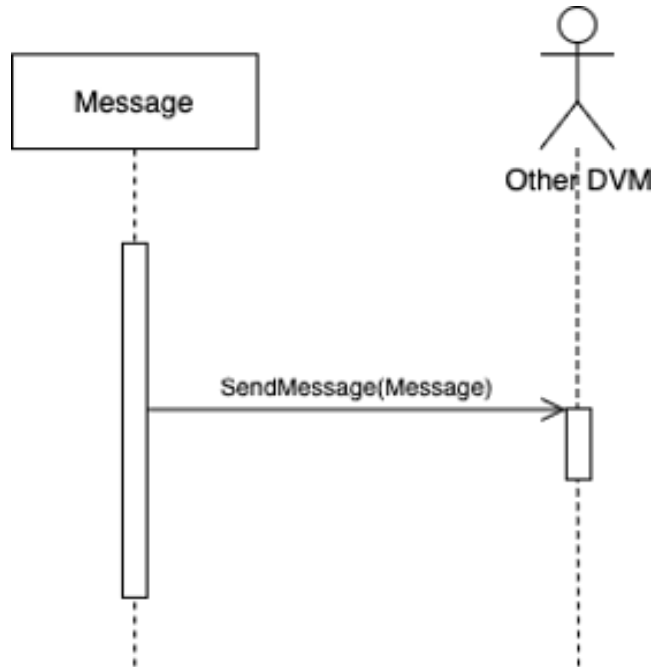
SDS-9

17

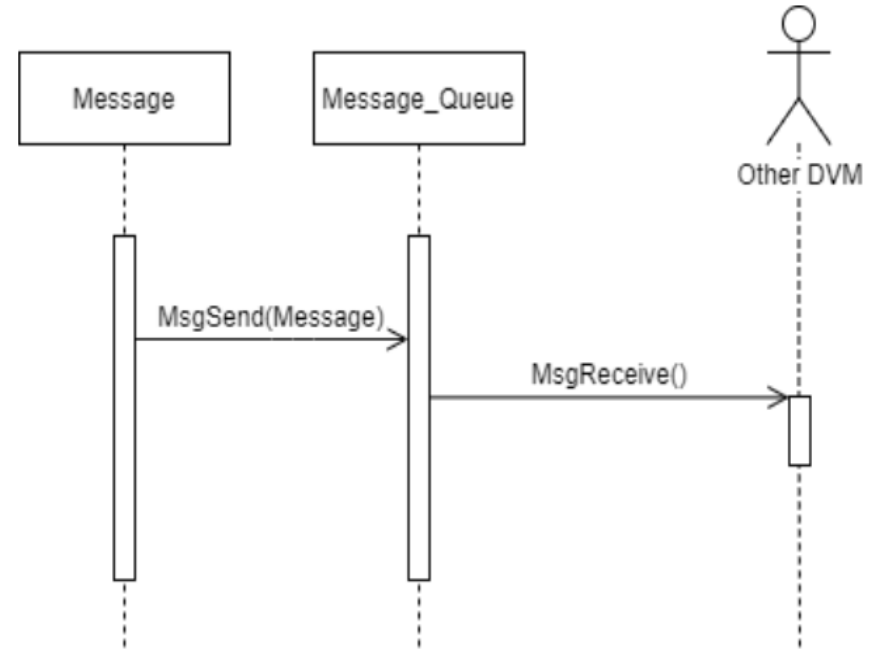
Message Queue Class에 대한 내용이 누락되었음.

Sequence Diagram 21. 메시지 발신

수정 전



수정 후



Activity 2051. Implement Class & Methods Definitions

재고 유통기한 관련 테스트 실패

3	유통 5월이 더 수 (5054년 5월)	9회
5	유통 5월이 더 수	9회

수정 전 코드

```
public AddItemMenu() {
    {...}
    //선택창
    Calendar cal = Calendar.getInstance();
    int year = cal.get(Calendar.YEAR);
    String[] year_list = new String[50];
    String[] month_list = new String[12];
    String[] day_list = new String[31];
    for (int i = 0; i < year_list.length; i++) {
        year_list[i] = Integer.toString(i + year + 1);
    }
    for (int i = 0; i < month_list.length; i++) {
        if (i + 1 < 10) {
            month_list[i] = "0".concat(Integer.toString(i + 1));
        } else {
            month_list[i] = Integer.toString(i + 1);
        }
    }
    for (int i = 0; i < day_list.length; i++) {
        if (i + 1 < 10) {
            day_list[i] = "0".concat(Integer.toString(i + 1));
        } else {
            day_list[i] = Integer.toString(i + 1);
        }
    }
    {...}
}
```

수정 후 코드

```
public AddItemMenu() {
    for (int i = 0; i < day_list1.length; i++) {
        if (i + 1 < 10) { day_list1[i] = "0".concat(Integer.toString(i + 1)); }
        else { day_list1[i] = Integer.toString(i + 1); }
    }
    for (int i = 0; i < day_list2.length; i++) {
        if (i + 1 < 10) { day_list2[i] = "0".concat(Integer.toString(i + 1)); }
        else { day_list2[i] = Integer.toString(i + 1); }
    }
    for (int i = 0; i < day_list3.length; i++) {
        if (i + 1 < 10) { day_list3[i] = "0".concat(Integer.toString(i + 1)); }
        else { day_list3[i] = Integer.toString(i + 1); }
    }
    for (int i = 0; i < day_list4.length; i++) {
        if (i + 1 < 10) { day_list4[i] = "0".concat(Integer.toString(i + 1)); }
        else { day_list4[i] = Integer.toString(i + 1); }
    }
    {...}
}

@Override
public void actionPerformed(ActionEvent e) {
    timer.restart();
    if (e.getSource() == Yearselect) {
        int y = Integer.parseInt((String) Yearselect.getSelectedItem());
        int m = Integer.parseInt((String) Monthselect.getSelectedItem());
        if (m == 2) {
            DefaultComboBoxModel<String> model;
            if ((y % 4 == 0 && y % 100 != 0) || y % 400 == 0) {
                model = new DefaultComboBoxModel<>(day_list3);
            } else { model = new DefaultComboBoxModel<>(day_list4); }
            Dayselect.setModel(model);
        }
    }
    if (e.getSource() == Monthselect) {
        timer.restart();
        int y = Integer.parseInt((String) Yearselect.getSelectedItem());
        int m = Integer.parseInt((String) Monthselect.getSelectedItem());
        DefaultComboBoxModel<String> model;
        if (m == 2) {
            if ((y % 4 == 0 && y % 100 != 0) || y % 400 == 0) {
                model = new DefaultComboBoxModel<>(day_list3);
            } else { model = new DefaultComboBoxModel<>(day_list4); }
        } else if (m == 1 || m == 3 || m == 5 || m == 7 || m == 8 || m == 10 || m == 12) {
            model = new DefaultComboBoxModel<>(day_list1);
        } else { model = new DefaultComboBoxModel<>(day_list2); }
        Dayselect.setModel(model);
    }
    {...}
}
```

실행 결과



달 별, 윤년 별 날짜수 정상 출력 확인

Activity 2051. Implement Class & Methods Definitions

재고 추가 관련 테스트 실패

4	유동기한 지난 것의 등록 가능 여부	fail
159	1.12.1.1.2.0.3.0.	F 재고 있는 음료 종류가 7개 초과로 들어가 PFR을 만족하지 않음.

수정 전 코드

```

} else if (del == 1) { //AddItem
    k.setVisible(false);
    k = new AddItemMenu();
    {
        int del2 = -1;
        while (del2 == -1) {
            System.out.print("");
            del2 = ((AddItemMenu) k).return_value;
        }
        if (del2 == -2) {
            return -2;
        } else if (del2 == 0) {
            return 1;
        } else if (del2 == 1) {
            Title_List.get(TitleID - 1).AddItem(new Item(((AddItemMenu) k).return_date));
            return 1;
        } else {
            //error
            return -1;
        }
    }
}
    
```

수정 후 코드

```

} else if (del == 1) { //AddItem
    if (Title_List.get(TitleID - 1).getItem_List().size() >= 30) {
        k.setVisible(false);
        k = new InfoUI("더 이상 재고를 추가할 수 없습니다.", "이전");

        int del4 = -1;
        while (del4 == -1) {
            return 1;
        }
    }
    int count = 0;
    for (int i = 0; i < 20; i++) {
        if (Title_List.get(i).CheckStock() > 7) count++;
    }
    if (count >= 7 && Title_List.get(TitleID - 1).CheckStock() == false) {
        k.setVisible(false);
        k = new InfoUI("음료를 7종류 이상 추가할 수 없습니다.", "이전");

        int del4 = -1;
        while (del4 == -1) {
            return 1;
        }
    }
    k.setVisible(false);
    k = new AddItemMenu();
    int del2 = -1;
    while (del2 == -1) {
        System.out.print("");
        if (((AddItemMenu) k).getReturn_value() == 1) {
            Calendar cal = Calendar.getInstance();
            int today = (cal.get(Calendar.YEAR) + 10000) +
                ((cal.get(Calendar.MONTH) + 1) * 100) + cal.get(Calendar.DATE);
            if (today > ((AddItemMenu) k).getReturn_date()) {
                k.setVisible(false);
                k = new InfoUI("유동기한이 지난 재고를 추가할 수 없습니다.", "이전");
                int del3 = -1;
                while (del3 == -1) {
                    return 1;
                }
            }
        }
        del2 = ((AddItemMenu) k).getReturn_value();
    }
    if (del2 == -2) {
        return -2;
    } else if (del2 == 0) {
        return 1;
    } else if (del2 == 1) {
        Title_List.get(TitleID - 1).AddItem(new Item(((AddItemMenu) k).return_date));
        return 1;
    } else {
        //error
        return -1;
    }
}
} else if (del == 2) { //delete item
    }
    
```

실행 결과



재고 추가 제한 확인

Activity 2051. Implement Class & Methods Definitions

자판기 3대 이상일 경우 선결제 관련 테스트 실패

3.1.1.6. Use case 6 - 구매 가능한 자판기 안내

회수	1	2	3	4	5	6	7	8	9	10	평균
결과	P	P	P	P	P	F	F	F	F	F	5/10

DVM이 2대 이하 있을 때는 성공하지만, DVM이 3대 이하 있을 때 실패했다.

3.1.1.7. Use case 7 - 사용자 자판기 선택

회수	1	2	3	4	5	6	7	8	9	10	평균
결과	P	P	P	P	P	P	P	F	F	F	3/10

DVM이 3대 이하 있을 때는 OK가 아닌 DVM이 정거고, 이 DVM은 선택이 불가능하다.

3.1.1.11. Use case 11 - 카드 결제

회수	1	2	3	4	5	6	7	8	9	10	평균
결과	P	P	P	P	P	P	P	P	P	P	10/10
Alter	P	P	P	P	P	F	F	F	F	F	5/10

3대의 자판기가 있는 상황에서 다른 자판기 재고 검색이 되지 않아 테스트가 불가능하다.

3.1.1.19. Use case 19 - 위치 정보 확인

회수	1	2	3	4	5	6	7	8	9	10	평균
결과	P	P	P	P	P	F	F	F	F	F	5/10

3대의 자판기가 있는 상황에서 다른 자판기 위치 정보 확인이 되지 않음

3.1.1.20. Use case 20 - 재고 정보 확인

회수	1	2	3	4	5	6	7	8	9	10	평균
결과	P	P	P	P	P	F	F	F	F	F	5/10

3대의 자판기가 있는 상황에서 다른 자판기 재고 정보 확인이 되지 않음

수정 전 코드

```
public void MsgRecv(int myid) {
    Socket socket = null; //Client와 통신하기 위한 Socket
    ServerSocket server_socket = null; //서버 생성을 위한 ServerSocket
    BufferedReader in = null; //Client로부터 데이터를 읽어들이기 위한 입력스트림
    PrintWriter out = null; //Client로 데이터를 보내내기 위한 출력 스트림
    int port = myid + 50000;
    ObjectInputStream objectInputStream; // 직렬화된 객체를 읽어올때 사용
    PrintWriter printWriter; // 같은 전달받아 사용
    Message message = new Message(myid, -1);

    try {...} catch (IOException e) {...}
    try {...} catch (IOException e) {...} finally {...}
}

public static void MsgSend(Message message) {
    Socket socket = null; //Server와 통신하기 위한 Socket
    BufferedReader in = null; //Server로부터 데이터를 읽어들이기 위한 입력스트림
    PrintWriter out = null; //서버로 보내내기 위한 출력 스트림
    InetAddress ia = null;
    int port = message.getTargetID() + 50000;
    try
    //서버로 접속하고 인풋스트림을 지정하는 부분
    {...} catch (IOException e) {
        System.err.println("서버 접속 오류, 오류 DVM : " + message.getMyID());
        if (message.getType() == 1) {
            stk--;
            Dequeue();
        }
    } finally {...}
}

public static void Dequeue() {
    while (msgQueue.size() > 0) {...}
    if (StkmsgQueue.size() == stk) {
        loc = 0;
        while (StkmsgQueue.size() > 0) {
            Message stk = StkmsgQueue.poll();
            if (stk.isBoolData()) {
                Message sm = new Message(CurrentID);
                sm.setmsg(stk.getMyID(), "None 3");
                loc++;
            }
        }
        System.out.println("위치 요청 메시지 전송완료");
        stk = 0;
    }
    if (LocmsgQueue.size() == loc) {...}
}
```

Activity 2051. Implement Class & Methods Definitions

수정 후 코드

```
public void MyHttpd(int myid) {
    Socket socket = null; //Server의 통신하기 위한 Socket
    ServerSocket server_socket = null; //서버 생성을 위한 ServerSocket
    BufferedReader in = null; //Server로부터 데이터를 읽어들이기 위한 입력스트림
    PrintWriter out = null; //Server로 보내기 위한 출력 스트림
    int port = myid + 50000;
    ObjectInputStream objectInputStream; // 직렬화된 객체를 읽어오는데 사용
    PrintWriter printWriter; // 공통 인터페이스 사용

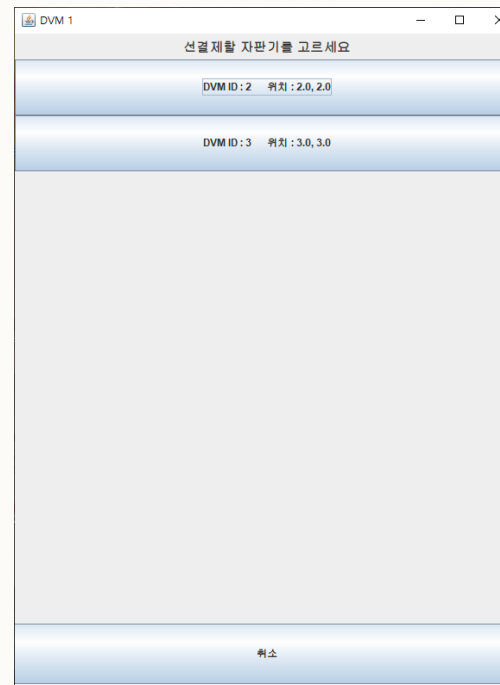
    try {
        catch (IOException e) {
            try {
                while (true) {
                    if (this.isInterrupted()) break;
                    socket = server_socket.accept(); //서버 오픈, 클라이언트 접속 대기.
                    printWriter = new PrintWriter(new BufferedWriter(new OutputStreamWriter(socket.getOutputStream())));
                    in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
                    String msg = in.readLine();
                    String[] temp = msg.split(" ");
                    //DVM ID를 받기
                    Message message = new Message(msg, -1);
                }
            } catch (IOException e) {
                finally {
                }
            }
        }
    }

    public static void MySend(Message message) {
        Socket socket = null; //Server의 통신하기 위한 Socket
        BufferedReader in = null; //Server로부터 데이터를 읽어들이기 위한 입력스트림
        PrintWriter out = null; //Server로 보내기 위한 출력 스트림
        InetAddress ip = null;
        int port = message.getTargetID() + 50000;
        try {
            //서버로 접속하고 인포스트리밍을 지정하는 부분
            catch (IOException e) {
                System.err.println("서버 접속 오류, 오류 DVM " + message.getTargetID());
                if (message.getType() == 1) {
                    stk--;
                    if (stk == StkmsgQueue.size()) {
                        Dequeue();
                    }
                }
                if (message.getType() == 0) {
                }
            }
        } finally {
        }
    }

    public static void Dequeue() {
        int result = -1;
        while (msgQueue.size() > 0) {
            if (StkmsgQueue.size() == stk) {
                int i = 0;
                while (StkmsgQueue.size() > 0) {
                    Message stk = StkmsgQueue.poll();
                    if (stk.isBufferData()) {
                        Message sm = new Message(DVM.getCurrentID());
                        sm.setmsg(stk.getMyID(), msg, 4);
                        System.out.println("여기 요청 메시지 전송 완료");
                        i++;
                    }
                }
                loc = i;
                stk = 0;
            }
            if (LocmsgQueue.size() == loc) {
            }
            if (CmsgQueue.size() == cnum) {
            }
        }
    }
}
```



실행 결과



재고가 있는 자판기 출력 확인, 결제 성공 확인

Activity 2051. Implement Class & Methods Definitions

QR 코드 출력 관련 테스트 실패

3.1.1.10. Use case 10 - 간편 결제

회수	1	2	3	4	5	6	7	8	9	10	평균
결과	F	F	F	F	F	F	F	F	F	F	0/10

시스템이 화면에 QR코드를 출력하지 않아 테스트가 불가능하다.

수정 전 코드

```
public SmartPayUI() {
    this.addKeyListener(new KeyListener() {...});
    timer.start();
    this.setPreferredSize(new Dimension( width: 600, height: 800));
    this.setTitle("DVM");

    //라벨패널
    {...}
    //안내패널
    JPanel informpanel = new JPanel();
    //informpanel.setPreferredSize(new Dimension(600,300));
    infolabel = new JLabel( text: "간편결제");
    infolabel.setFont(infolabel.getFont().deriveFont(20.0f));
    informpanel.add(infolabel);
    //버튼패널
    {...}
}
```

수정 후 코드

```
private Timer timer = new Timer( delay: 1000, new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if (s <= 10) {
            {...}
            if (s == 10) {
                informpanel.remove(imageLabel);
                informpanel.updateUI();
                labelpanel.setPreferredSize(new Dimension( width: 600, height: 300));
                infolabel.setFont(infolabel.getFont().deriveFont(20.0f));
                informpanel.add(infolabel);
            }
            if (s == 0) {...}
        } else {...}
        s--;
    }
});

private JButton cancel;
private JLabel label;
private JLabel imageLabel;
private JLabel infolabel = new JLabel();
private ImageIcon img;
private JPanel labelpanel;
private JPanel informpanel;
private JPanel buttonpanel;
private int return_value = -1;

public SmartPayUI() {
    this.addKeyListener(new KeyListener() {...});

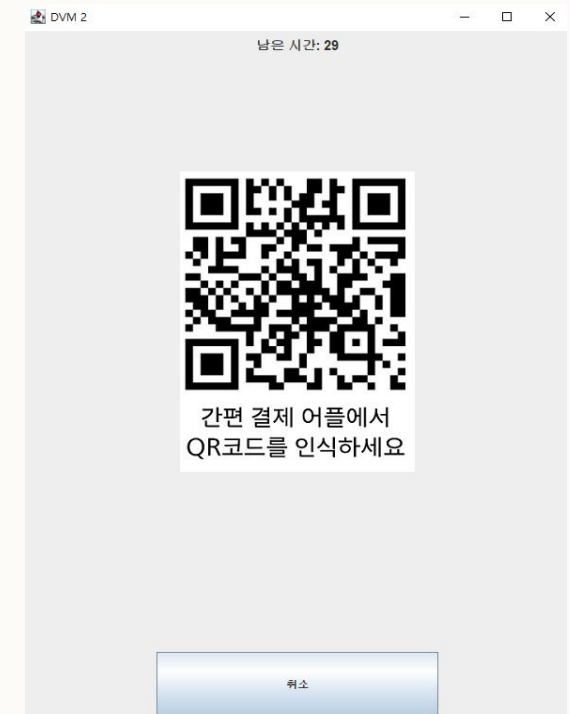
    timer.start();

    this.setPreferredSize(new Dimension( width: 600, height: 800));
    this.setTitle("DVM " + DVM.getCurrentID());

    //라벨패널
    {...}
    //안내패널
    informpanel = new JPanel();
    informpanel.setPreferredSize(new Dimension( width: 600, height: 500));
    img = new ImageIcon( filename: "src/main/java/601/image/QR.jpg");
    imageLabel = new JLabel(img);
    informpanel.add(imageLabel);

    //버튼패널
    {...}
}
```

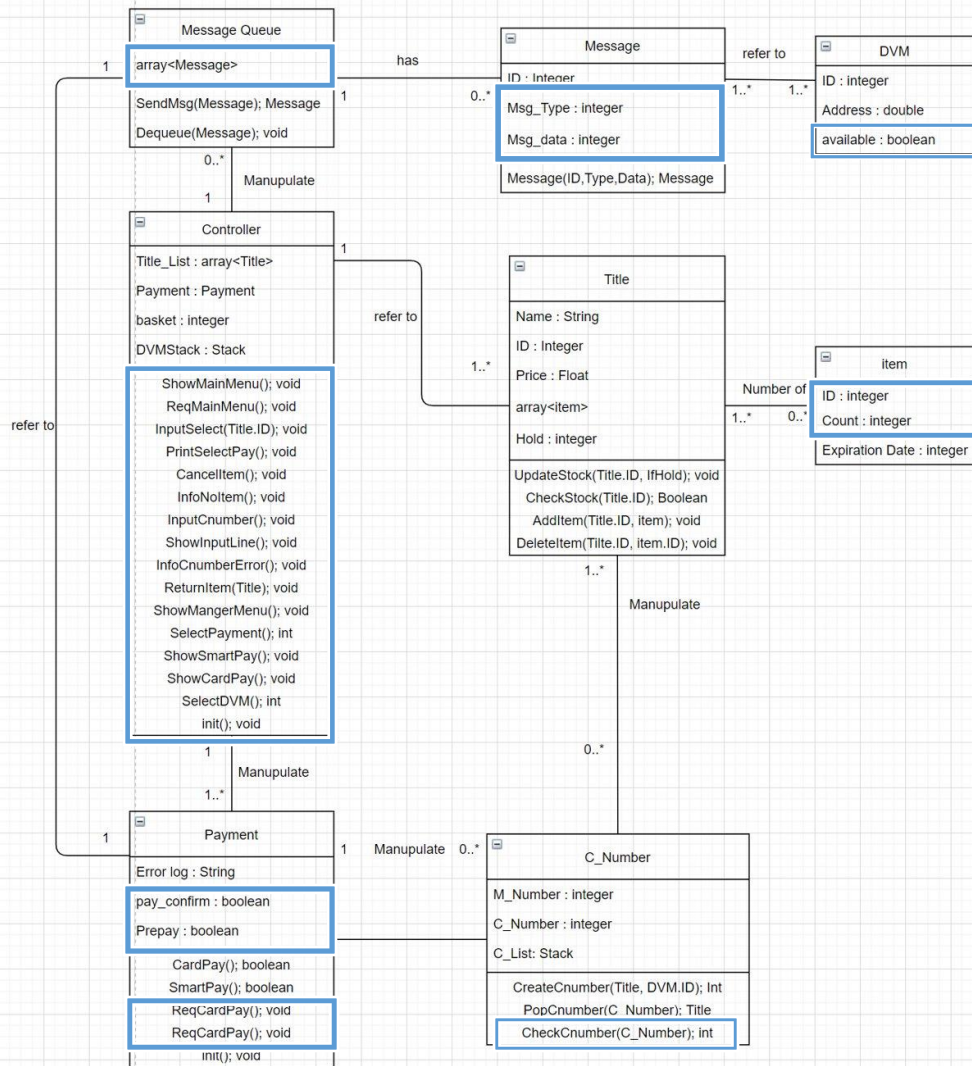
실행 결과



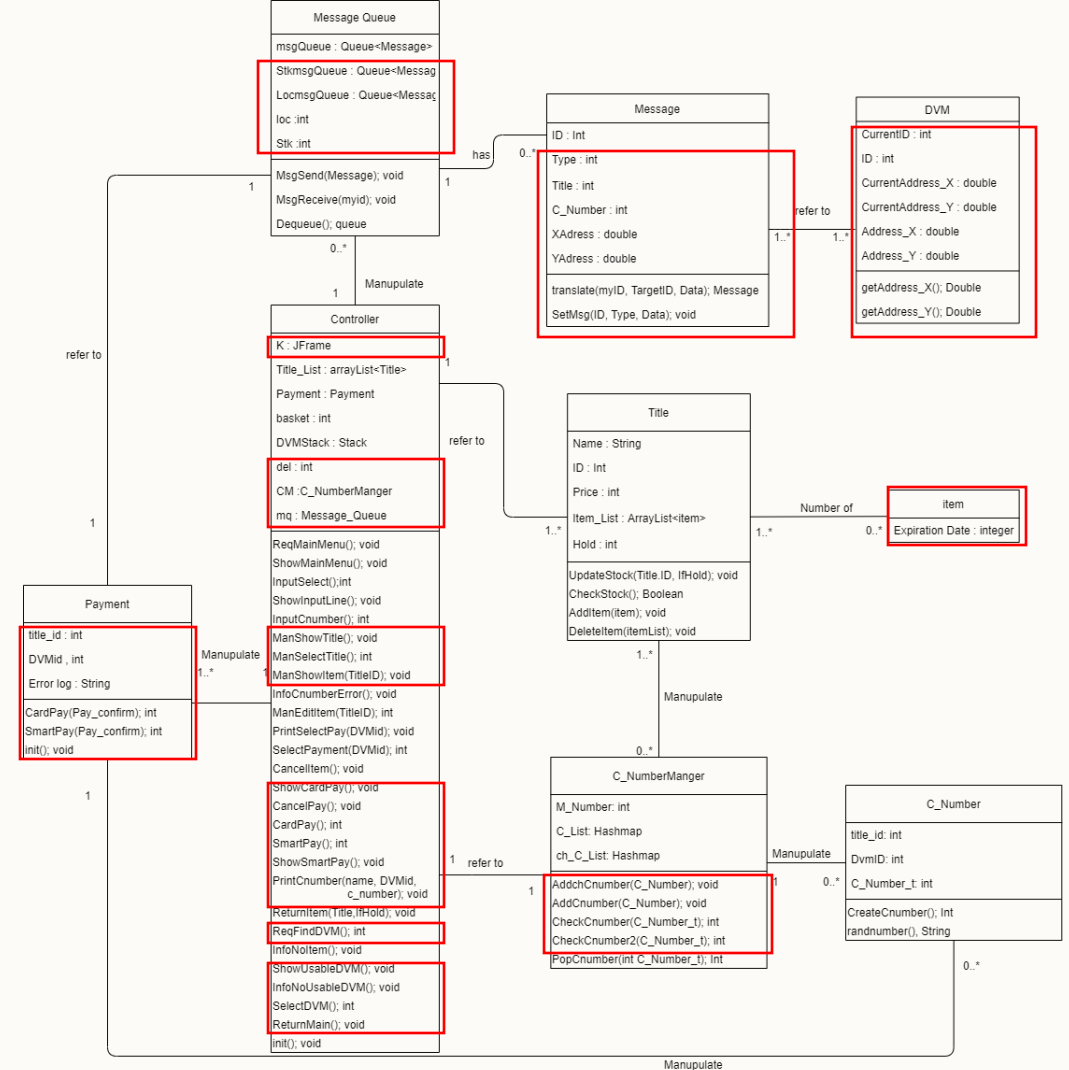
QR 코드 정상 출력 확인

Activity 2051. Implement Class & Methods Definitions

수정 전



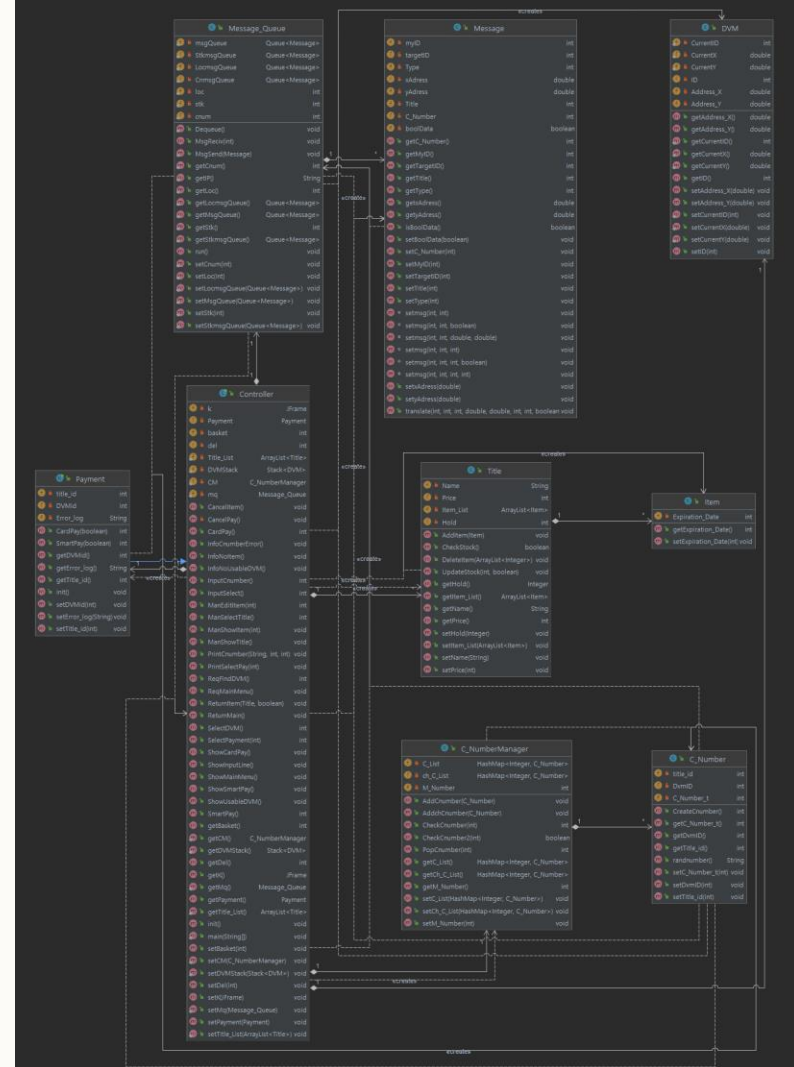
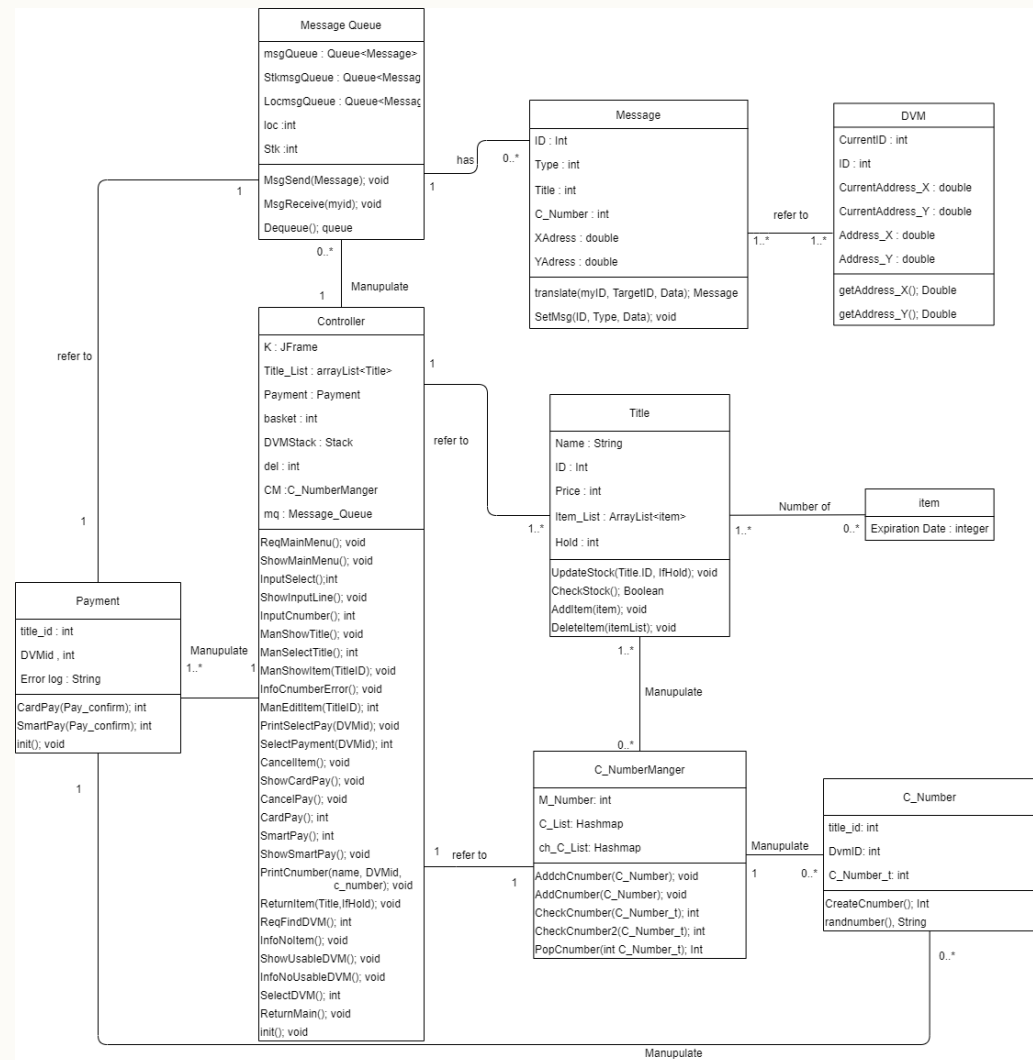
수정 후



Activity 2051. Implement Class & Methods Definitions

수정 후

실제 구현



Activity 2055. Write Unit Test Code & Unit Testing

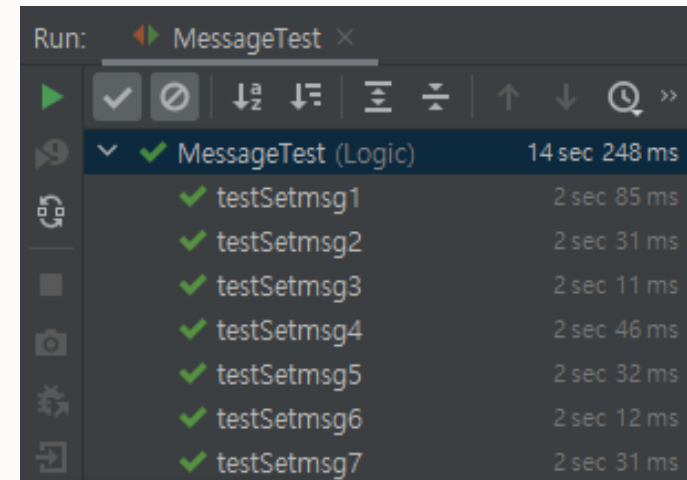
추가된 메소드에 대한 Unit Test Code 작성

Class: Message

```
@Test
public void testSetmsg6() {
    msg.setmsg( id: 1, type: 6, data: 971125);
    Assert.assertEquals( expected: 1, msg.getTargetID());
    Assert.assertEquals( expected: 6, msg.getType());
    Assert.assertEquals( expected: 971125, msg.getC_Number());
}

@Test
public void testSetmsg7() {
    msg.setmsg( id: 1, type: 7, data1: 971125, data2: true);
    Assert.assertEquals( expected: 1, msg.getTargetID());
    Assert.assertEquals( expected: 7, msg.getType());
    Assert.assertEquals( expected: 971125, msg.getC_Number());
    Assert.assertTrue(msg.isBoolData());
}
```

결과



Test Method	Execution Time
MessageTest (Logic)	14 sec 248 ms
testSetmsg1	2 sec 85 ms
testSetmsg2	2 sec 31 ms
testSetmsg3	2 sec 11 ms
testSetmsg4	2 sec 46 ms
testSetmsg5	2 sec 32 ms
testSetmsg6	2 sec 12 ms
testSetmsg7	2 sec 31 ms

Activity 2055. Write Unit Test Code & Unit Testing

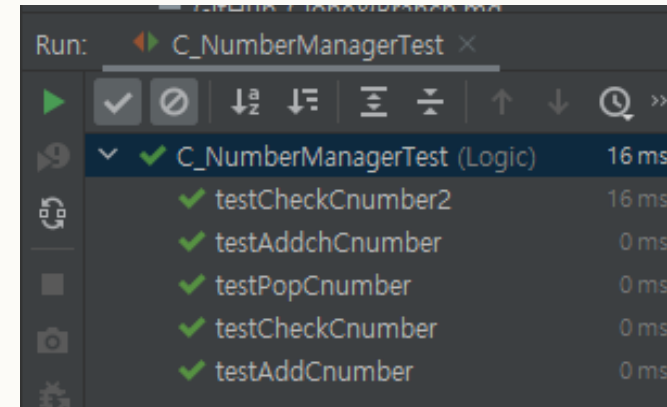
추가된 메소드에 대한 Unit Test Code 작성

Class: C_NumberManager

```
@Test
public void testCheckCnumber2() {
    Cn.setC_Number_t(999999);
    CM.AddchCnumber(Cn);
    boolean ActualResult = CM.CheckCnumber2( C_Number_t 999999);
    Assert.assertTrue(ActualResult);
    ActualResult = CM.CheckCnumber2( C_Number_t 888888);
    Assert.assertFalse(ActualResult);
}

@Test
public void testAddchCnumber() {
    Cn.setC_Number_t(971125);
    CM.AddchCnumber(Cn);
    int ExpectedResult = 1;
    int ActualResult = CM.getCh_C_List().size();
    Assert.assertEquals(ExpectedResult, ActualResult);
}
```

결과



Activity 2052. Implement Windows (사용자 매뉴얼)

재고가 존재하지 않는 제품 선택



Activity 2052. Implement Windows (사용자 매뉴얼)

선결제 된 인증번호 입력



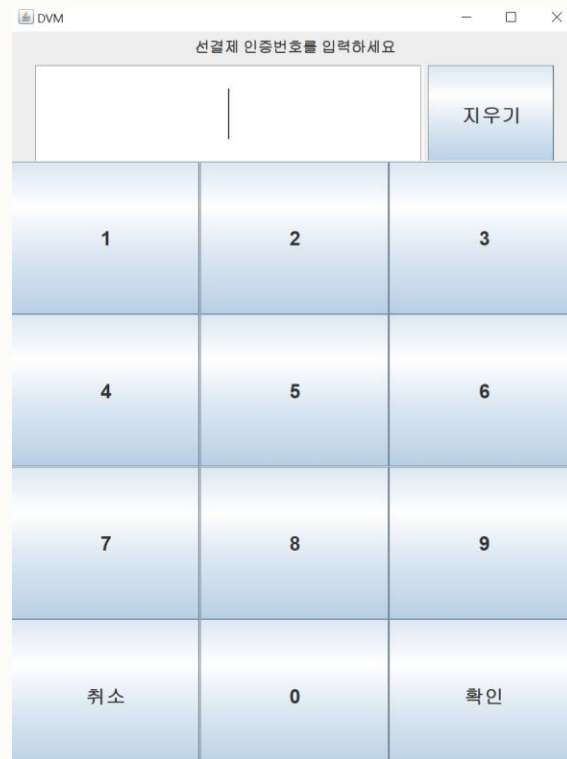
The screenshot shows a window titled "선결제 인증번호를 입력하세요" (Enter pre-paid authentication code). The window contains a text input field with the value "117345" and a "지우기" (Clear) button. Below the input field is a numeric keypad with buttons for digits 1-9, 0, and "취소" (Cancel). The "확인" (Confirm) button is located at the bottom right of the keypad.



The screenshot shows a window titled "2초 후 메인화면으로 돌아갑니다." (Return to main screen in 2 seconds). The window contains a confirmation message: "말키스 음료가 나왔습니다." (Malke's beverage is ready).

Activity 2052. Implement Windows (사용자 매뉴얼)

존재하지 않는 인증번호 입력



The screenshot shows a window titled 'DVM' with the subtitle '선결제 인증번호를 입력하세요'. It features a text input field with a vertical cursor and a '지우기' (Erase) button. Below the input field is a 4x3 grid of buttons containing the digits 1 through 9, '취소' (Cancel), and '확인' (Confirm).



The screenshot shows a window titled 'DVM' with the subtitle '2초 후 이전화면으로 돌아갑니다.' (Returns to the previous screen in 2 seconds). The main text reads '해당 인증번호에 대한 선결제 정보를 확인할 수 없습니다. 다시 입력하세요.' (Cannot check pre-payment information for the corresponding authentication number. Please re-enter it.) and there is a '확인' (Confirm) button at the bottom.

Activity 2052. Implement Windows (사용자 매뉴얼)

관리자 모드

관리자 고유번호 입력

선결제 인증번호를 입력하세요		
111111		
1	2	3
4	5	6
7	8	9
취소	0	확인



코카콜라 재고수량 : 1
나랑드사이다 재고수량 : 0
솔의눈 재고수량 : 0
게토레이 재고수량 : 0
스프라이트 재고수량 : 0
포카리 스웨트 재고수량 : 0
덕터비피 재고수량 : 0
맥콜 재고수량 : 0
제티 재고수량 : 0
나가기

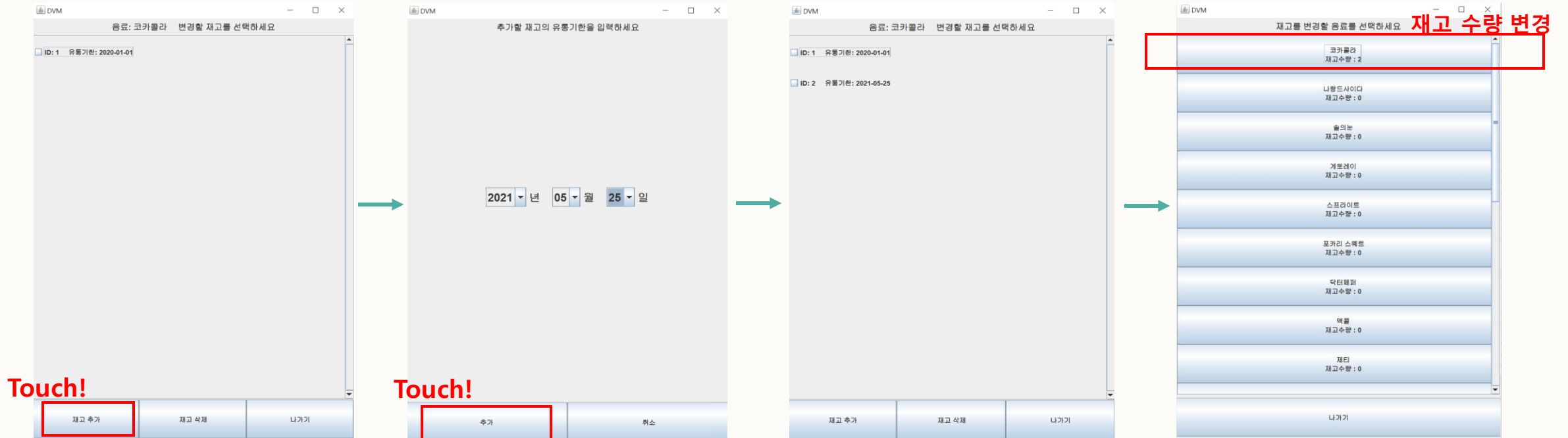
Activity 2052. Implement Windows (사용자 매뉴얼)

*간편결제 시 QR코드 출력



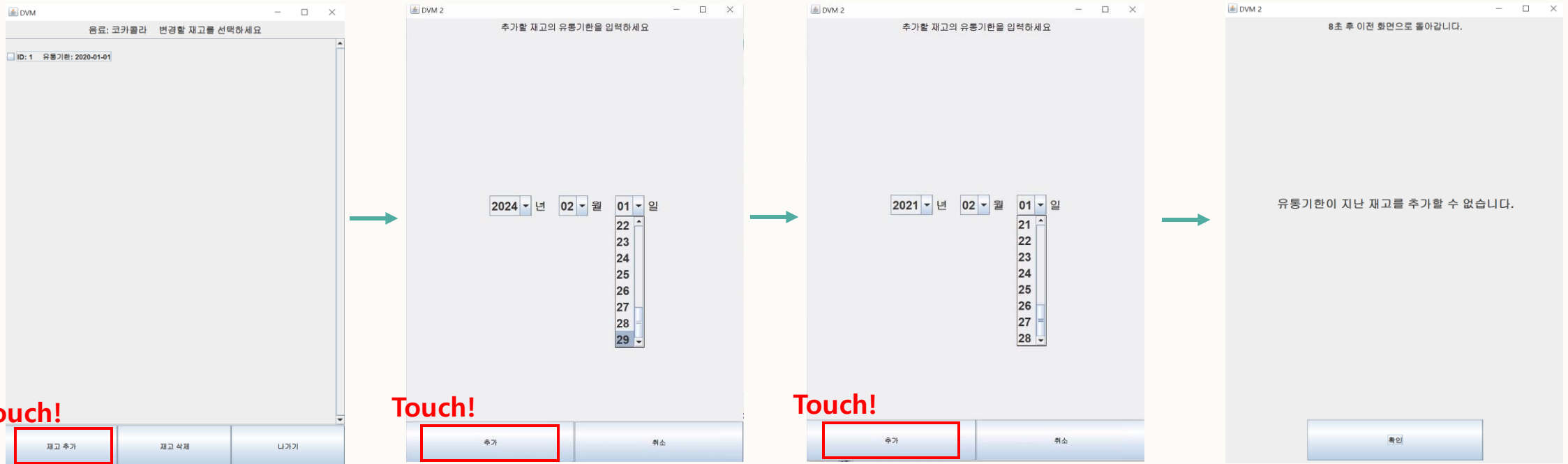
Activity 2052. Implement Windows (사용자 매뉴얼)

재고 추가



Activity 2052. Implement Windows (사용자 매뉴얼)

*재고 추가



윤달 설정

유통기한이 현재 날짜보다 이전인 재고 추가 불가

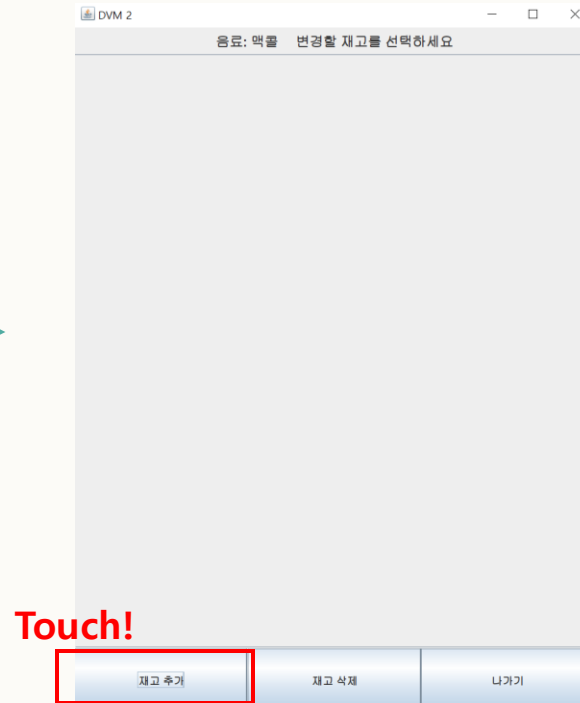
Activity 2052. Implement Windows (사용자 매뉴얼)

*재고 추가

한 DVM에 추가할 수 있는 음료의 종류는 최대 7

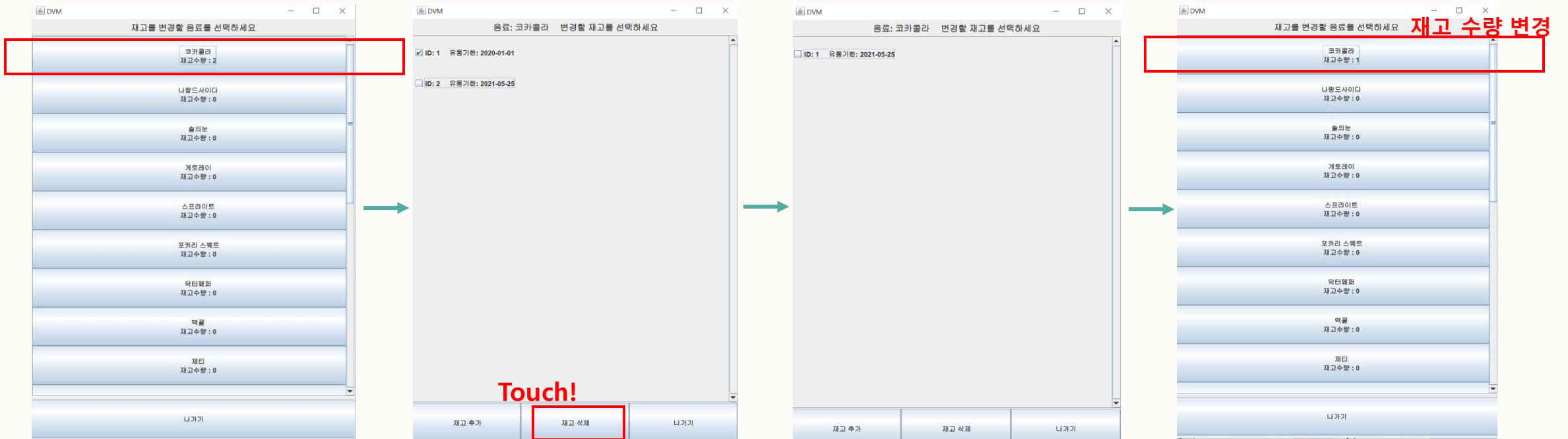


현재 DVM에 존재하는 음료 종류는 7개



Activity 2052. Implement Windows (사용자 매뉴얼)

재고 삭제



Activity 2063. System Testing

Ref	Use Case name	Test Description	P/F
R1.1	1. 자판기 음료 종류 출력	- 20가지 메뉴가 정상적으로 출력되는지 확인	P
R1.2.1	2. 구매할 음료 입력	- 사용자가 선택한 음료가 정확히 기록되는지 확인	P
R.1.2.2	3. 사용자 선택 취소	- 사용자 선택 취소 시 메인 화면으로 돌아가는 지 확인	P
R1.2.3	4. 사용자 인증번호/바코드 입력	- 유효하지 않은 인증번호가 입력된 경우 오류 메시지를 출력하는지 확인	P
		- 올바른 인증번호가 입력된 경우 보관된 제품 중 인증번호에 맞는 제품을 전달하는지 확인	P
R1.3	5. 재고가 부족한 제품 안내	- '재고가 부족한 제품' 이라는 안내문구를 정확히 출력하는지 확인	P
R1.4	6. 구매 가능한 자판기 안내	- 정확한 위치정보를 출력하는지 확인	P
		- 지정된 물품의 재고가 정확히 존재하는지 확인	P
		- 지정된 물품의 재고가 존재하는 자판기가 모두 출력되는지 확인	P

Activity 2063. System Testing

Ref	Use Case name	Test Description	P/F
R1.5	7. 사용자 자판기 선택	- 사용자가 선택한 DVM의 id가 정확히 기록되는 지 확인	P
R2.1	8. 결제 수단 목록 출력	- 카드결제, 간편결제 버튼이 모두 출력되는지 확인.	P
		- 사용자가 선택한 제품 이름과 가격이 정상적으로 출력되는지 확인	
R2.2	9. 결제 수단 결정	- 사용자가 선택한 결제 수단에 따른 결제 화면으로 넘어가는지 확인	P
R2.3.1	10. 간편 결제	- 입력된 정보에 따라 정상적으로 성공/실패 여부가 출력되는지 확인	P
R2.3.2	11. 카드 결제		
R2.4	12. 결제 취소	- 결제 취소 시 메인 화면으로 돌아가는 지 확인	P
R3.1.1	13. 인증번호/바코드 생성	- 인증번호/바코드가 중복되지 않게 생성되는지 확인	P
R3.1.2	14. 인증번호/바코드 출력	- 생성된 인증번호/바코드가 정상적으로 출력되는지 확인	P

Activity 2063. System Testing

Ref	Use Case name	Test Description	P/F
R3.2	15. 음료 전달	- 사용자가 결제를 성공한 제품이 제대로 전달되는지 확인	P
R3.3	16. 재고 수량 업데이트	- 사용자 이용 이후 변화된 재고가 정확히 변화하는지 확인	P
R4.1	17. 관리자 메뉴 출력	- 인증번호 입력화면에서 관리자 고유 번호를 입력함에 따라 메뉴가 출력 되는지 확인	P
R4.2	18. 재고 변경/관리	- 관리자의 동작에 따라 재고의 수량이 잘 변경되는지 확인	P
R4.3	19. 위치 정보 확인	- 정확한 위치 정보를 반환하는지 확인	P
R4.4	20. 재고 정보 확인	- 정확한 재고 정보를 반환하는지 확인	P
R4.5.1	21. 메시지 발신	- DVM네트워크를 통한 메시지가 정확히 목표 DVM으로 전송되는지 확인	P
R4.5.2	22. 메시지 수신	- DVM네트워크를 통한 메시지가 정확히 수신되는지 확인	P

