Documenting Software Architecture for Distributed Vending Machine System

Version 1.0, June 14th, 2021

전다윤, 허윤아, 김두리

Table of Content

1. Introduction Section ***3***

2. System Purpose Section ***3***

2.1 Context Section *3*

2.2 System Interface Section *3*

2.3 Non-Functional Requirements Section *5*

3. Structure Section ***7***

3.1 Overview Section *7*

3.2 Components Section *8*

3.3 Interfaces Section *10*

4. Dynamic Behavior Section ***11***

4.1 Scenarios Section *11*

4.2 Mechanisms Section *14*

5. Other Views Section *오류! 책갈피가 정의되어 있지 않습니다.*

5.1 Process View Section ***오류! 책갈피가 정의되어 있지 않습니다.***

5.2 Development View Section ***오류! 책갈피가 정의되어 있지 않습니다.***

5.3 Physical View Section ***오류! 책갈피가 정의되어 있지 않습니다.***

6. Conceptual Framework Section *오류! 책갈피가 정의되어 있지 않습니다.*

7. Conclusion Section *오류! 책갈피가 정의되어 있지 않습니다.*

1. Introduction Section

Name of the architecture	분산형 자판기 아키텍처(Distributed Vending Machine Architecture)
Team information	- 전다윤(건국대학교 일반대학원 컴퓨터공학부 HPE Lab) - 허윤아(건국대학교 일반대학원 컴퓨터 공학부 DS Lab) - 김두리(건국대학교 일반대학원 컴퓨터공학부 INNS Lab)
Creation and modification	None
Audience and purpose	DVM Company, Explain how to make DVM
Selected viewpoints	Logical View – Decomposition & Layered View Process View – Peer-to-Peer View Development View – Deployment View, Install View, Work Assignment View

2. System Purpose Section

2.1 Context Section

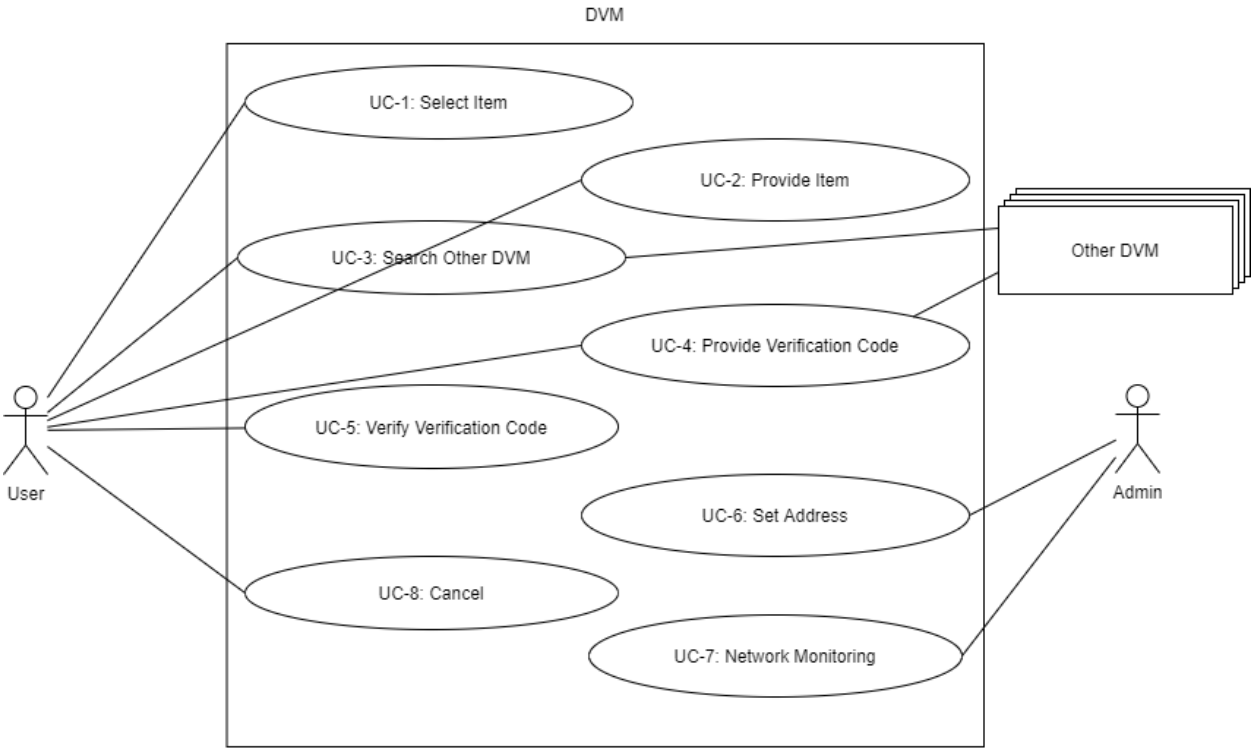


Figure 1 Use Case Diagram

2.2 System Interface Section

Table 1 User Input Interface

Interface	User input Interface

Use Cases	**UC-1. Select Item:** 사용자가 선택한 메뉴의 재고가 있을 경우 사용자에게 '구매하시겠습니까?' 등의 메시지로 알리고, 재고가 없을 경우 사용자에게 '다른 자판기의 재고를 찾아보시겠습니까?' 등의 메시지로 알린다. System operations of this use case: - RequestItem(itemID): 사용자가 선택한 item의 재고가 있는지 확인할 것을 요청한다. - StockExistence: 상품의 재고 여부를 반환한다.
	UC-2. Provide Item: 사용자가 결제를 진행하고 나면 상품을 제공한다. System operations of this use case - ProvideItem() : Item을 제공해준다.
	UC-3. Search Other DVM : 사용자가 다른 자판기에서 재고를 찾겠다고 선택했을 때, 선택된 상품의 재고 여부를 가장 가까운 자판기부터 순서대로 요청하면, 메시지를 받은 자판기는 상품의 재고 여부를 확인한 후 메시지로 응답한다. 재고가 있는 경우, 사용자에게 해당 자판기의 위치를 알리고, 선결제를 진행할 것인지 묻는다. System operations of this use case: - SearchOtherDVM(itemID, DVMID): 재고를 보유한 DVM 찾기를 요청한다. - DVMLocation: 재고를 보유한 DVM의 위치를 반환한다.
	UC-4. Provide Verification Code : 사용자가 선결제를 하면 인증코드를 생성하고 사용자와 해당 자판기에 제공한다. - System operations of this use case - RequestPayement() : 사용자가 결제를 진행하면 Item의 재고가 있는 다른 DVM에 Verification Code를 요청해 생성하고 받아 사용자에게 제공한다..
	UC-5. Verify Verification Code : 사용자가 다른 자판기에서 선결제를 통해 받은 인증코드를 입력하면, 제공받았던 인증코드와 동일한 지 그 유효성을 확인한다. 유효한 인증코드라면, 사용자에게 해당 상품을 제공한다. System operations of this use case: CheckVerificationCode(VerificationCode) : 사용자가 VericationCode를 DVM에 입력하면 DVM은 DB에 Verification Code와 비교하여 유효한 코드인지 확인하고 맞으면 사용자에게 Item을 제공한다.
	UC-8. Cancel : 사용자가 했던 모든 입력을 취소하고 초기 상태로 되돌아간다. System operations of this use case: Cancel() : 사용자가 Cancel버튼을 누르면 진행중인 과정을 멈추고 초기화면으로 돌아간다.

Table 2 Admin Input Interface

Interface	Admin Input Interface
Use Cases	**UC-6. Set Address** : 관리자가 자판기의 주소를 입력한다. System operations of this use case: SetAddress(Address) : 관리자가 DVM 에 각 DVM 의 위치를 입력하면 DVM 이 이것을 저장한다.
	UC-7. Network Monioring : 관리자가 자판기 사용이력을 본다. System operations of this use case: NetworkMonitoring() : 자판기의 사용이력을 관리자에게 보여준다.

2.3 Non-Functional Requirements Section

1. **Quality**

Table 3 Quality Attribute

ID	Quality Attribute	Scenario
QA-1	Usability	사용자는 자판기에서 원하는 상품을 뽑기 위해 버튼을 누른다. 자판기는 재고가 있는 경우 결제 이후 최소 5 초 안에 상품을 제공하고 재고가 없는 경우 최소 30 초 안에 다른 자판기의 재고를 30 초 안에 파악하여 안내한다.
QA-2	Marketability	마케터는 출시 이전 분산 자판기에 대한 설문조사를 한다. 설문조사를 통해 사용자들의 니즈를 파악하여 DVM 의 시장 경쟁성을 높인다.
QA-3	Reusability	코드 안에서 코드의 중복성을 최소화하기 위하여 정적분석을 한다. 정적분석을 통하여 코드의 중복성을 최대 10%로 줄여 다른 시스템이나 애플리케이션에서 해당 시스템을 활용할 수 있도록 한다.
QA-4	Performance	하나의 자판기에서 여러 자판기로부터 최대 4 개의 메시지를 수신한다. 메시지를 수신한 자판기는 60 초 이내에 메세지를 모두 처리한다.
QA-5	Maintainability	3 명의 개발자가 하나의 컴포넌트 코드를 수정하고자 할 때, 3 시간 이내에 수정 가능하다.
QA-6	Testability	테스터가 12 시간 이내로 테스트를 위한 가상의 환경을 구축한다. 테스터가 테스트를 위한 가상의 환경이 구축된 상태에서 코드를 테스트하고 자 할 때, 6 시간 이내로 85%이상 테스트한다.
QA-7	Security	외부시스템에서 시스템 탈취를 시도하거나 네트워크로 전달되는 메세지에 접근하고자 할 때 , 30 초 이내에 접근을 감지 후, 5 초 이내에 관리자에게 위험 메세지를 전달한다.

2. Constraints

Table 4 Constraints

ID	Constraint
CON-1	여러 자판기에서 같은 동작이 일어날 수 있어야 한다.
CON-2	네트워크는 낮은 대역폭을 가질 수 있지만, 신뢰할 수 있어야 한다.
CON-3	매번 동일한 환경에서 동일한 동작을 해야 한다.

3. **Principles**

Table 5 Principles

Design Decisions and Location	Rationale and Assumptions
DVM SW에 **Cancel** 기능을 위해 UI와 system을 분리	사용자가 결제 전, 진행 중이던 프로세스를 취소할 수 있도록 취소 기능을 추가하여 사용자의 편의성을 증가시킨다. 언제든 User가 Cancel을 요청했을 때 입력한 정보들을 전부 초기화하고 메인 화면으로 돌아가야 하므로, UI는 내부의 controller와는 Asynchronous하게 동작해야 한다. Cancel 에 대한 Use-Case를 추가하고 해당 Use-Case의 Sequence diagram를 생성한다. Usability가 향상된다.
Receiver로 수신한 메시지를 처리할 때 **Priority**를 부여하여 처리	다른 자판기로부터 메시지를 수신할 때, Receiver가 처리해야 하는 메시지의 우선순위를 부여하여 처리한다. Receiver는 메시지를 처리할 때 아래와 같은 우선순위를 토대로 처리하게 되며, sequence diagram에 변화가 생기고, Performance가 향상된다. 처리 우선 순위: 재고 확인 메시지 > 인증코드 수신
Forwarder-Receiver사이 와 DataAccessManger 에 메시지 무결성 확인을 위해 **Private Key, Public Key** 적용	Forwarder가 메시지를 보내기 전 Public Key를 통해 Encryption을 진행하고, Receiver가 메시지를 수신할 때 Public Key를 통해 수신한 메시지의 Decryption을 진행한다. Data를 DBServer에 저장할 때, DataAccessManager가 Private Key를 통해 Encryption을 진행하고, Data를 DBServer로부터 읽어 오려고 할 때, DataAccessManager가 Private Key를 통해 Decryption을 진행한다. Data를 암호화할 때, DVMID와 itemID를 제외한 내용만 암호화하여 데이터의 접근성은 유지하되, security는 향상시킬 수 있도록 한다. Sequence Diagram, Module View에 변화가 생긴다.

3. Structure Section

3.1 Overview Section

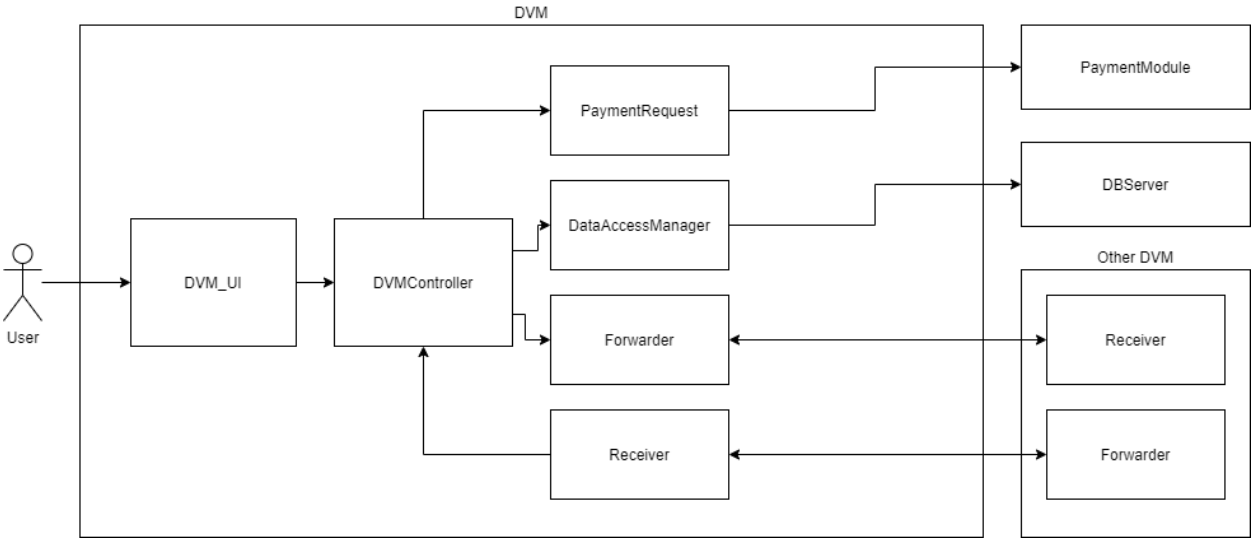


Figure 2 Overview Architecture Diagram

User는 DVM_UI를 통해 입력하고, DVM_UI를 통해 들어온 입력은 DVMController를 통해 어떤 동작을 할지 결정하고 작동하게 된다. Payment를 할 경우, PaymentRequest를 통해 Payment Module로 가 결제를 진행한다.

Verfication Code를 체크하거나, 재고를 확인할 때는 DataAcessManger를 통해 DBServer로 가 안전하게 Data를 확인한다.

다른 DVM과 통신할 때는 Forwarder를 통해 보내고 Receiver를 통해 받으며 통신한다.

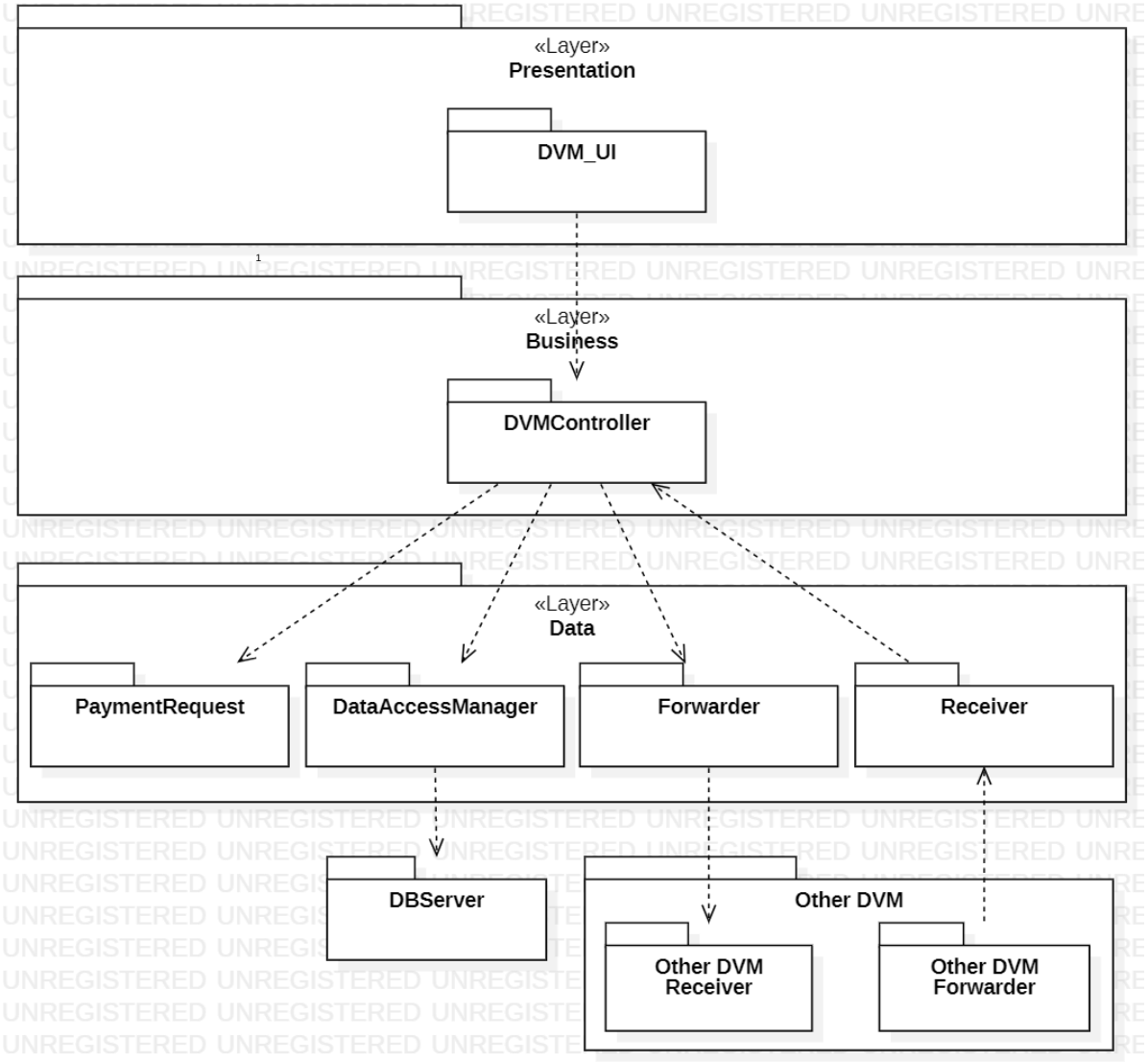


Figure 3 Module View using Rich Client Reference Architecture

Commentary

Rich Client Application reference architecture : Rich Client application은 자판기에 설치되는 프로그램의 개발에 도움을 준다. 또한, QA-1의 usability를 향상시킬 수 있고, 사용자의 요청이 있을 때만 네트워크에 접속하면 된다는 점에서 네트워크 비용을 아낄 수 있다.

3.2 Components Section

Component	DVM_UI
Responsibilities	사용자와 직접 interaction이 가능한 UI로, 사용자가 선택할 수 있는 상품을 보여주고, 결제를 완료한 상품을 제공하거나 다른 자판기에서 사용 가능한 인증코드를 제공한다.
Collaborators	*DVMController*
Notes	-
Issues	-

Component	DVMController
Responsibilities	사용자의 입력에 따른 전체적인 동작을 위한 기능을 포함한다.
Collaborators	*DVM_UI, PaymentRequest, DataAccessManager, Forwarder, Receiver*
Notes	-
Issues	-

Component	PaymentRequest
Responsibilities	사용자가 (선)결제를 요청할 경우, 외부 결제 모듈에 결제를 요청을 한다.
Collaborators	*DVMController, PaymentModule*
Notes	-
Issues	-

Component	DataAcessManager
Responsibilities	각 DVM 의 위치, DVM 의 ID, 취급하는 상품 종류, 각 상품의 재고에 대한 정보와 송수신한 인증번호를 DBServer 에 저장하거나 불러온다..
Collaborators	*DVMController, DBServer*
Notes	-
Issues	-

Component	Forwarder
Responsibilities	전송할 정보를 파일의 형태로 marshalling 하여 다른 자판기의 receiver 로 전송한다.
Collaborators	*DVMController, (Other DVM's) Receiver*
Notes	-
Issues	-

Component	Receiver
Responsibilities	다른 자판기의 forwarder 로부터 파일의 형태로 수신한 정보를 unmarshalling 하여 자판기에 전달한다.
Collaborators	*DVMController, (Other DVM's) Forwarder*
Notes	-
Issues	-

Component	DBServer
Responsibilities	각 자판기의 log 정보와 위치, 식별번호, 상품 종류 정보를 가지고 있다.
Collaborators	*DataAccessManager*
Notes	-
Issues	-

3.3 Interfaces Section

생략

4. Dynamic Behavior Section

4.1 Scenarios Section

Scenario Specification

Use Case	1. Select Item
Description	사용자가 선택한 메뉴의 재고가 있을 경우 사용자에게 '구매하시겠습니까?' 등의 메시지로 알리고, 재고가 없을 경우 사용자에게 '다른 자판기의 재고를 찾아보시겠습니까?' 등의 메시지로 알린다.
Actors	User, DVM, DBServer
Assumption	모든 자판기의 전원은 항상 켜져있고, 네트워크에 항시 연결되어 있다.
Step	1: 사용자가 자판기 화면에서 원하는 메뉴를 선택한다. 2: 원하는 메뉴의 재고가 있을 경우 '구매하시겠습니까?' 메시지를 출력한다.
Variations	2: 재고가 없는 경우 '다른 자판기의 재고를 찾아보시겠습니까?' 메시지를 출력한다.

Use Case	2. Provide Item
Description	사용자가 결제를 진행하고 나면 상품을 제공한다.
Actors	User, DVM, DBServer
Assumption	UC-1 을 수행하였고, 사용자가 접속한 자판기에 원하는 메뉴의 재고가 있다.
Step	1: 사용자가 구매를 선택하고 결제를 진행한다. 2: 결제가 진행된 것을 확인한 후 상품을 제공한다. 3: 재고 변경사항을 업데이트한다.
Variations	1: 사용자가 구매를 선택하지 않는다면 첫 화면으로 돌아간다.

Use Case	3. Search Other DVM
Description	사용자가 다른 자판기에서 재고를 찾겠다고 선택했을 때, 선택된 상품의 재고 여부를 가장 가까운 자판기부터 순서대로 요청하면, 메시지를 받은 자판기는 상품의 재고 여부를 확인한 후 메시지로 응답한다. 재고가 있는 경우, 사용자에게 해당 자판기의 위치를 알리고, 선결제를 진행할 것인지 묻는다.
Actors	User, DVM, DBServer, OtherDVM
Assumption	UC-1 을 수행하였고, 사용자가 접속한 자판기에 원하는 메뉴의 재고가 없다.
Step	1: 사용자가 다른 자판기의 재고를 찾는 것을 선택한다. 2: 재고가 있는 다른 자판기를 찾는다. 3: 재고가 있는 다른 자판기의 위치를 사용자에게 알린다. 4: 사용자에게 선결제를 진행할 것인지 묻는다.
Variations	-

Use Case	4. Provide Verification Code
Description	사용자가 선결제를 하면 인증코드를 생성하고 사용자와 해당 자판기에 제공한다.

Actors	User, DVM, DBServer, OtherDVM
Assumption	UC-3 을 수행하였다.
Step	1: 사용자가 선결제를 진행한다. 2: 결제가 진행된 것을 확인한 후 인증코드를 제공한다. 3: 인증코드를 재고가 있는 자판기로 보낸다.
Variations	1: 선결제를 진행하지 않는다면 첫 화면으로 돌아간다.

Use Case	5. Verify Verification Code
Description	사용자가 다른 자판기에서 선결제를 통해 받은 인증코드를 입력하면, 제공받았던 인증코드와 동일한 지 그 유효성을 확인한다. 유효한 인증코드라면, 사용자에게 해당 상품을 제공한다.
Actors	User
Assumption	
Step	1. (U) User 가 VerificationCode 를 입력한다 2. (S) DVM_UI 가 DVM_Controller 로 Verification Code 를 보내준다. 3. (S) DVM Controller 가 DatabaseAccessManger 로 Verification Code 를 보낸다. 4. (S) DatabaseAccessManger 가 DBServer 에서 Verification Code 를 통해 읽은 Verificaiton Code 가 생성된 DVM ID, ItemID 를 읽이 이것과 맞는 VerificationCode 를 가져온다. 5. (S) DatabaseAccessManger 가 가져온 Verification Code 와 User 가 입력한 Verification Code 를 비교한다. 6. (S) 같은 Verification Code 라고 나오거나, 더이상 DBServer 에서 가져올 Verification Code 가 없을 때 까지 4,5 를 반복한다. 7. (S) 같은 Verificatoin Code 가 나오면 해당 Item 을 User 에게 주고, 아닌 경우, 유효하지 않은 Verification Code 라는 메세지를 User 에게 보낸다.
Variations	

Use Case	6. Set Address
Description	관리자가 자판기의 주소를 입력한다.
Actors	Manager
Assumption	관리자모드로 로그인한 상태이다.
Step	1. (M) DVM 시스템에 DVM Address 를 입력한다. 2. (S) DVM_UI 가 DVM Controller 로 DVM Address 를 보내준다. 3. (S) DVM Controller 가 DatabaseAccessManger 를 에게 DVM Address 를 보낸다. 4. (S) DatabaseAccessManager 가 DBServer 에 DVM Address 를 저장한다.
Variations	

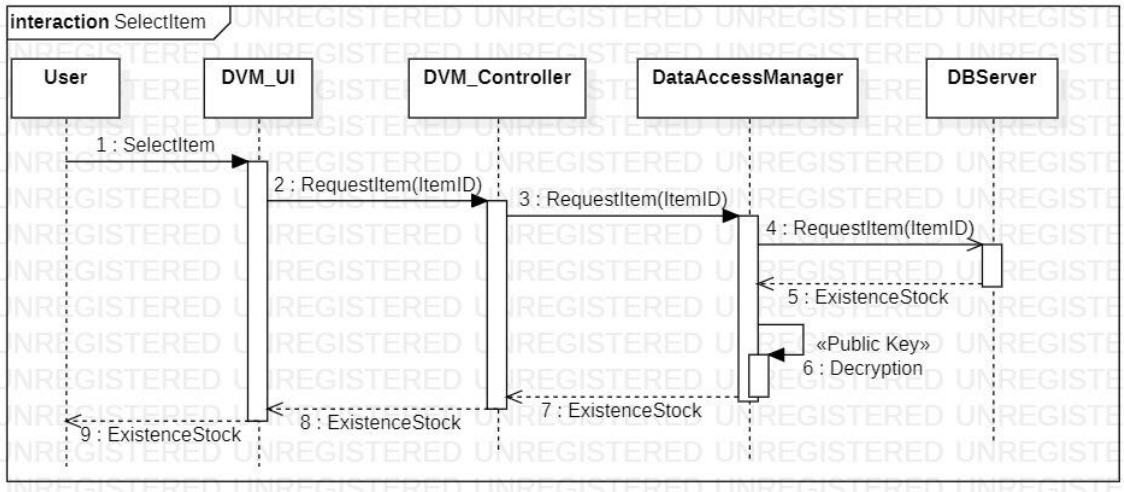
Use Case	7. Network Monitoring
Description	관리자가 자판기 사용이력을 본다.
Actors	Manager

Assumption	관리자 모드에서만 할 수 있다.
Step	1. (M) 관리자 DVM 의 Network 보기를 실행한다. 2. (S) DVM_UI 가 DVMController 를 통해 DatabaseAccessManger 에게 Log 를 요청한다. 3. (S) DatabaseAccessManager 가 DBServer 에서 Log 를 받아온다. 4. (S) Log 를 받은 DVM Controller 가 DVM_UI 에 전달해 Log 를 보여준다.
Variations	

Use Case	8. Cancel
Description	Cancel 버튼을 누르면 진행중인 동작을 멈추고 초기 상태으로 돌아간다.
Actors	User
Assumption	
Step	(U)User 가 Cancel 버튼을 누른다. (S) DVMController 가 진행중인 동작을 멈추고 취소한다. (S) 초기 상태로 돌아간다.
Variations	

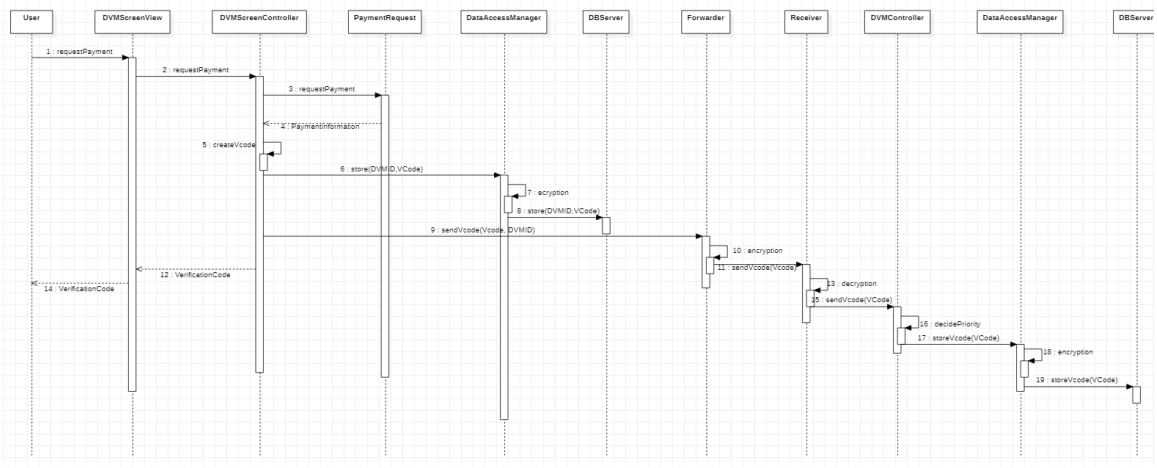
Component Interaction Model

<UC-1>

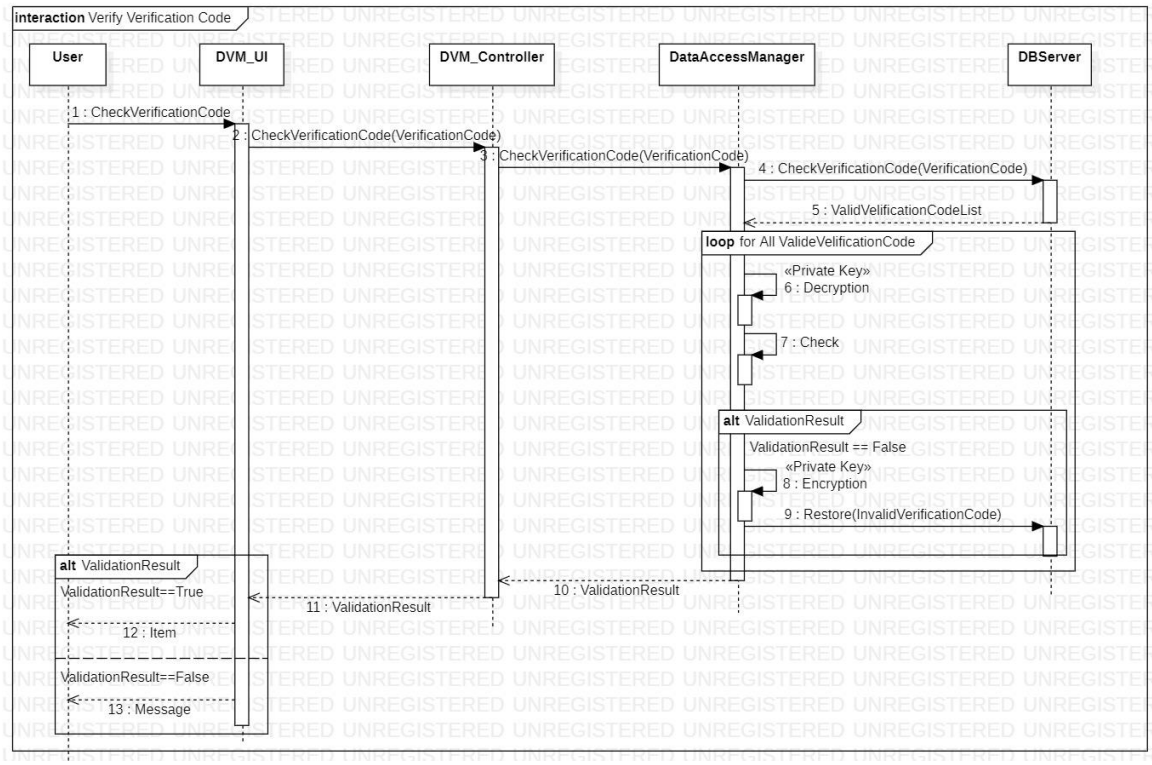


<UC-3>

<UC-4>



<UC-5>



<UC-8>

4.2 Mechanisms Section

해당 없음