ADD
3 Iteration

소프트웨어 공학 1팀
윤상혁, 최슬아, 루카이

ADD Step 1: Review Inputs and Identifying Drivers

<table>
<tr><td>Category</td><td>Details</td></tr>
<tr><td>Design Purpose</td><td>분산 자판기를 위한 소프트웨어와 그 관리 소프트웨어를 개발한다.</td></tr>
<tr><td>Primary functionality requirements</td><td>UC-2: 서비스의 핵심과 직접적으로 연관된다.
UC-5: 서비스의 핵심과 직접적으로 연관된다.
UC-6: 서비스의 핵심과 직접적으로 연관된다.
UC-7: 서비스 운영의 원활함을 위해 필요하다.</td></tr>
<tr><td>Quality attributes scenarios</td><td>
<table>
<tr><th>Scenario ID</th><th>Importance to the Customer</th><th>Difficulty of Implementation According to the Architect</th></tr>
<tr><td>QA-1</td><td>Medium</td><td>Medium</td></tr>
<tr><td>QA-2</td><td>High</td><td>High</td></tr>
<tr><td>QA-3</td><td>Medium</td><td>High</td></tr>
<tr><td>QA-4</td><td>Hight</td><td>High</td></tr>
<tr><td>QA-5</td><td>Medium</td><td>High</td></tr>
<tr><td>QA-6</td><td>High</td><td>High</td></tr>
<tr><td>QA-7</td><td>High</td><td>High</td></tr>
<tr><td>QA-8</td><td>Low</td><td>High</td></tr>
<tr><td>QA-9</td><td>High</td><td>Medium</td></tr>
<tr><td>QA-10</td><td>High</td><td>High</td></tr>
</table>
이 리스트에서 QA-2, QA-3, QA-4, QA-6, QA-7, QA-10이 선택되었다.
</td></tr>
<tr><td>Constrains</td><td>CON-1: 모든 음료를 제공하기 위해선 최소 3개의 자판기가 존재해야 한다.
CON-2: 사용자의 구매 내역은 최소 45일동안 저장되어야 한다.</td></tr>
<tr><td>Architectural concerns</td><td>CRN-1: 여러 사용자가 접근할 경우 재고 데이터 무결성
CRN-2: 네트워크가 끊어졌을 경우 다른 자판기에 미치는 영향
CRN-3: 여러 자판기를 쉽게 관리할 수 있는 매니지먼트 시스템</td></tr>
</table>

Selected Architectural Drivers

Use Case	Description
UC-2: 재고 상태 확인	실제 업체에서 보충 가능한 재고의 양에 따라 일시 품절 / 품절의 구분이 가능해야 한다. 모든 자판기의 재고 확인이 가능해야 한다. 재고가 부족할 경우 알림이 와야 한다.
UC-5: 음료 구매	사용자는 자판기의 화면을 보고 음료를 카드로 결제한다. 잔액이 부족한 경우 결제되지 않는다.
UC-6: 선결제 구매	만약 다른 자판기에 있는 음료를 구매할 경우 해당 자판기에서 결제 후 인증 코드를 발급한다. 발급 받은 인증 코드를 입력하면 음료를 받을 수 있다.
UC-7: 새로운 자판기 추가, 삭제	자판기를 추가 혹은 삭제를 하면 주변 자판기에 상태를 업데이트 한다.

ID	Constraints
CON-1	모든 음료를 제공하기 위해선 최소 3개의 자판기가 존재해야 한다.
CON-2	사용자의 구매 내역은 최소 45일동안 저장되어야 한다.

ID	Quality Attribute	Scenario	Associated Use Case
QA-2	Availability	인증 코드 발급 시 해당 자판기에 코드 발급을 확인한 후 사용자에게 코드를 발급한다. 인증 코드의 유효 시간은 10분이며 넘어갈 경우 결제 취소가 진행된다.	UC-6
QA-3	Interoperability	새로운 자판기를 추가하면 모든 자판기에 1분 안에 자신의 정보를 업데이트 한다.	UC-7
QA-4	Performance	현재 자판기에 재고가 없을 경우 항상 구매 가능한 자판기를 찾아 출력해야 한다. 만약 해당 자판기가 멀리 떨어져 있더라도 결과는 10초 안에 출력 되어야 한다.	UC-6
QA-6	Performance	재고의 상태는 실시간으로 확인 가능해야 하며 재고가 부족한 경우 1분 안에 알림이 와야 한다.	UC-2
QA-7	Security	모든 결제 정보는 암호화가 되어 전송 및 저장이 되어야 한다.	UC-5
QA-10	Usability	동시에 여러 사용자가 한 자판기의 음료 구매를 원할 경우에도 항상 정확한 판매 수량을 보여주어야 한다.	UC-5

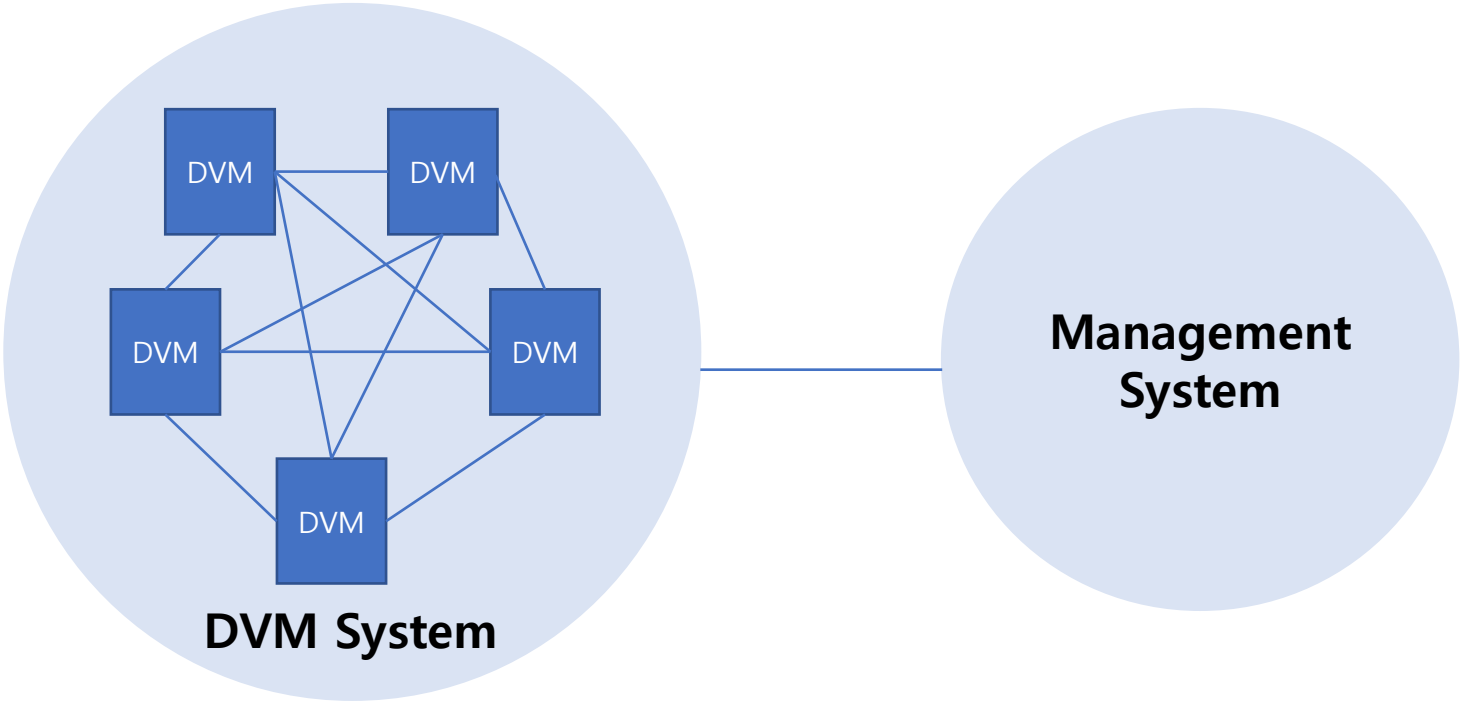
ID	Concern
CRN-1	여러 사용자가 접근할 경우 재고 데이터 무결성
CRN-2	네트워크가 끊어졌을 경우 다른 자판기에 미치는 영향
CRN-3	여러 자판기를 쉽게 관리할 수 있는 매니지먼트 시스템

ADD Iteration 1

ADD Step 2: Establish Iteration Goal by Selecting Drivers

- The iteration goal: 전체적인 시스템을 만족시킬 수 있는 시스템 구조를 선정한다.

ADD Step 3: Choose One or More Elements of the System to Refine



ADD Step 4: Choose One or More Design Concepts that Satisfy the Selected Drivers

- Design concepts selected
  - DVM 시스템을 위한 각각의 자판기에 **Service Application Reference Architecture**를 사용한다.
  - DVM 관리 시스템을 위해 **Rich Client Application Reference Architecture**를 사용한다.
  - DVM 시스템을 위한 물리적 구조로는 **Three-tier Deployment Pattern**을 사용한다.
  - 유저 인터페이스를 위해 **Swing Java Framework**를 사용한다.

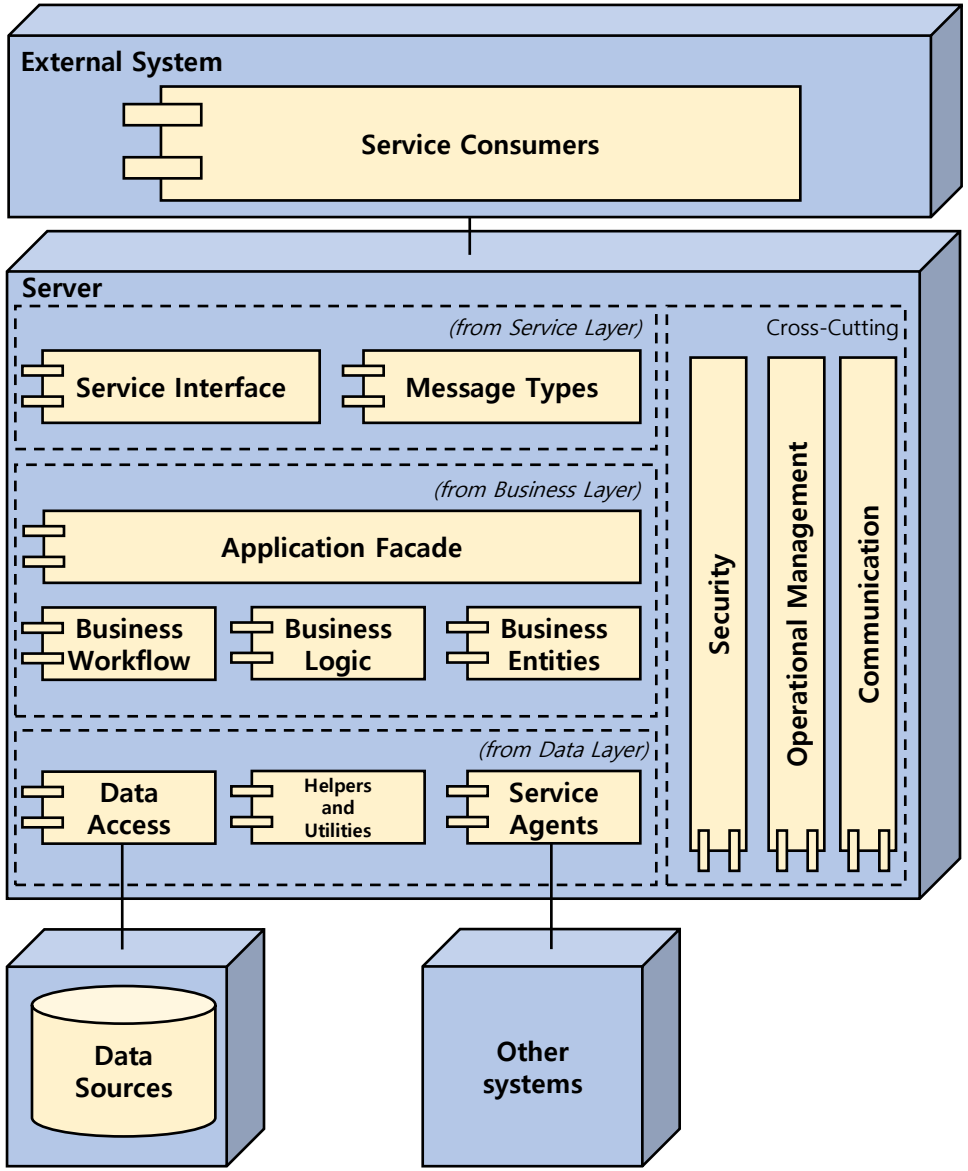
Choosing Design Concepts

- 분산 자판기 시스템을 위해 Service Application Reference Architecture를 사용한다.

Design Decisions and Location	Rationale
DVM을 위한 논리적 구조로 **Service Application Reference Architecture**를 사용한다.	UI 지원은 없지만 클라이언트와 서버가 loosely coupled 되어 있어 분산 시스템에서 사용이 적합하다. Message Broadcasting으로 재고 확인을 하는 DVM과 맞게 메시지 기반으로 통신이 가능하며 관리 시스템과 같은 다른 시스템과도 동작 가능하다(Interoperability). 자판기를 위한 UI는 외부 시스템 연결로 구현이 필요하다.

Alternative	Reason for Discarding
Rich Client Application Reference Architecture	사용자와 풍부한 상호작용과 빠른 반응성을 보여주지만 DVM에서 중요한 것은 UI보단 자판기간의 연결성이 더 중요하므로 선택하지 않았다.

Service Application Reference Architecture



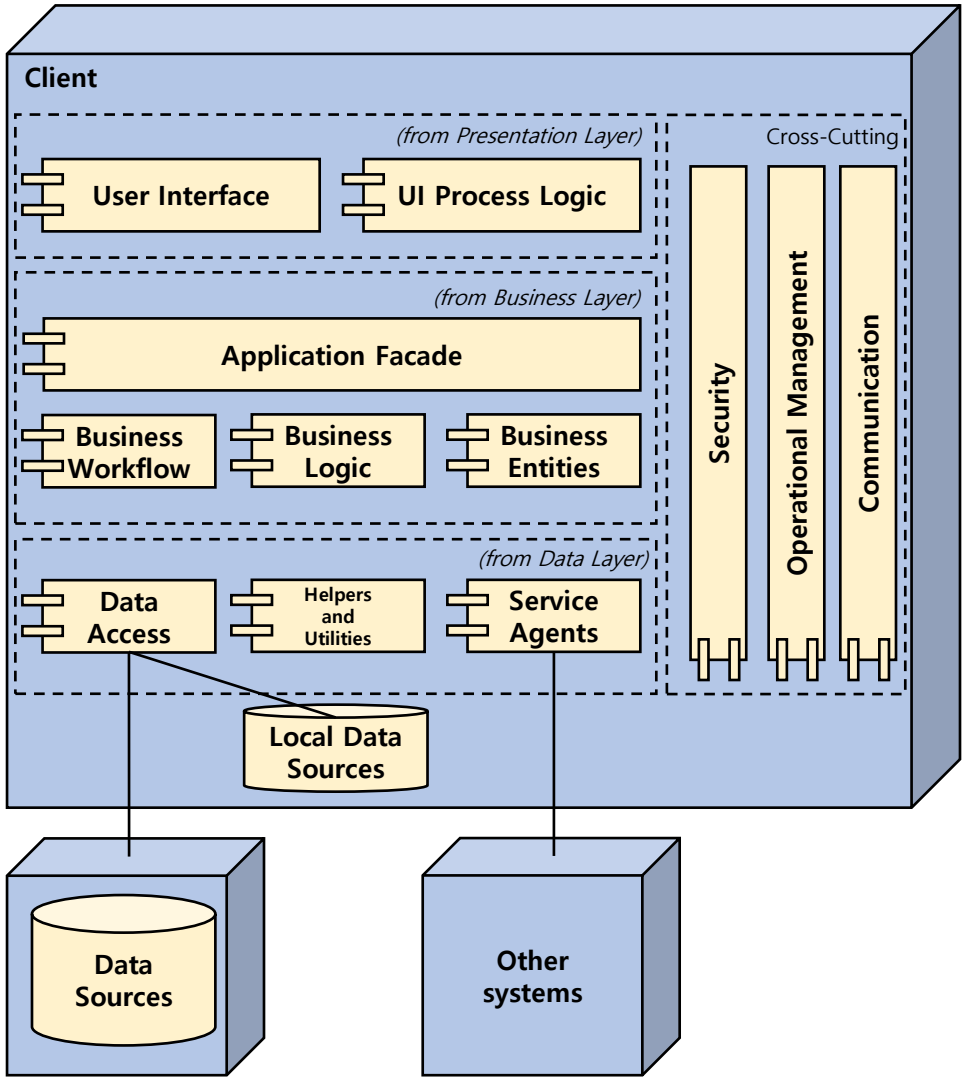
Choosing Design Concepts

- 분산 자판기 관리 시스템을 위해 Rich Client Application Reference Architecture를 사용한다.

Design Decisions and Location	Rationale
DVM 관리 시스템을 위한 논리적 구조로 **Rich Client Application Reference Architecture**를 사용한다.	관리자가 자판기의 재고 수량을 확인하기 쉽게 풍부한 UI 환경을 제공한다. 또한 자판기와 네트워크 통신을 통해 재고 사항을 확인할 수 있다. 데이터베이스에 저장되어 있는 데이터를 읽어와 판매 기록 및 분석을 확인할 수 있다.

Alternative	Reason for Discarding
Rich Internet Application Reference Architecture	웹 브라우저로 접근이 가능하여 접근성은 높지만 여러 자판기 관리, 재고 관리, 재고 부족 시 알림 등 사용자와 풍부한 상호작용에 제약사항이 존재하여 선택하지 않았다.
Web Application Reference Architecture	웹 브라우저로 접근이 가능하여 접근성은 높지만 여러 자판기 관리, 재고 관리, 재고 부족 시 알림 등 사용자와 풍부한 상호작용에 제약사항이 존재하여 고르지 않았다.
Mobile Application Reference Architecture	여러 자판기와 통신하여 재고 사항을 읽어와야 하기 때문에 성능과 네트워크 연결 사항에 제약이 생겨 선택하지 않았다.

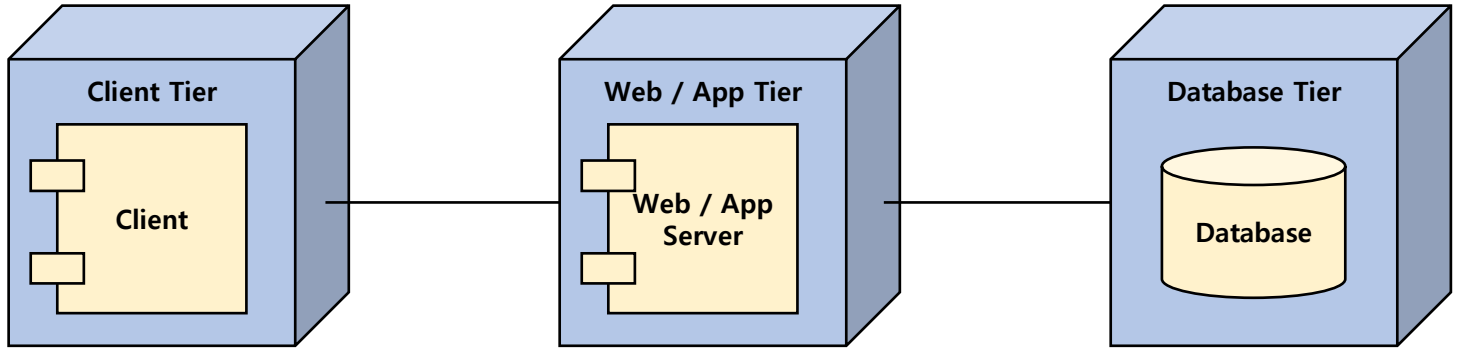
Rich Client Application Reference Architecture



Choosing Design Concepts

- Deployment Pattern: Three-tier Deployment

Design Decisions and Location	Rationale
물리적 구조로 Three-tier deployment pattern을 사용한다.	Con-2를 만족시키기 위해 외부 데이터베이스가 필요하며 UC-2를 위해 외부에서 접속 가능한 클라이언트가 필요하다. DVM은 분산 시스템이기 때문에 여러 web/app tier가 존재해야 CRN-3, QA-6, uc-2를 만족시키기 위해 별도의 클라이언트가 필요하다.

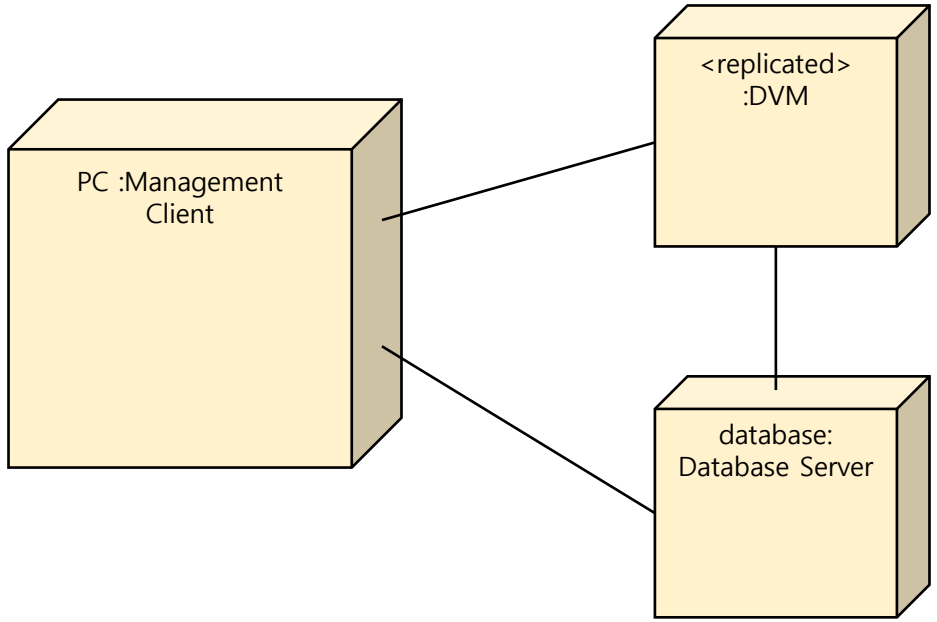


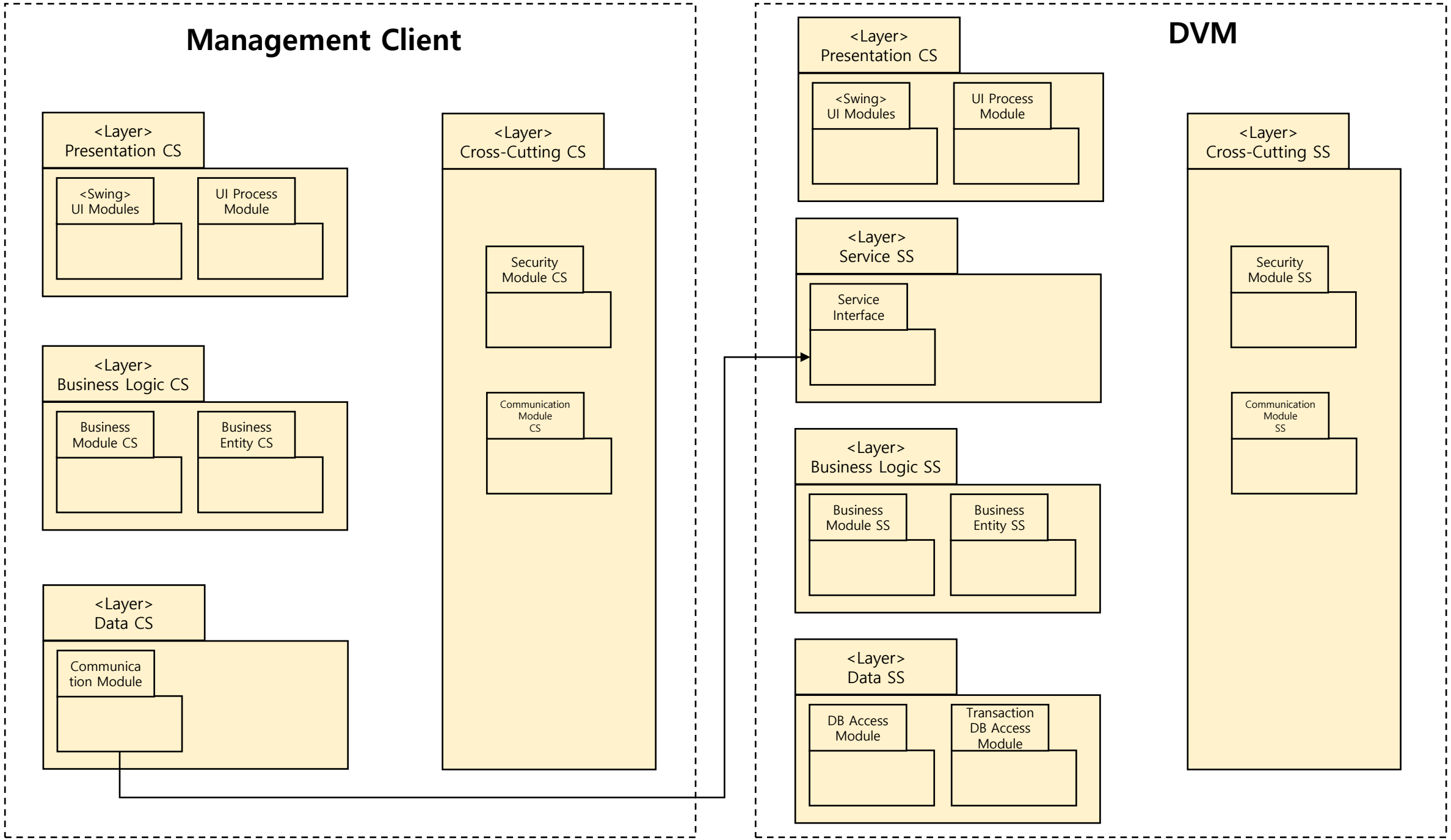
ADD Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

Design Decision and Location	Rationale
Rich Client application의 local data sources를 제거한다.	클라이언트는 Service Application과 Database에 접속하여 재고 사항과 거래 내역을 읽기 때문에 local data sources는 필요하지 않다.
Service application에 presentation layer를 추가한다.	자판기에서 사용자와 상호작용을 하기 위해 presentation layer를 추가한다.
Service application에 local data sources를 추가한다.	현재 자판기의 재고 사항을 저장할 로컬 데이터베이스를 추가한다.

ADD Step 6: Sketch Views and Record Design Decisions

Initial Deployment Diagram




Management Client
<Layer>
Presentation CS
<Swing>
UI Modules
UI Process
Module
<Layer>
Cross-Cutting CS
Security
Module CS
Communication
Module
CS
<Layer>
Business Logic CS
Business
Module CS
Business
Entity CS
<Layer>
Data CS
Communica
tion Module
DVM
<Layer>
Presentation CS
<Swing>
UI Modules
UI Process
Module
<Layer>
Service SS
Service
Interface
<Layer>
Business Logic SS
Business
Module SS
Business
Entity SS
<Layer>
Data SS
DB Access
Module
Transaction
DB Access
Module
<Layer>
Cross-Cutting SS
Security
Module SS
Communication
Module
SS

Element Description

Element	Responsibility
Presentation CS	유저와의 상호작용을 담당하는 레이어이다.
Business Logic CS	클라이언트에서 처리 가능한 Business Logic 들을 모아둔 레이어이다.
Data CS	서버의 데이터에 접근하는 레이어이다.
Cross-Cutting CS	모든 레이어에 적용되는 내용으로 보안, 로깅, I/O 등을 위해 존재하는 레이어이다.
UI Module (CS, SS)	유저의 입력을 받고 인터페이스를 렌더링 한다.
UI Process Module (CS, SS)	모든 유즈케이스에서 컨트롤 플로우를 담당한다.
Business Modules CS	내부에서 실행되는 기능들 뿐만 아니라 서버에서 실행되는 기능들을 구현한다.
Communication modules CS	서버에서 실행되는 서비스들과 통신한다.

Element	Responsibility
Presentation SS	자판기를 사용하는 유저와 상호작용을 담당하는 레이어이다.
Service SS	다른 서버들과 클라이언트에서 필요한 서비스들을 제공하는 레이어이다.
Business Logic SS	서버에서 처리해야하는 business logic을 담당하는 레이어이다.
Data SS	거래 내역과 재고사항을 저장하는데 필요한 레이어이다.
Cross-Cutting SS	다른 레이어들의 보안, 로깅, I/O 등을 담당하는 레이어이다.
Service Interface SS	DVM에서 제공하는 서비스들을 외부로 노출한다.
Business Module SS	비즈니스 작동 모듈들을 구현한다.
DB Access Module SS	현재 자판기의 재고 사항을 저장하고 있는 local database에 접근한다.
Transaction DB Access Module SS	모든 자판기들의 거래 내역을 저장하는 데이터베이스에 접근한다.

ADD Step 7: Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
	UC-2		Service application에 presentation layer를 추가하여 사용자와 상호작용할 수 있게 되었고 Rich client application을 추가하여 자판기들의 재고 사항을 확인할 수 있게 되었다.
	UC-5		Service application에 presentation layer를 추가하여 사용자와 상호작용할 수 있게 되었다.
	UC-6		Service application reference architecture를 적용하여 자판기들간 서비스 통신이 가능하게 되었다.
	UC-7		Service application reference architecture를 적용하여 자판기들간 서비스 통신이 가능하게 되었다.
QA-2			인증 코드를 위한 상세한 모듈이 정의되지 않았다.
	QA-3		Service application reference architecture를 적용하여 자판기들간 서비스 통신이 가능하게 되었다.
	QA-4		Service application reference architecture를 적용하여 자판기들간 서비스 통신이 가능하게 되었다.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
	QA-6		Rich client application을 사용하여 자판기들의 재고사항을 읽어올 수 있다.
	QA-7		Cross-Cutting에 보안을 지원하는 모듈을 추가하였다.
QA-10			데이터 정합성에 대한 검증이 적용되지 않았다.
	CON-1		Initial Deployment Diagram에 3개의 DVM device를 추가하였다.
	CON-2		Database tier를 추가하였다.
CRN-1			데이터 정합성에 대한 검증이 적용되지 않았다.
CRN-2			네트워크 안정성에 대한 검증이 적용되지 않았다.
	CRN-3		Rich client application을 사용하여 자판기들의 재고사항을 읽어올 수 있다.

ADD Iteration 2

ADD Step 2: Establish Iteration Goal by Selecting Drivers

- Iteration goal: primary functionality를 만족시킨다.

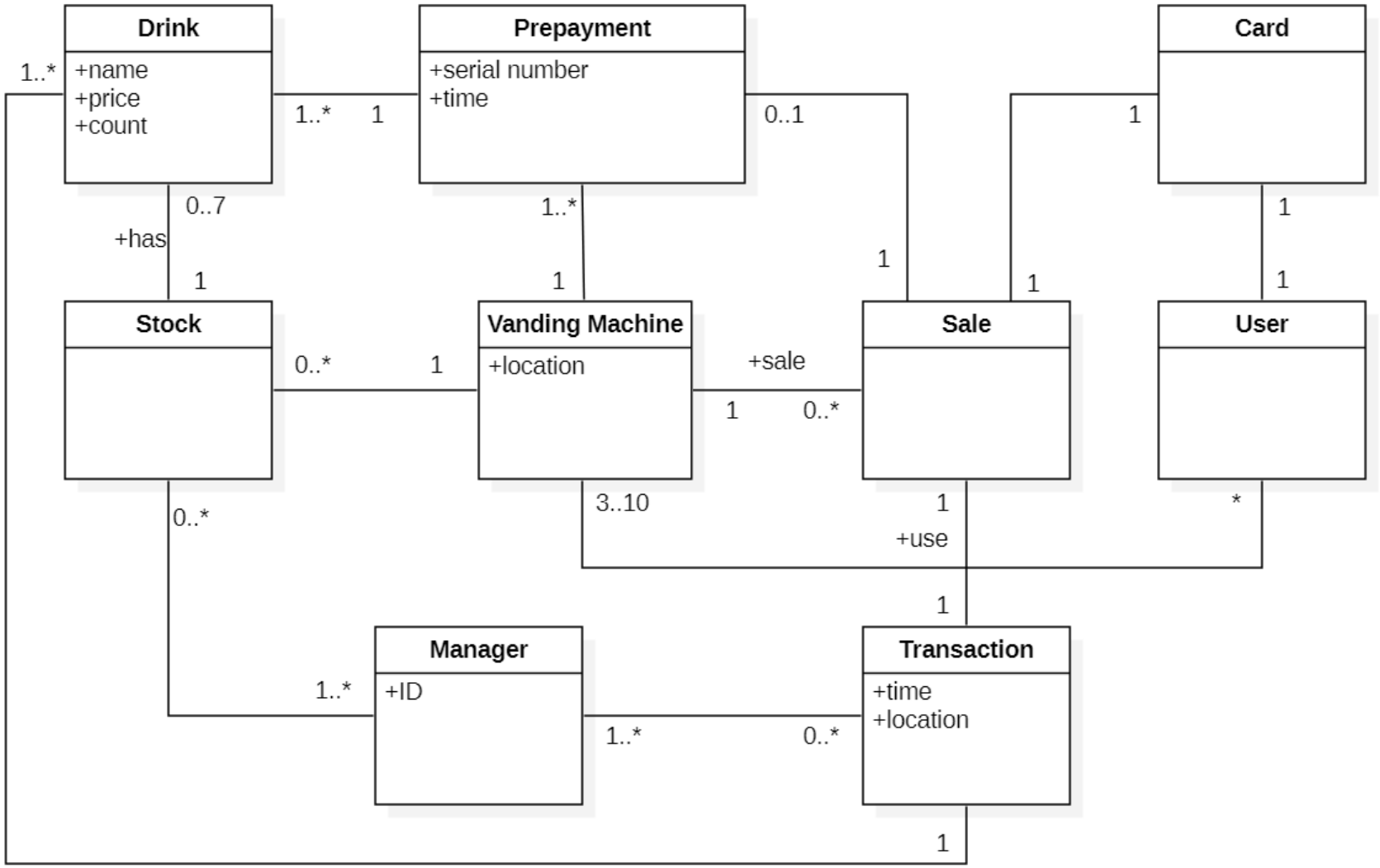
ADD Step 3: Choose One or More Elements of the System to Refine

- 두가지 reference architecture에서 정의된 레이어들과 각 레이어에 속한 모듈들을 정의하고 다른 아키텍처간 통신을 정의한다.

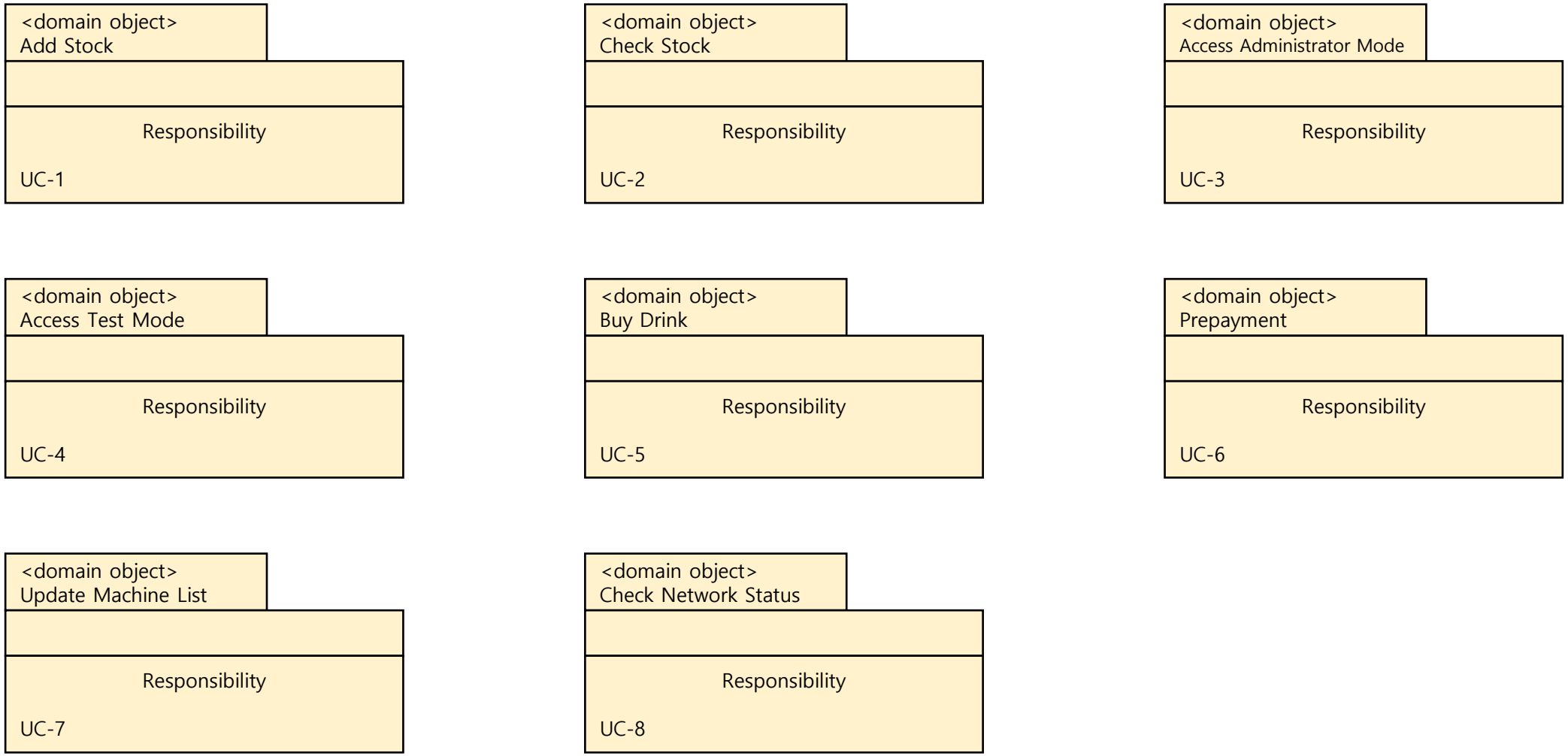
ADD Step 4: Choose One or More Design Concepts that Satisfy the Selected Drivers

- 어플리케이션을 위해 Domain Model을 만든다.
- Functional Requirement를 할당하기 위한 Domain Object를 구분한다.
- Domain Object를 일반화되고 특성화된 Component들로 분할한다.

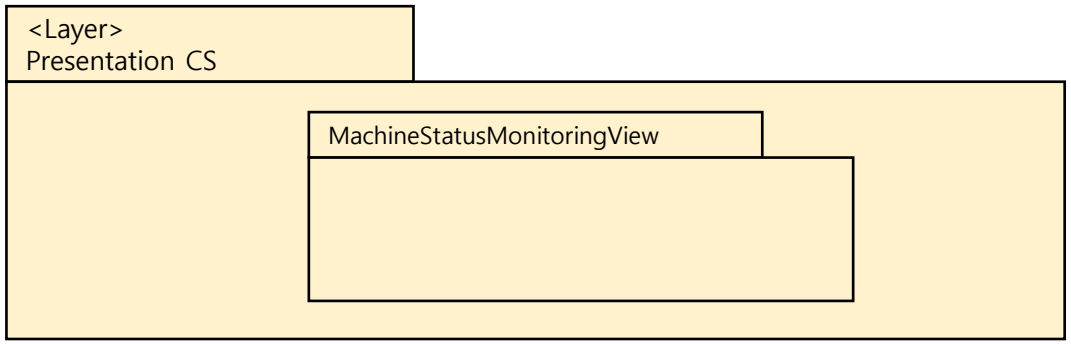
Initial Domain Model

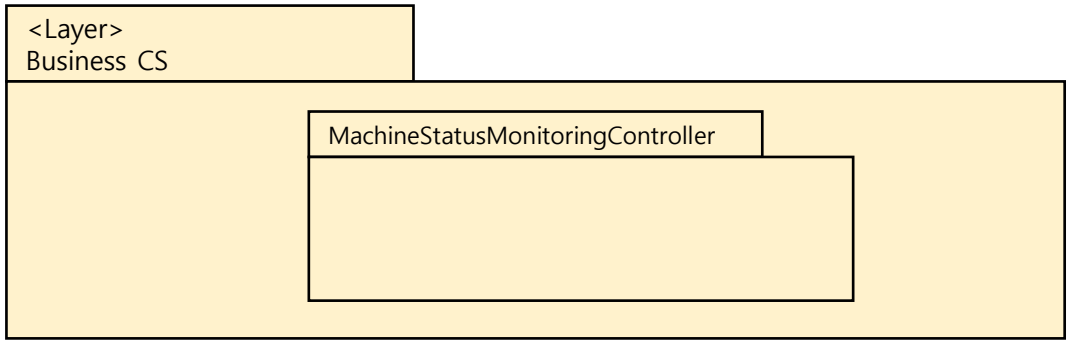


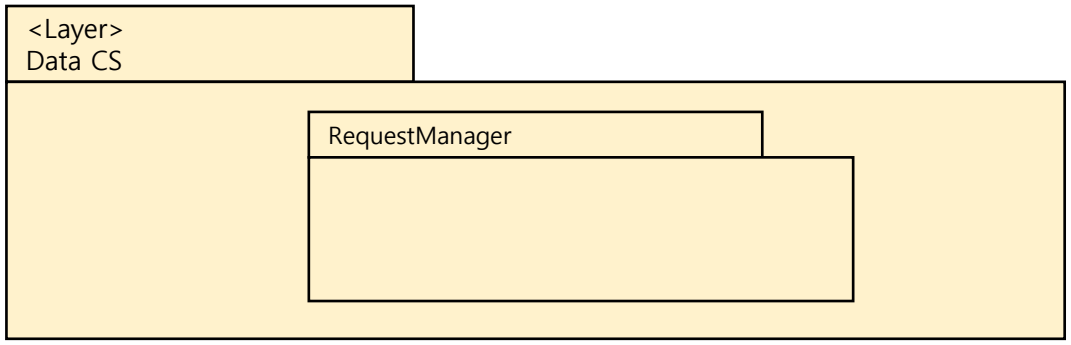
Domain Objects Associated with Use Case



Management System

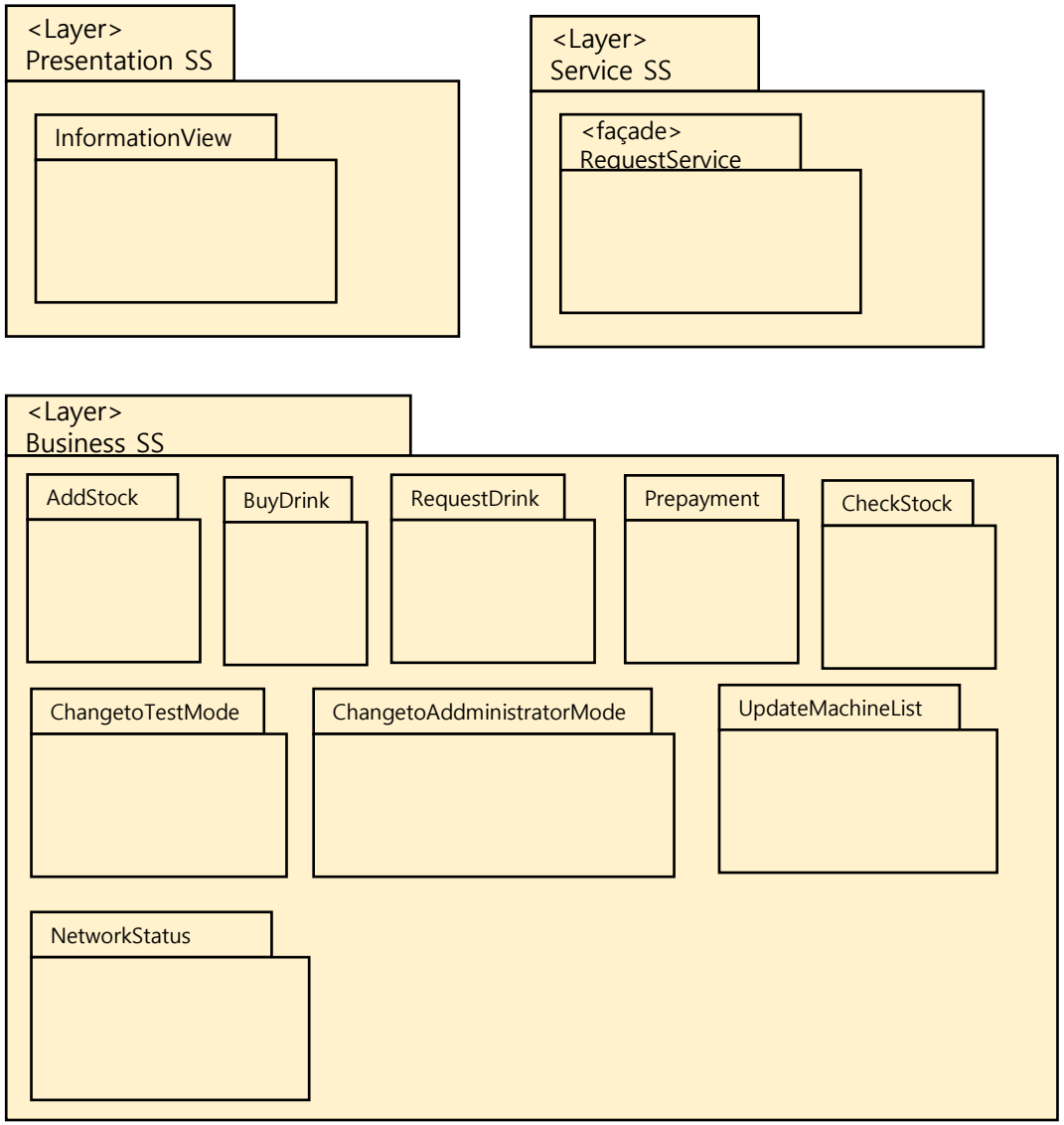




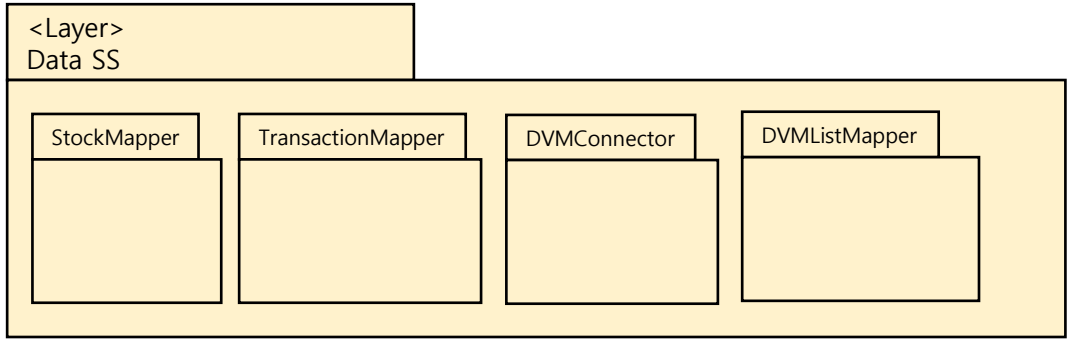


Element	Responsibility
MachineStatusMonitoringVIew	머신의 재고사항과 판매 실적을 보여준다. 사용자가 선택하면 거래 내역도 보여준다.
MachineStatusMonitoringController	재고사항을 나타내기 위해 필요한 정보들을 가공한다.
RequestManager	머신의 RequestService와 통신한다.

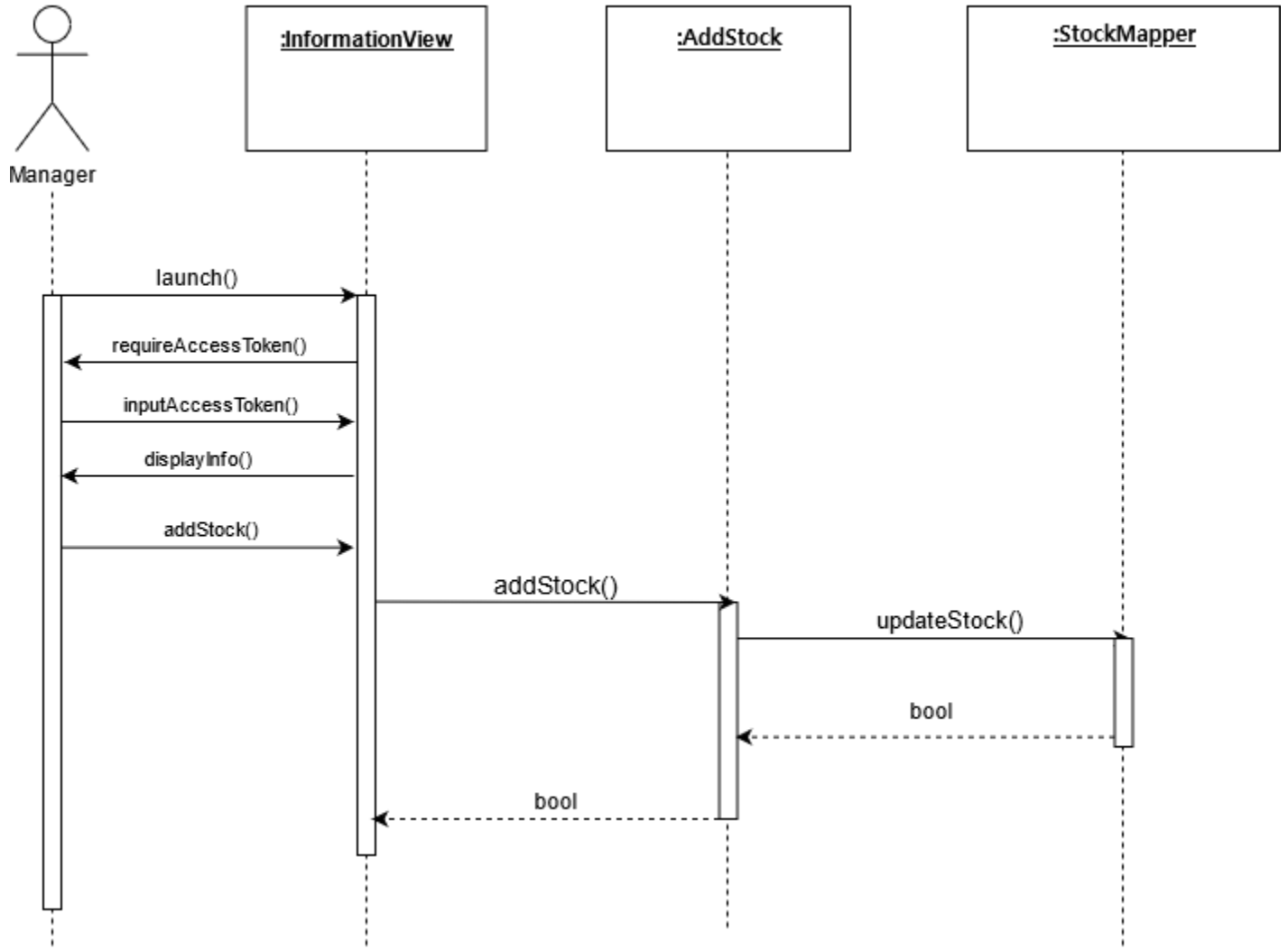
DVM



Element	Responsibility
InformationView	사용자가 음료를 구매하기 위한 정보를 표시한다.
RequestService	클라이언트로부터 들어오는 요청사항을 façade 패턴으로 처리한다.
AddStock	재고를 추가한다.
BuyDrink	음료 판매를 위한 재고 확인, 결제, 재고 상황 변경을 처리한다.
RequestDrink	현재 자판기에 재고가 없을 경우 다른 자판기에 재고 확인을 요청한다.
Prepayment	다른 자판기의 음료를 구매하는 선결제를 지원한다.
CheckStock	현재 재고 사항을 확인한다.
ChangetoTestMode	테스트모드로 진입한다.
ChangetoAddministratorMode	관리자모드로 진입한다.
UpdateMachineList	자판기 리스트를 업데이트한다.
NetworkStatus	자판기의 네트워크 상황을 확인한다.
StockMapper	자판기의 재고 데이터를 관리한다.
TransactionMapper	모든 자판기들의 거래 사항을 기록하는 데이터베이스에 접속한다.
DVMConnector	다른 자판기들과 연결한다.
DVMListMapper	현재 등록 되어있는 자판기들의 리스트를 관리한다.



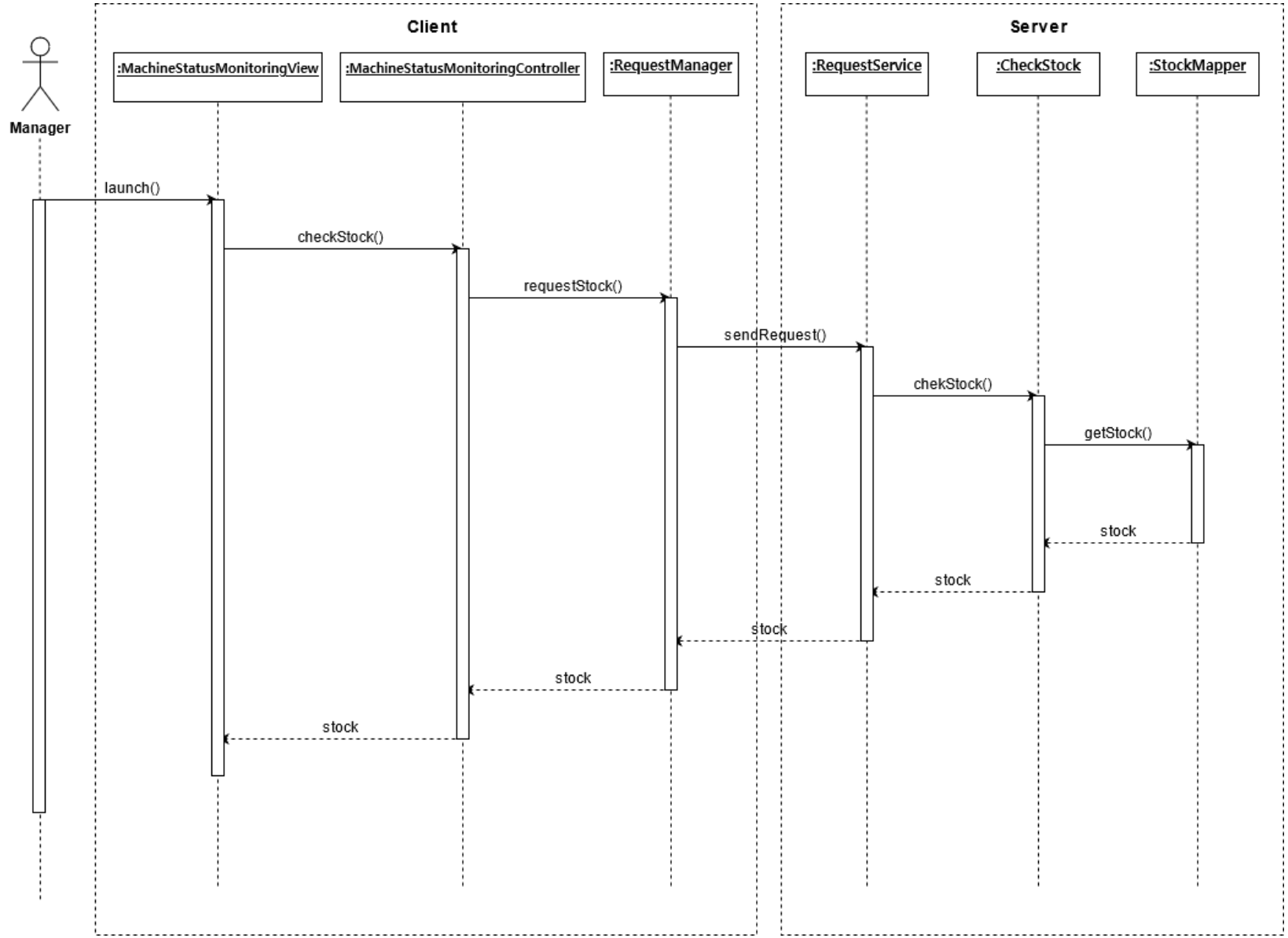
Sequence Diagram for UC-1: Add Stock



Interfaces from UC-1 Sequence Diagram

Method Name	**Description**
Element: InformationView	
requireAccessToken	접속 권한을 확인하기 위해 관리자에게 접근 토큰을 요구한다.
inputAccessToken	관리자가 접근 토큰을 입력한다.
Element: AddStock	
addStock	추가하고 싶은 재고를 입력한다.
Element: StockMapper	
updateStock	재고 사항을 업데이트 한다.

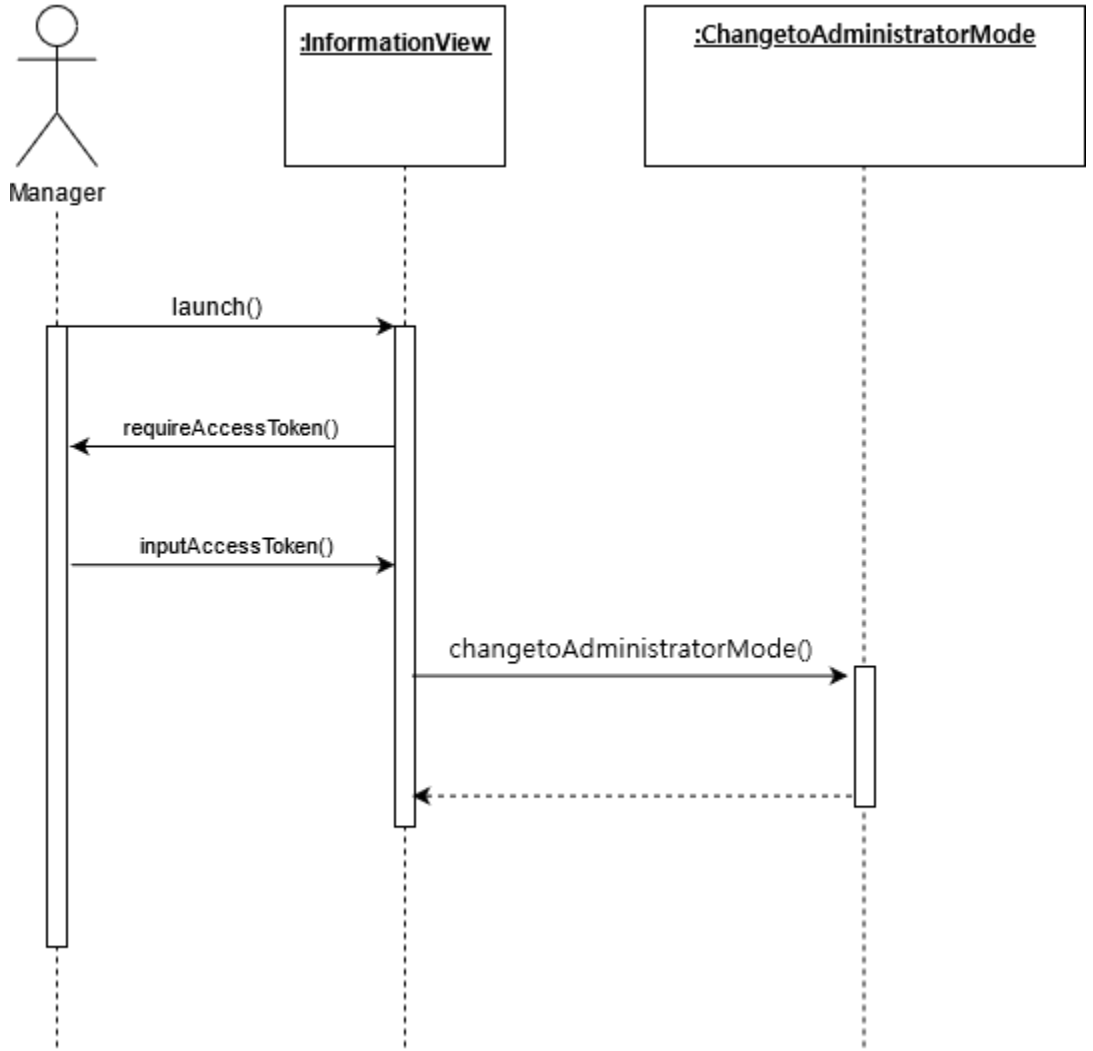
Sequence Diagram for UC-2: Check Stock



Interfaces from UC-2 Sequence Diagram

Method Name	**Description**
Element: MachineStatusMonitoringController	
checkStock	자판기의 재고사항 확인을 요청한다.
Element: RequestManager	
requestStock	재고 사항을 요청한다.
Element: RequestService	
sendRequest	서버에 요청사항을 전달한다.
Element: CheckStock	
checkStock	재고 사항을 확인을 요청한다.
Element: StockMapper	
getStock	재고 사항을 확인하여 반환한다.

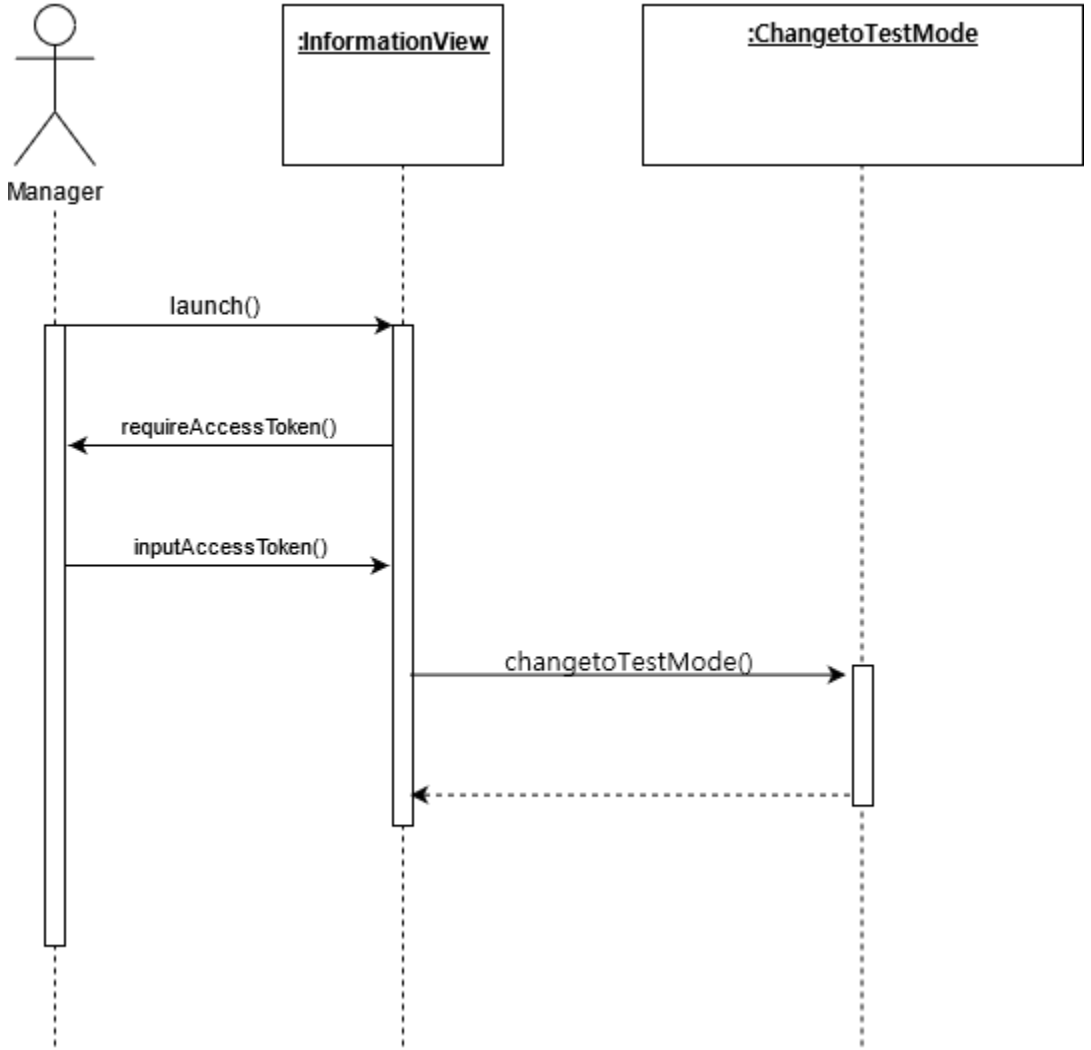
Sequence Diagram for UC-3: Access Administrator Mode



Interfaces from UC-3 Sequence Diagram

Method Name	**Description**
Element: InformationView	
requireAccessToken	접속 권한을 확인하기 위해 관리자에게 접근 토큰을 요구한다.
inputAccessToken	관리자가 접근 토큰을 입력한다.
Element: ChangetoAddministratorMode	
changetoAdministratorMode	관리자 모드로 변경한다.

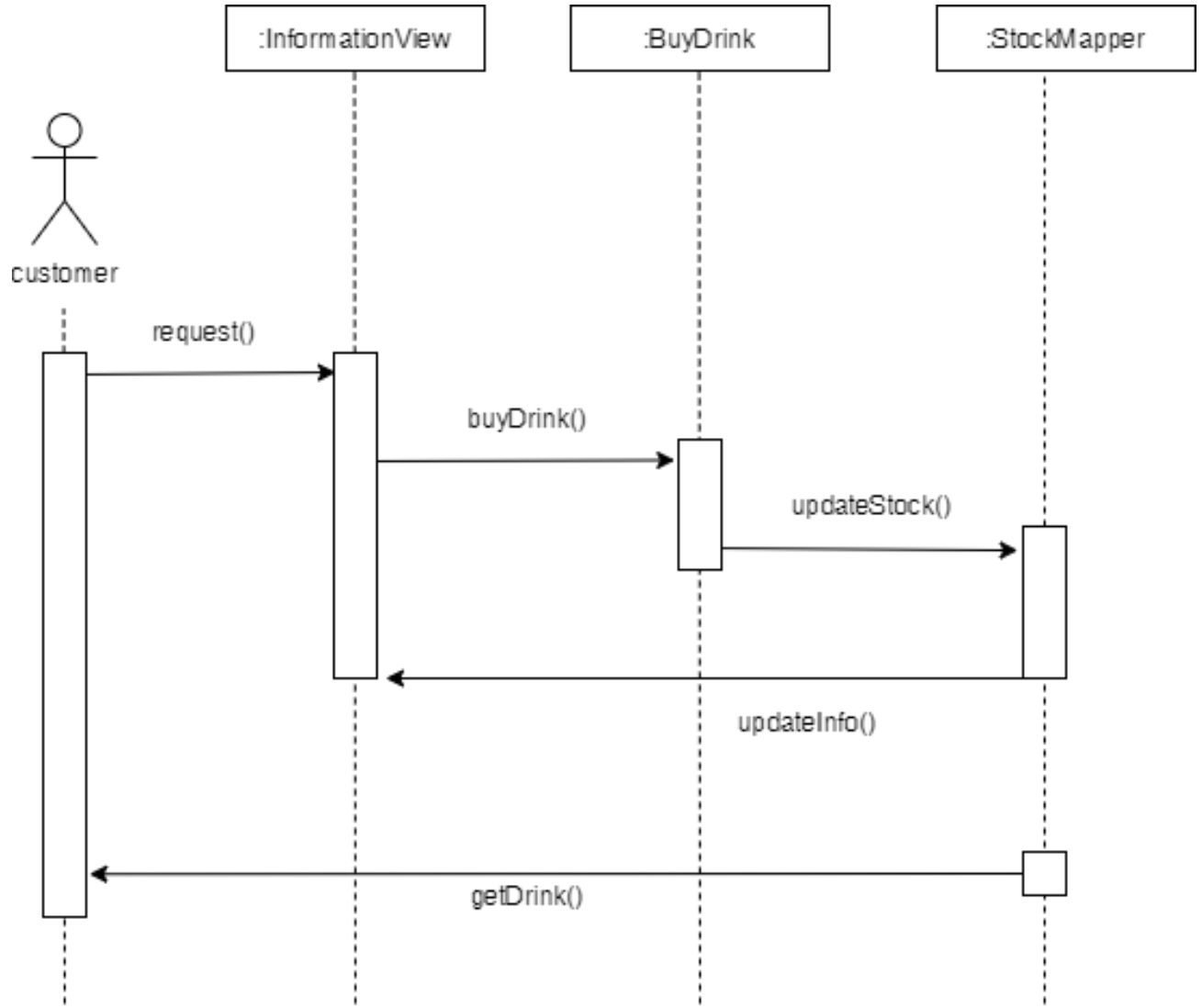
Sequence Diagram for UC-4: Access Test Mode



Interfaces from UC-4 Sequence Diagram

Method Name	**Description**
Element: InformationView	
requireAccessToken	접속 권한을 확인하기 위해 관리자에게 접근 토큰을 요구한다.
inputAccessToken	관리자가 접근 토큰을 입력한다.
Element: ChangetoAddministratorMode	
changetoAdministratorMode	테스트 모드로 변경한다.

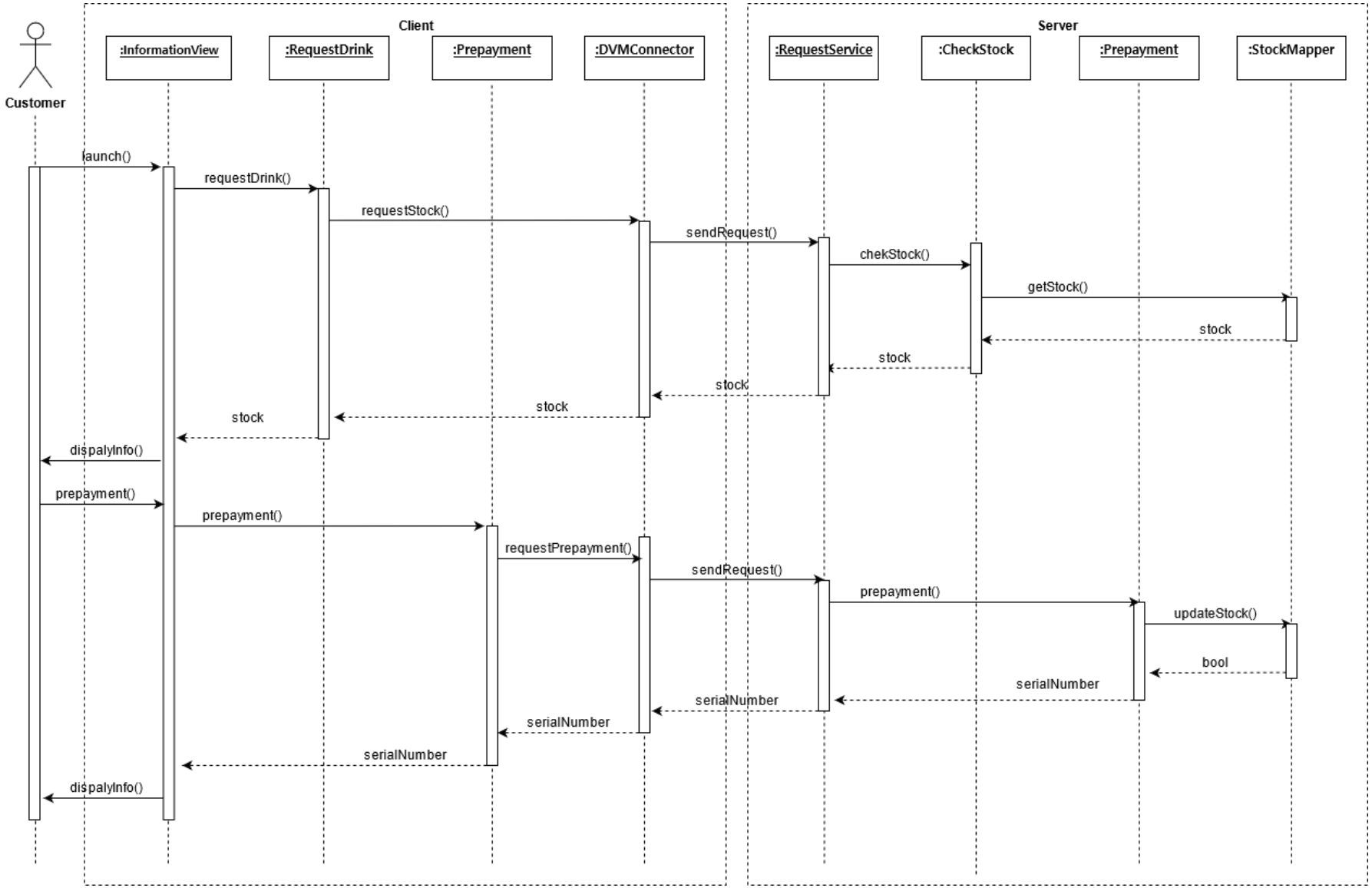
Sequence Diagram for UC-5: Buy Drink



Interfaces from UC-5 Sequence Diagram

Method Name	Description
Element: BuyDrink	
buyDrink	사용자가 음료 구매를 요청한다.
Element: StockMapper	
updateStock	변경된 재고 사항을 업데이트 한다.
getDrink	고객에게 음료를 반환한다.

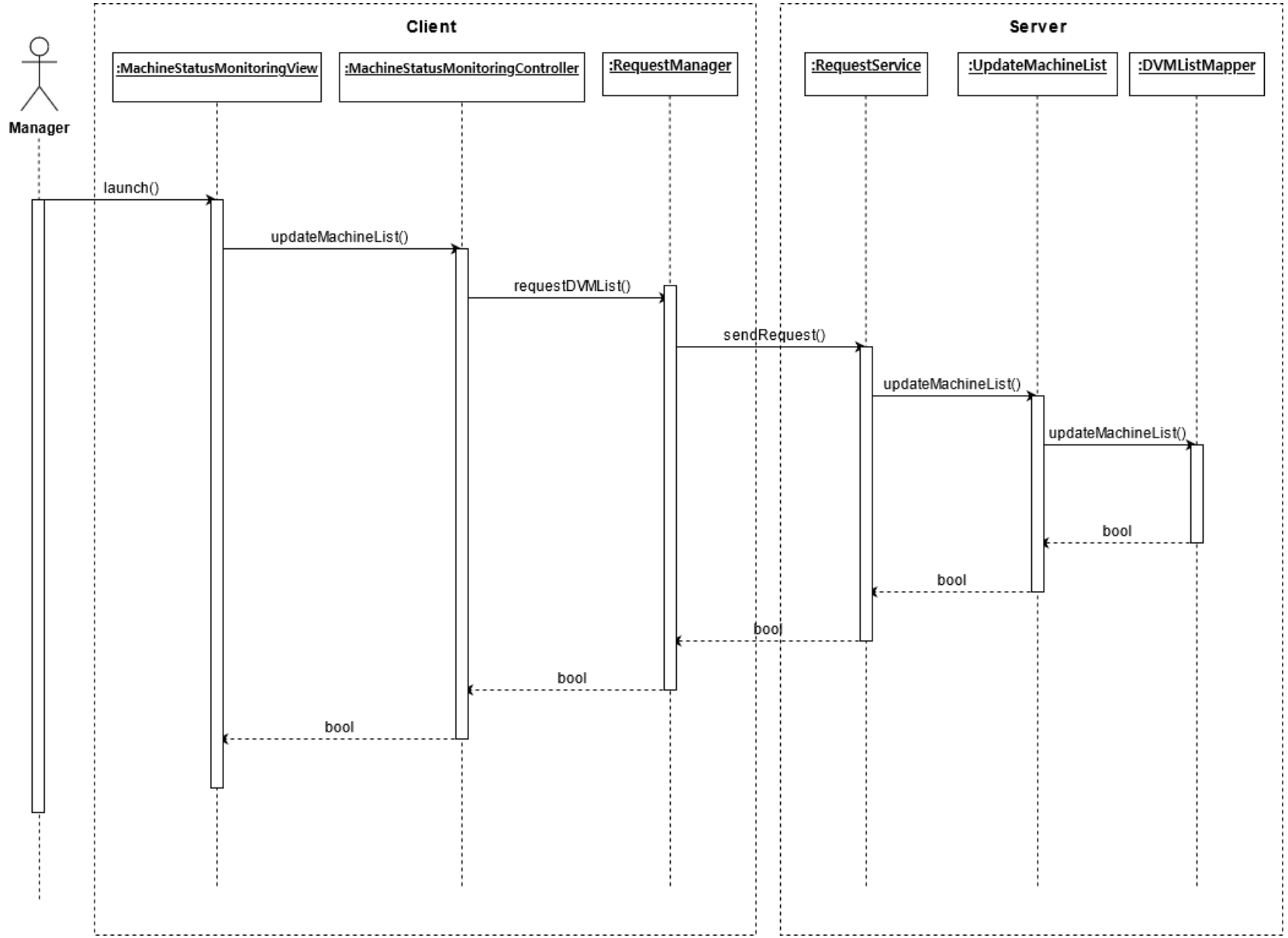
Sequence Diagram for UC-6: Prepayment



Interfaces from UC-6 Sequence Diagram

Method Name	Description
Element: InformationView	
displayInfo	확인한 자판기의 위치와 선결제 가능 상황을 표시한다.
Element: RequestDrink	
requestDrink	현재 자판기에 없는 음료를 다른 자판기에 확인 요청한다.
Element: Prepayment(Client)	
prepayment	선택한 자판기에 선결제를 요구한다.
Element: DVMConnector	
requestStock	주변 자판기에 재고 사항 확인을 요청한다.
requestPrepayment	재고가 존재하는 자판기에 선결제를 요청한다.
Element: RequestService	
sendRequest	요청받은 요구 사항을 처리한다.
Element: CheckStock	
checkStock	요청 받은 자판기의 재고 사항을 반환한다.
Element: Prepayment(Server)	
prepayment	요청 받은 선결제를 진행하고 시리얼 넘버를 반환한다.
Element: StockMapper	
getStock	재고 사항을 반환한다.
updateStock	재고 사항을 업데이트 한다.

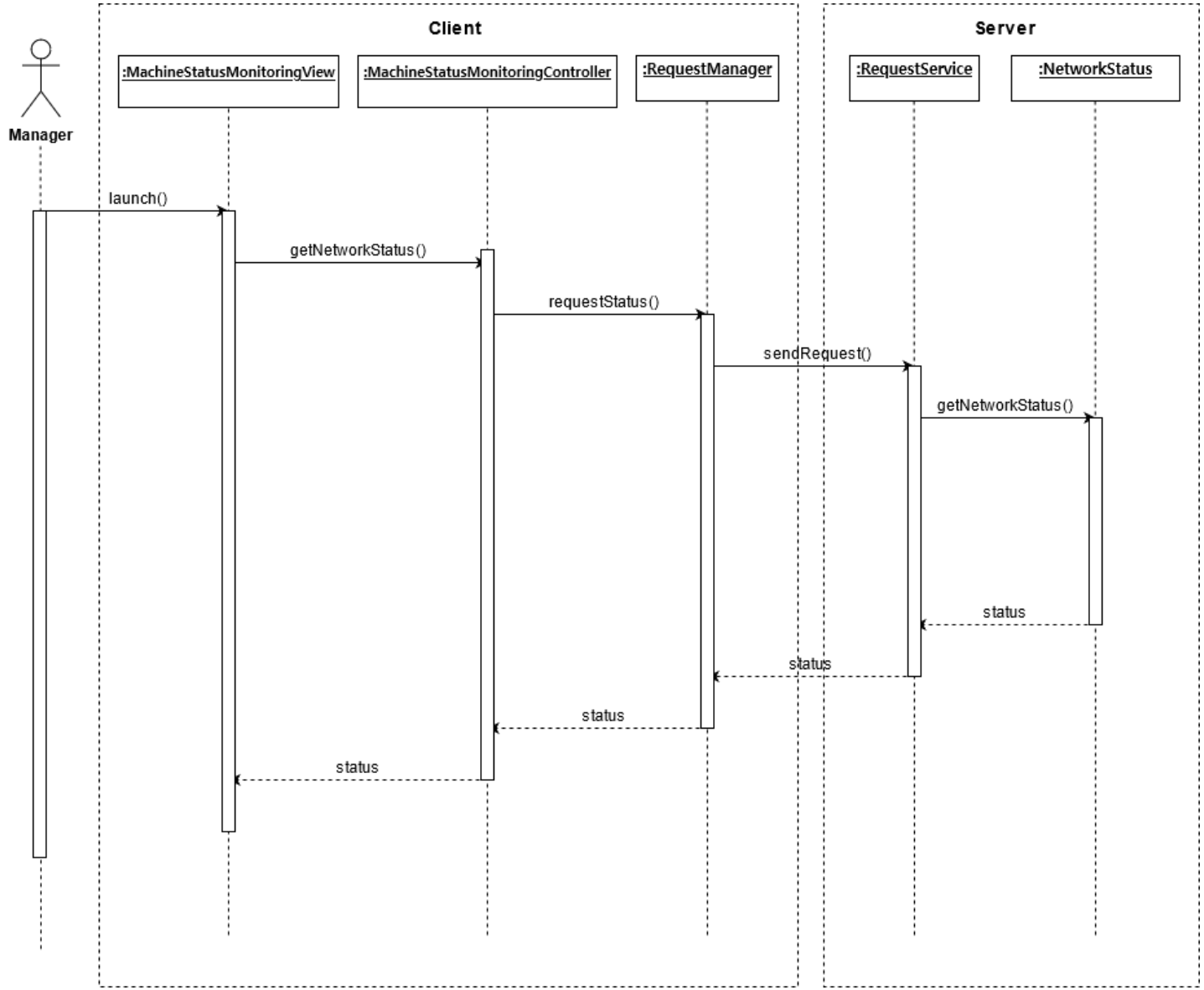
Sequence Diagram for UC-7: Update Machine List



Interfaces from UC-7 Sequence Diagram

Method Name	Description
Element: MachineStatusMonitoringController	
updateMachineList	자판기 리스트 업데이트를 요청한다.
Element: RequestManager	
requestDVMList	자판기 리스트 업데이트를 요청한다.
Element: RequestService	
sendRequest	서버에 요청사항을 전달한다.
Element: UpdateMachineList	
updateMachineList	자판기 리스트 업데이트를 요청한다.
Element: DVMListMapper	
updateMachineList	자판기 리스트를 업데이트 한다.

Sequence Diagram for UC-8: Check Network Status



Interfaces from UC-8 Sequence Diagram

Method Name	**Description**
Element: MachineStatusMonitoringController	
getNetworkStatus	원하는 자판기의 네트워크 연결사항 확인을 요청한다.
Element: RequestManager	
requestStatus	자판기의 네트워크 연결사항 확인을 요청한다.
Element: RequestService	
sendRequest	요청 사항을 받는다.
Element: NetworkStatus	
getNetworkStatus	현재 네트워크 연결 사항을 반환한다.

ADD Step 7: Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
		UC-1	이 유즈케이스를 지원하기 위한 모듈과 인터페이스가 모두 정의되었다.
		UC-2	이 유즈케이스를 지원하기 위한 모듈과 인터페이스가 모두 정의되었다.
		UC-3	이 유즈케이스를 지원하기 위한 모듈과 인터페이스가 모두 정의되었다.
		UC-4	이 유즈케이스를 지원하기 위한 모듈과 인터페이스가 모두 정의되었다.
		UC-5	이 유즈케이스를 지원하기 위한 모듈과 인터페이스가 모두 정의되었다.
		UC-6	이 유즈케이스를 지원하기 위한 모듈과 인터페이스가 모두 정의되었다.
		UC-7	이 유즈케이스를 지원하기 위한 모듈과 인터페이스가 모두 정의되었다.
		UC-8	이 유즈케이스를 지원하기 위한 모듈과 인터페이스가 모두 정의되었다.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
	QA-1		UC-6 구현으로 부분적으로 만족되었다.
	QA-2		UC-6 구현으로 부분적으로 만족되었다.
	QA-3		UC-7 구현으로 부분적으로 만족되었다.
	QA-4		UC-6 구현으로 부분적으로 만족되었다.
		QA-5	UC-8 구현으로 함께 요구사항이 충족되었다.
	QA-6		UC-2 구현으로 부분적으로 만족되었다.
QA-7			보안에 관련한 사항은 아직 구현되지 않았다.
		QA-8	UC-4 구현으로 함께 요구사항이 충족되었다.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
	QA-9		UC-5 구현으로 부분적으로 만족되었다.
	QA-10		UC-5 구현으로 부분적으로 만족되었다.
	CON-1		서버 구현으로 부분적으로 만족되었다.
		CON-2	모든 거래 내역 저장을 위한 데이터베이스 추가 및 접근 방법 정의로 만족되었다.
CRN-1			해당 사항은 아직 고려되지 않았다.
CRN-2			해당 사항은 아직 고려되지 않았다.
		CRN-3	매니지먼트 시스템 추가로 만족되었다.

ADD Iteration 3

ADD Step 2: Establish Iteration Goal by Selecting Drivers

- Iteration goal: Quality attribute(QA-1, 2, 4, 7, 10)를 만족시킨다.

ADD Step 3: Choose One or More Elements of the System to Refine

- Iteration goal을 만족시키기 위해 해당되는 컴포넌트들을 수정한다.

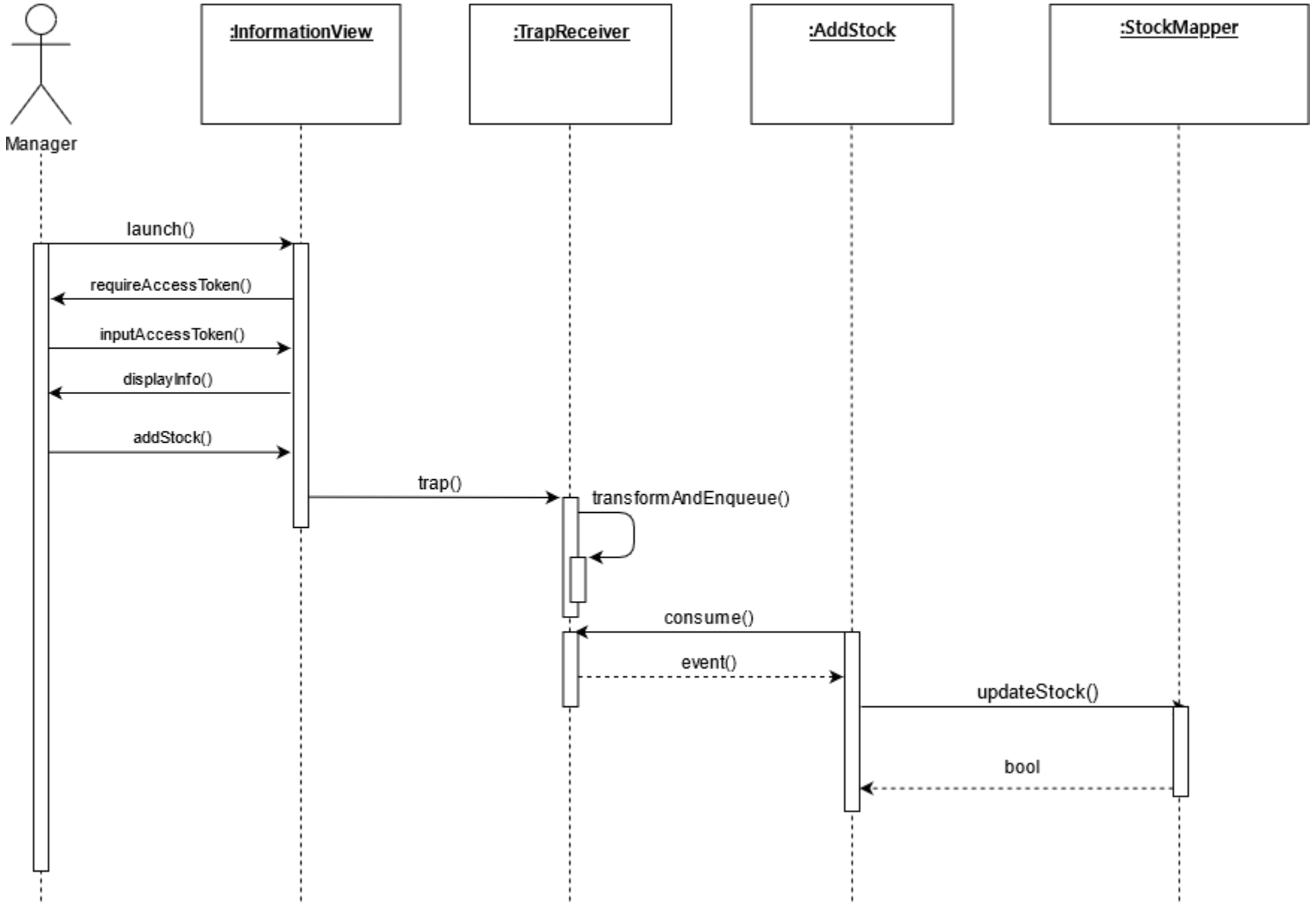
ADD Step 4: Choose One or More Design Concepts that Satisfy the Selected Drivers

Design Decisions and Location	Rationale and Assumptions
일정 시간 이상 반응이 없을 경우 원래 상태로 회복하기 위해 컴포넌트들에 Timestamp와 Rollback 기법을 사용한다.	만약 서버로부터 일정 시간동안 반응이 없을 경우 대기 화면에서 이전 화면으로 돌아가 사용자가 다시 선택이 가능하도록 이전 상태로 되돌린다. (QA-1, 2, 4)
결제 컴포넌트들에 Encrypt Data 기법을 적용한다.	사용자의 결제 정보를 안전하게 전송 및 저장하기 위해 결제 정보를 모두 암호화 한다. (QA-7)
재고 확인 및 추가 컴포넌트에 Message queue, Prioritize events 기법을 적용한다.	여러 자판기에서 재고 확인 요청 및 선결제 요청이 오더라도 정확한 재고 수량 표기를 위해 요청 순서대로 처리 가능한 message queue 기법을 사용한다. (QA-10)

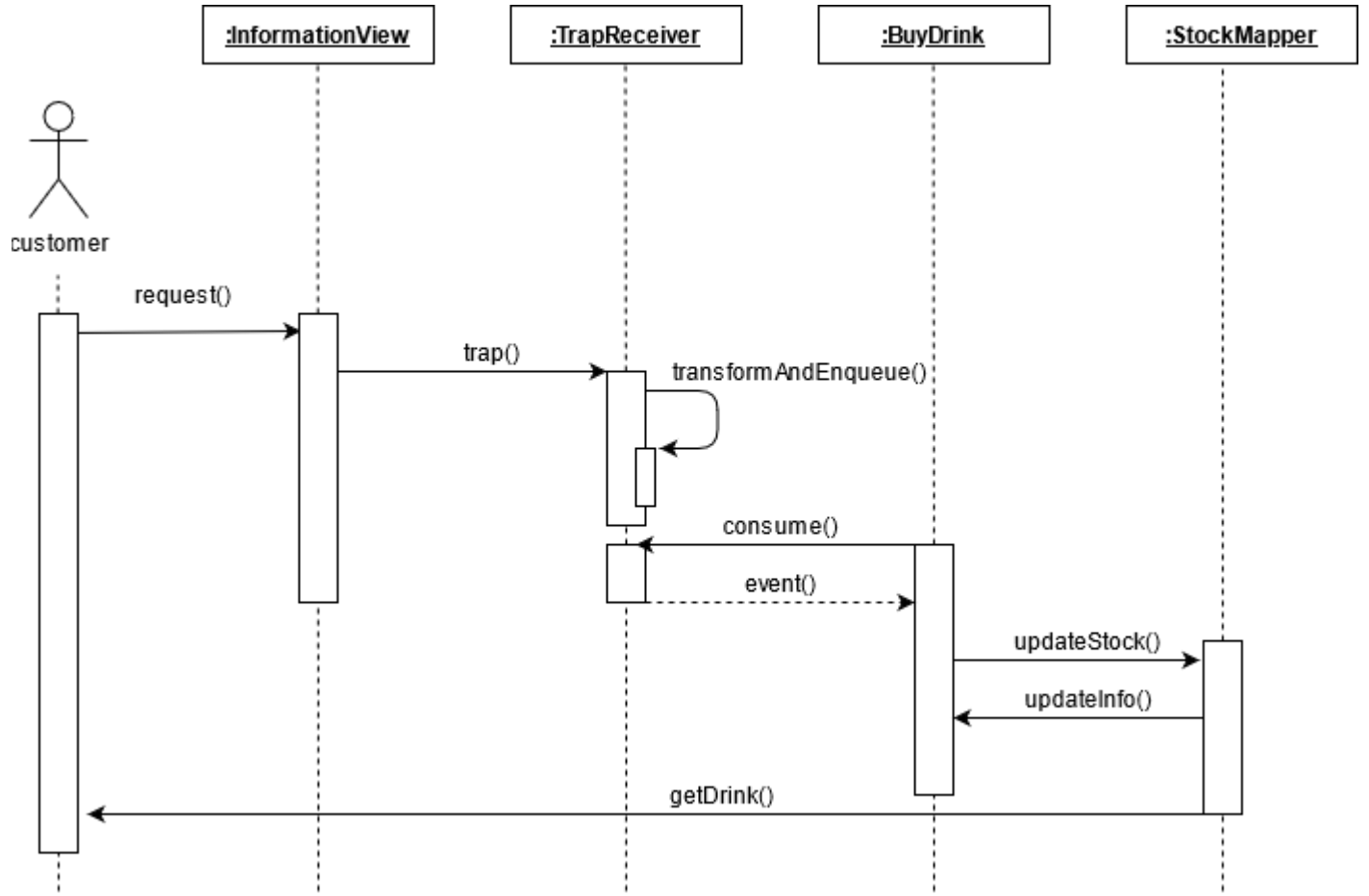
ADD Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

Design Decisions and Location	Rationale and Assumptions
Request, Prepayment 컴포넌트에 Timestamp 기법을 적용한다.	Request, Prepayment 컴포넌트에 Timestamp 기법을 적용하여 정해진 시간보다 요청 응답이 길어질 경우 요청을 취소시키고 이전 상태로 되돌린다.
Security 컴포넌트를 추가하여 Encrypt Data 기법을 적용시킨다.	BuyDrink, Prepayment 컴포넌트에서 사용자가 결제 정보를 입력하였을 때 Security 컴포넌트를 통하여 결제 정보를 암호화 및 복호화 한다.
StockMapper 컴포넌트를 호출하기 전 항상 모든 컴포넌트들은 TrapReceiver를 호출하여 Message queue, Prioritize events 기법이 적용되게 한다.	음료 구매 및 추가 상황에서 항상 정확한 재고 사항을 보여주기 위해 Message queue와 Prioritize events 기법을 적용한다. 선결제 요청이 올 경우 결제 요청을 받은 순서대로 처리하며 현장에서 구매 및 재고 추가는 항상 우선적으로 처리한다. 단순 재고 확인(UC-1)에는 적용하지 않는다.

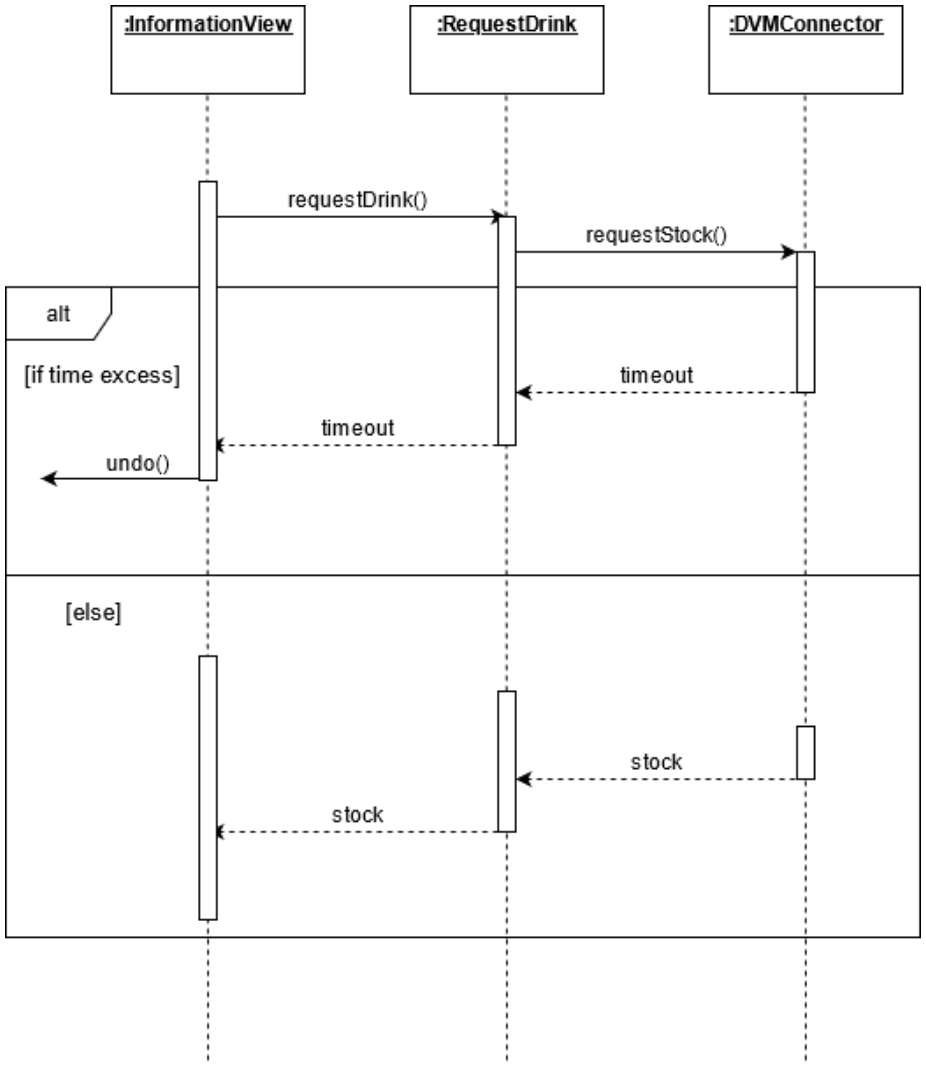
Sequence Diagram for UC-1: Add Stock

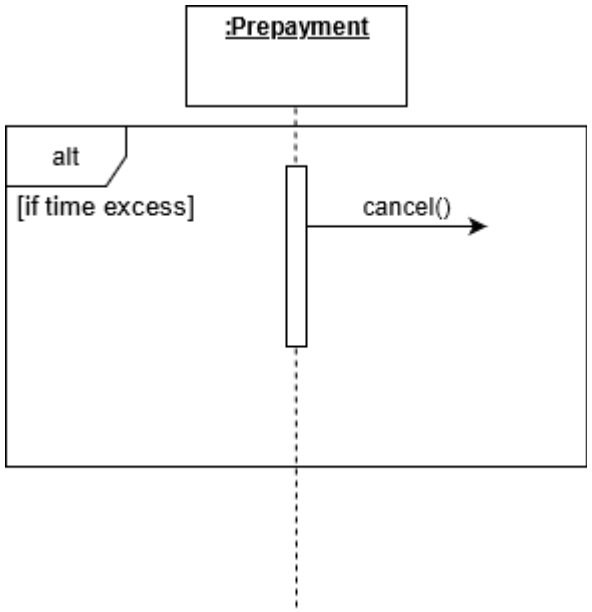


Sequence Diagram for UC-5: Buy Drink

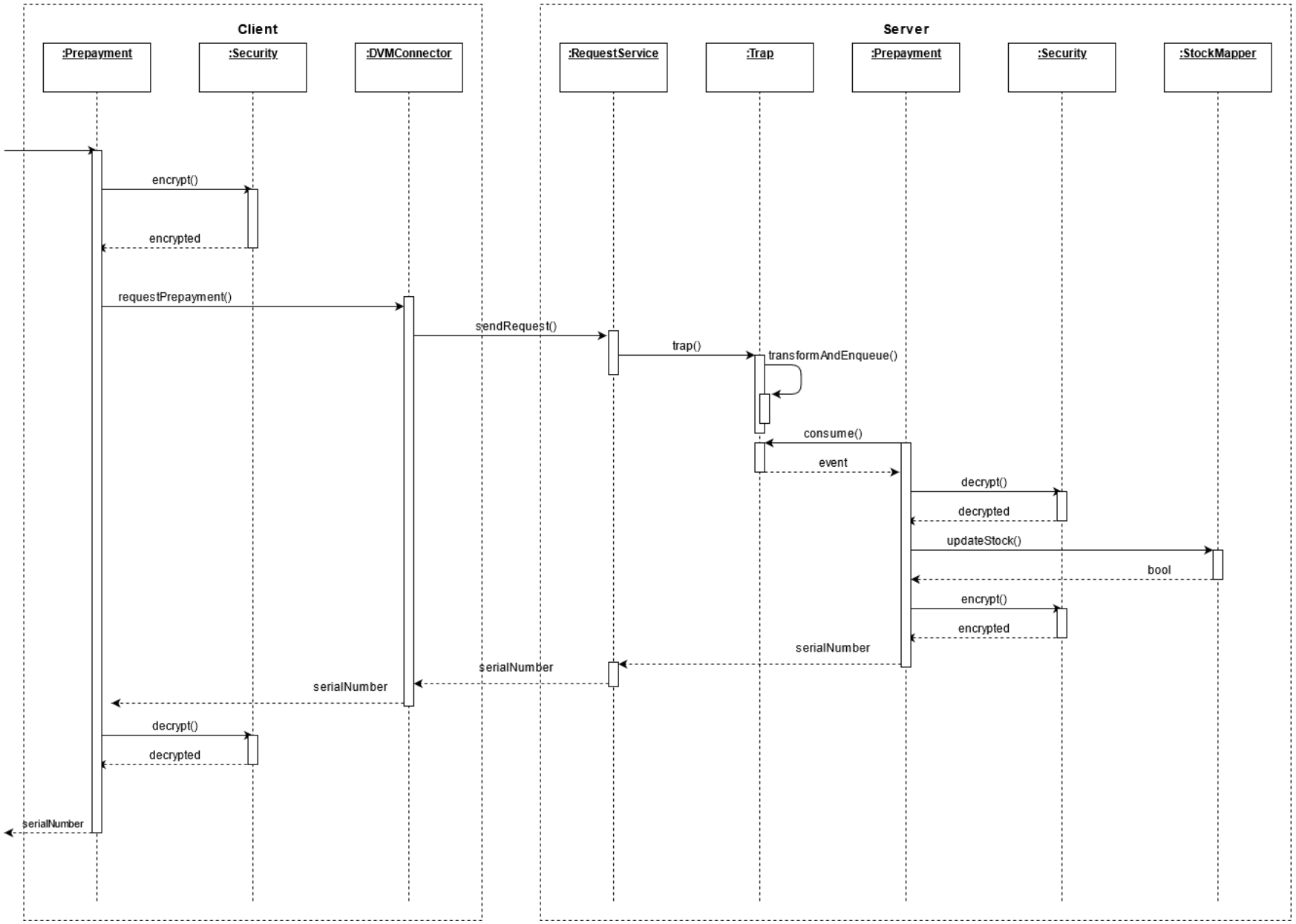


Sequence Diagram for UC-6: Prepayment





Sequence Diagram for UC-6: Prepayment



ADD Step 7: Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
		QA-1	Timestamp 기법을 적용하여 quality attribute를 만족하였다.
		QA-2	Timestamp 기법을 적용하여 quality attribute를 만족하였다.
	QA-3		UC-7 구현으로 부분적으로 만족되었다.
		QA-4	Timestamp 기법을 적용하여 quality attribute를 만족하였다.
	QA-6		UC-2 구현으로 부분적으로 만족되었다.
		QA-7	Encrypt Data 기법을 적용하여 quality attribute를 만족하였다.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
	QA-9		UC-5 구현으로 부분적으로 만족되었다.
		QA-10	Message queue, Prioritize events 기법을 적용하여 quality attribute를 만족하였다.
	CON-1		서버 구현으로 부분적으로 만족되었다.
		CRN-1	Message queue, Prioritize events 기법을 적용하여 데이터 무결성을 만족하였다.