NEW C.S.E. ATM

OOPT Stage 2040

Project Team

T3

Date

2018-05-11

Team Information

201311299 이원오

201311301 이재규

201311309 전홍준

INDEX

1. Activity 2041. Design Real Use Cases

- 1. Deposit

Use Case	1. Deposit
Actor	User
Purpose	사용자가 원하는 계좌에 입금을 한다.
Overview	User가 입금버튼을 누른다.
Type	Primary and Essential
Cross Reference	Functional Requirements : R.1
Pre-Requisites	N/A
Typical courses of events	(A) : Actor, (S) : System 1. (A) User가 프로그램(UI)에서 Deposit버튼을 누른다. 2. (S) User에게 Deposit UI를 띄워준다. 3. (A) 입금할 계좌(or 카드번호)와 비밀번호를 프로그램에 입력한다. 4. (S) 시스템에서 해당 계좌의 비밀번호를 확인한다. 5. (S) 금액 입력창으로 UI 변경 6. (A) User가 입금할 금액을 입력한다. 7. (S) User가 입력한 금액을 해당 계좌나 카드에 추가한다. 8. (S) 변경 내용을 저장한다. 9. (S) 변경 내용을 User에게 보여준다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	1. (S) 잘못된 계좌정보를 입력하거나 입금금액이 일정 금액(100만원)을 초과하면 알림을 띄운다.

- 2. Withdraw

Use Case	2. Withdraw
Actor	User
Purpose	사용자가 원하는 계좌에서 출금을 한다.
Overview	User가 출금버튼을 누른다.
Type	Primary and Essential
Cross Reference	Functional Requirements : R.2,
Pre-Requisites	N/A
Typical courses of events	(A) : Actor, (S) : System 1. (A) User가 프로그램(UI)에서 Withdraw버튼을 누른다. 2. (S) User에게 Withdraw UI를 띄워준다. 3. (A) 출금할 계좌(or 카드번호)와 비밀번호를 프로그램에 입력한다. 4. (S) 시스템에서 해당 계좌의 비밀번호를 확인한다. 5. (S) 금액 입력창으로 UI 변경 6. (A) User가 출금할 금액을 입력한다. 7. (S) User가 입력한 금액을 해당 계좌나 카드에서 빼준다. 8. (S) 변경 내용을 저장한다. 9. (S) 변경 내용을 User에게 보여준다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	1. (S) 잘못된 계좌정보 또는 비밀번호를 입력하거나 출금금액이 한계 금액(100만원)이나 계좌 잔액을 초과하면 알림을 띄운다.

- 3. Transfer

Use Case	3. Transfer
Actor	User
Purpose	사용자가 원하는 계좌로 자신의 계좌에서 돈을 이체한다.
Overview	User가 계좌이체 버튼을 누른다.
Type	Primary and Essential
Cross Reference	Functional Requirements : R.3
Pre-Requisites	N/A
Typical courses of events	(A) : Actor, (S) : System 1. (A) User가 프로그램(UI)에서 Transfer버튼을 누른다. 2. (S) User에게 Transfer UI를 띄워준다. 3. (A) 출금할 계좌(or 카드번호)와 비밀번호를 프로그램에 입력한다. 4. (S) 시스템에서 해당 계좌의 비밀번호를 확인한다. 5. (S) 이체 계좌정보 및 금액 입력창으로 UI 변경 6. (A) User가 이체할 계좌정보와 이체할 금액을 입력한다. 7. (S) User가 입력한 금액을 이체 계좌나 카드에 더해주고 해당 계좌에서는 그 금액만큼 빼준다. 8. (S) 변경 내용을 저장한다. 9. (S) 변경 내용을 User에게 보여준다.
Alternative Courses of Events	
Exceptional Courses of Events	1. (S) 잘못된 계좌정보 또는 비밀번호를 입력하거나 이체금액이 한계 금액(100만원)이나 계좌 잔액을 초과하면 알림을 띄운다.

- 4. Check Balance

Use Case	4. Check Balance
Actor	User
Purpose	사용자가 원하는 계좌의 잔액을 보여준다.
Overview	User가 잔액확인 버튼을 누른다.
Type	Primary and Essential
Cross Reference	Functional Requirements : R.4.
Pre-Requisites	N/A
Typical courses of events	(A) : Actor, (S) : System 1. (A) User가 프로그램(UI)에서 Check Balance버튼을 누른다. 2. (S) User에게 Check Balance UI를 띄워준다. 3. (A) 확인할 계좌(or 카드번호)와 비밀번호를 프로그램에 입력한다. 4. (S) 시스템에서 해당 계좌의 비밀번호를 확인한다. 5. (S) 해당계좌나 카드의 잔액을 화면에 띄운다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	1. (S) 잘못된 계좌정보 또는 비밀번호를 입력하면 알림을 띄운다.

-

- 5. Robbery Report

Use Case	5. Robbery Report
Actor	User
Purpose	사용자가 도난 신고를 한다.
Overview	User가 도난 신고 버튼을 누른다.
Type	Primary and Real
Cross Reference	Functional Requirements : R.5.
Pre-Requisites	N/A
Typical courses of events	(A) : Actor, (S) : System 1. (A) User가 프로그램(UI)에서 도난 신고버튼을 누른다. 2. (S) User에게 도난신고 UI를 띄워준다. 3. (A) 도난 신고할 계좌(or 카드번호)와 비밀번호를 프로그램에 입력한다. 4. (S) 시스템에서 해당 계좌의 비밀번호를 확인한다. 5. (S) 계좌정보(카드)를 도난 상태로 변경.
Alternative Courses of Events	N/A
Exceptional Courses of Events	1. (S) 잘못된 계좌번호나 비밀번호를 입력했을 때 화면에 알림을 띄운다.

- 6. Add Cash

Use Case	6. Add Cash
Actor	Admin
Purpose	관리자가 ATM에 현금을 추가한다.
Overview	관리자가 비밀번호를 입력해서 현금을 추가한다.
Type	Primary and Essential
Cross Reference	Functional Requirements : R.6.
Pre-Requisites	Actor가 Admin이어야한다.
Typical courses of events	(A) : Actor, (S) : System 1. (A) 관리자가 Admin버튼을 누른다. 2. (S) 관리자 확인 UI로 바꿔준다. 3. (A) 관리자 ID/PW를 입력한다. 4. (S) 관리자가 입력한 ID/PW가 올바르게 입력된지 확인한다. 5. (S) 관리자 UI로 바꿔준다. 6. (A) Add Cash 버튼을 누른다. 7. (S) Add Cash UI로 바꾼다. 8 (A) 관리자가 추가할 현금 액수를 입력한다. 9. (S) 입력한 액수만큼 잔액에 더해준다. 10. (S) 변경내용을 저장한다.
Alternative Courses of Events	
Exceptional Courses of Events	1. (S) 관리자 모드 정보(아이디와 비밀번호)를 잘못 입력하면 접근이 되지 않는다.

- 7. Check Cash

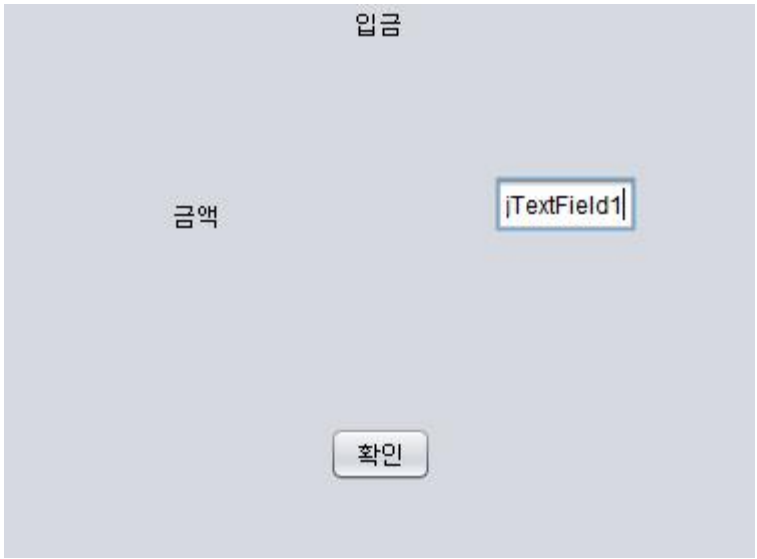
Use Case	7. Check Cash
Actor	Admin
Purpose	ATM의 현금 잔액을 확인한다.
Overview	관리자 모드에서 ATM잔액확인을 누른다.
Type	Primary and Essential
Cross Reference	Functional Requirements : R.7.
Pre-Requisites	Actor가 Admin이어야한다.
Typical courses of events	(A) : Actor, (S) : System 1. (A) 관리자가 Admin버튼을 누른다. 2. (S) 관리자 확인 UI로 바꿔준다. 3. (A) 관리자 ID/PW를 입력한다. 4. (S) 관리자가 입력한 ID/PW가 올바르게 입력된지 확인한다. 5. (S) 관리자 UI로 바꿔준다. 6. (A) Check Cash 버튼을 누른다. 7. (S) ATM의 잔액을 화면에 띄운다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	1. (S) 관리자 모드 비밀번호를 잘못 입력하면 접근이 되지 않는다.

2. Activity 2042. Define Reports, UI, and Storyboards

- 1. Main 화면

- 2. Deposit

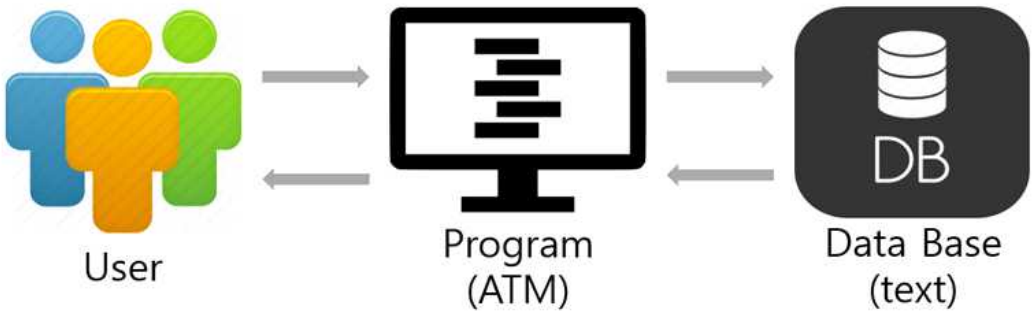
- 3. Amount (금액 확인)



- 4. Confirm (입금 확인)

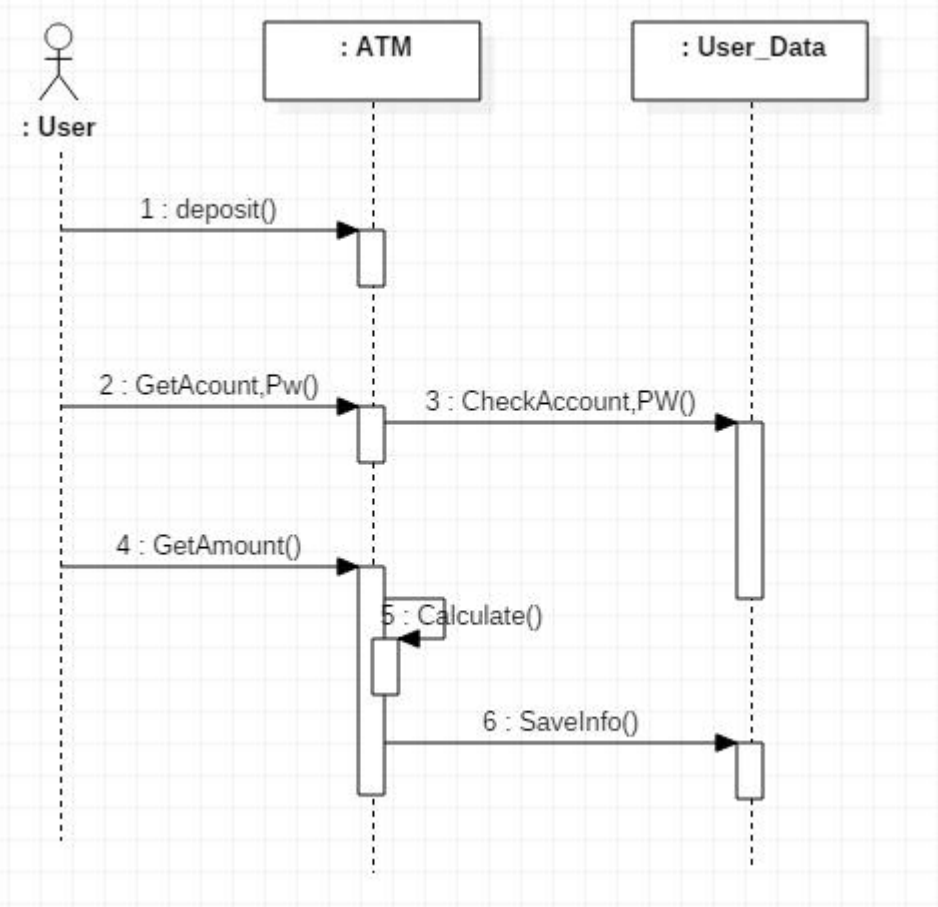
- 5. Complete (종료)

3. Activity 2043. Refine System Architecture

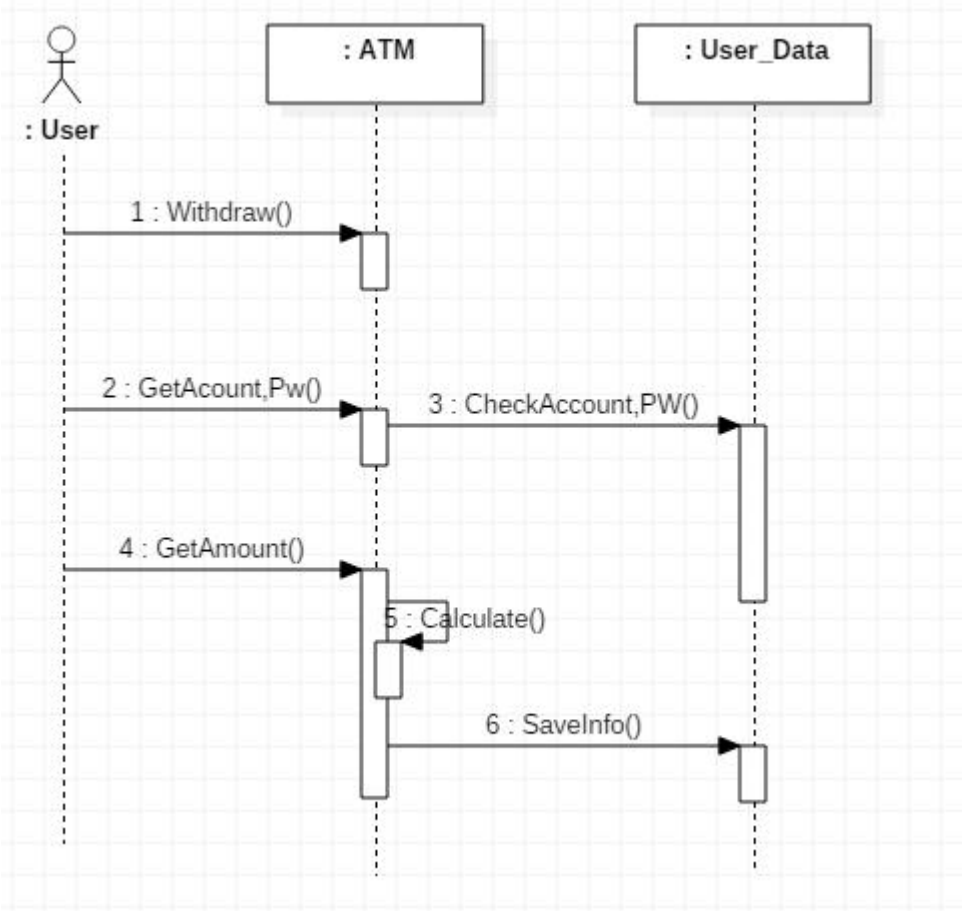


4. Activity 2044. Define Interaction Diagrams

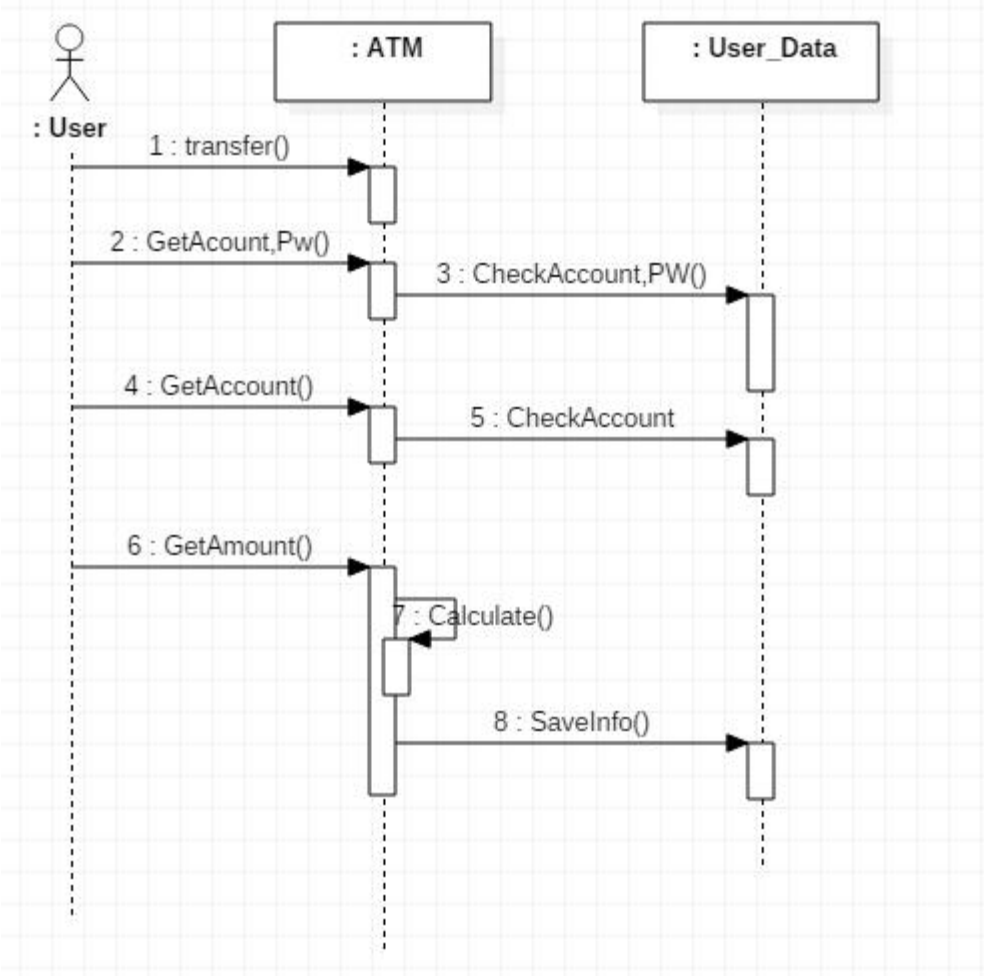
- Deposit



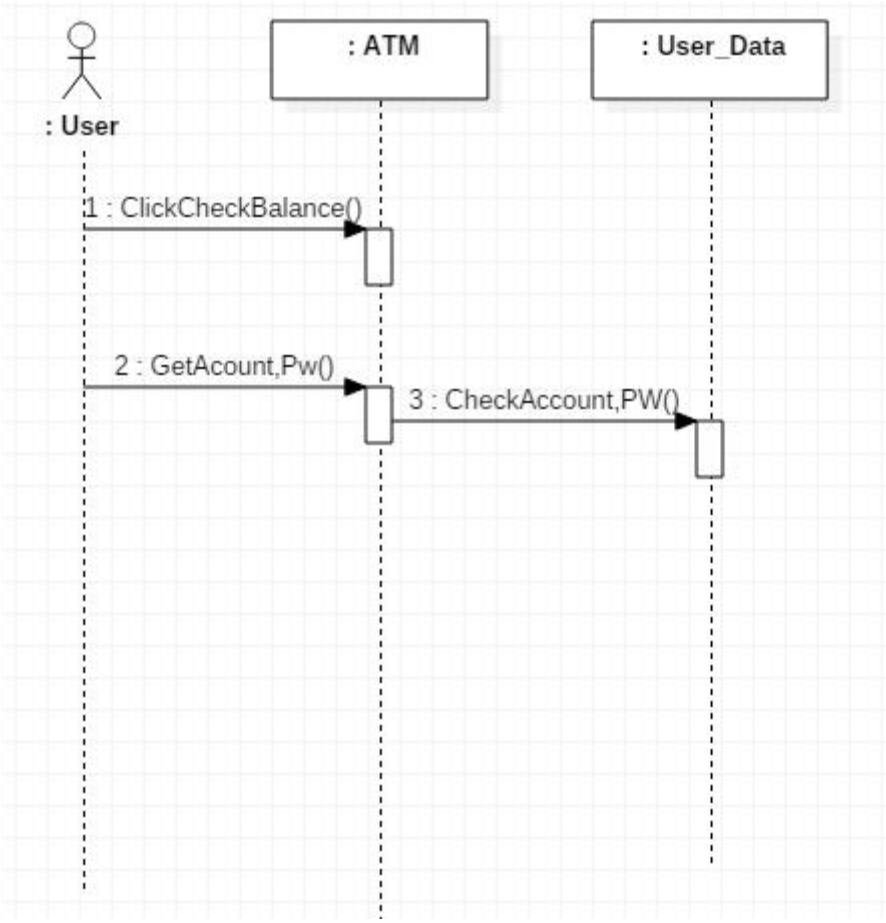
- Withdraw


: ATM
: User_Data
: User
1 : Withdraw()
2 : GetAcount,Pw()
3 : CheckAccount,PW()
4 : GetAmount()
5 : Calculate()
6 : SaveInfo()

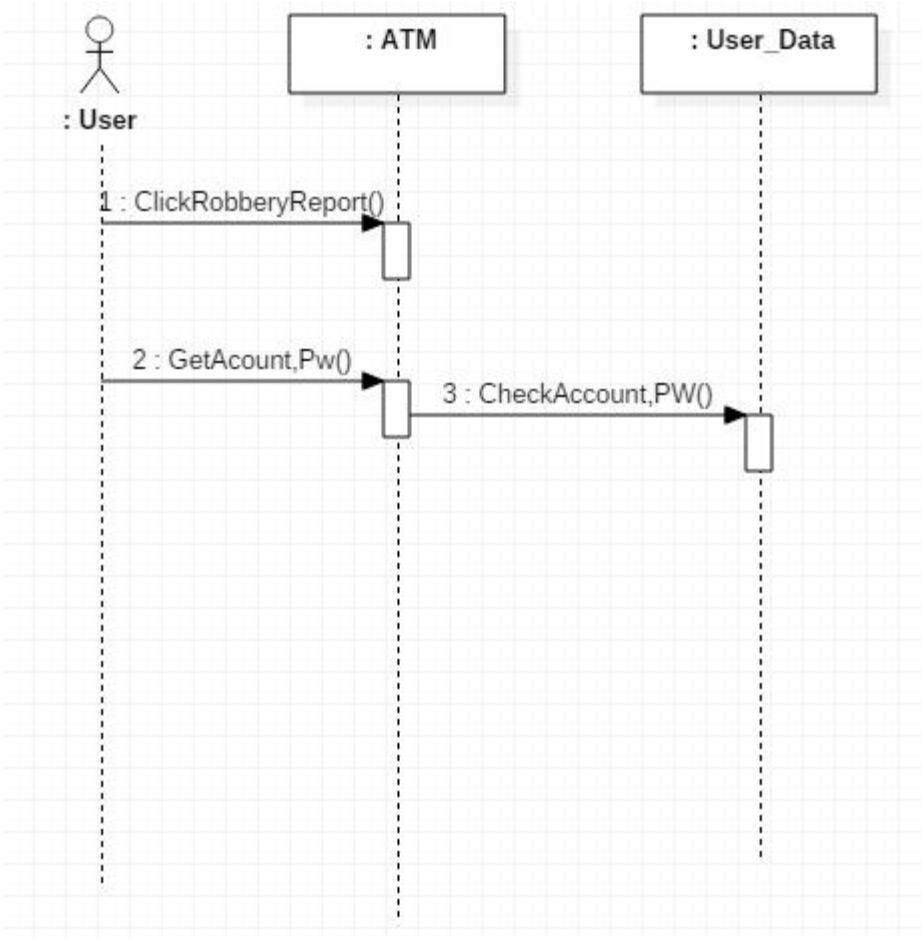
- Transfer


: ATM
: User_Data
: User
1 : transfer()
2 : GetAcount,Pw()
3 : CheckAccount,PW()
4 : GetAccount()
5 : CheckAccount
6 : GetAmount()
7 : Calculate()
8 : SaveInfo()

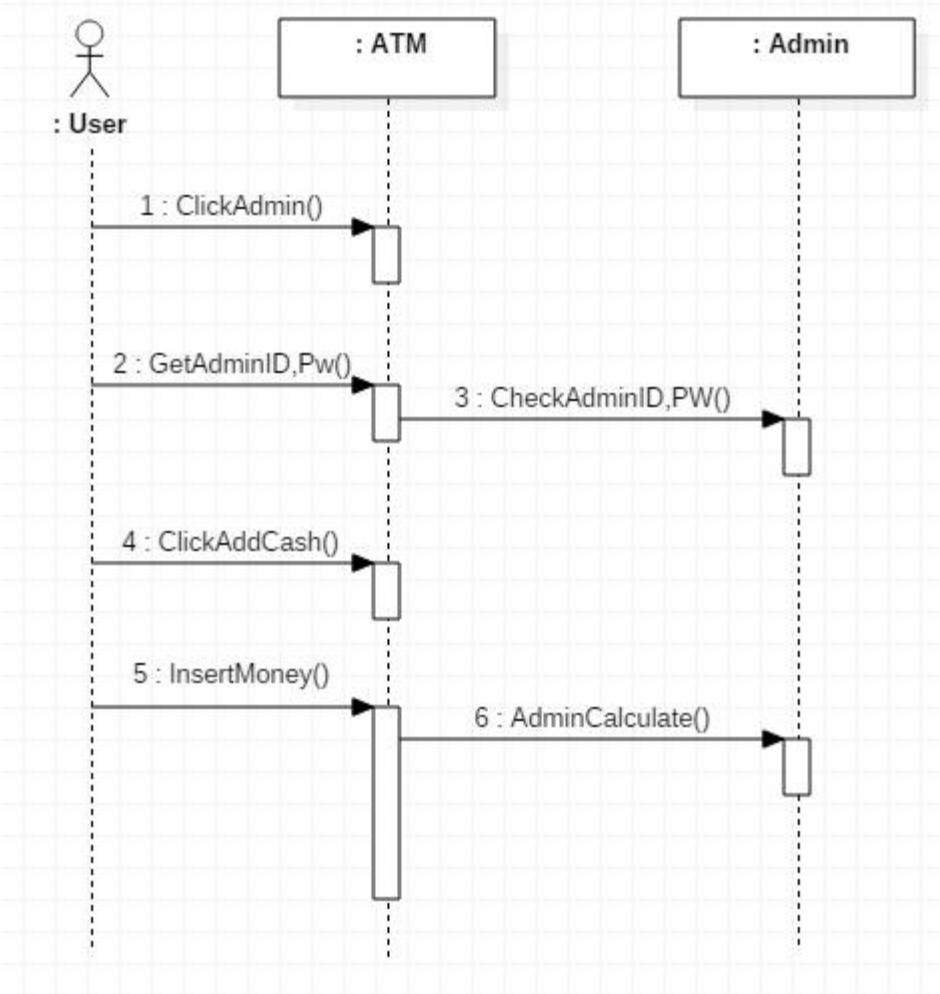
- CheckBalance



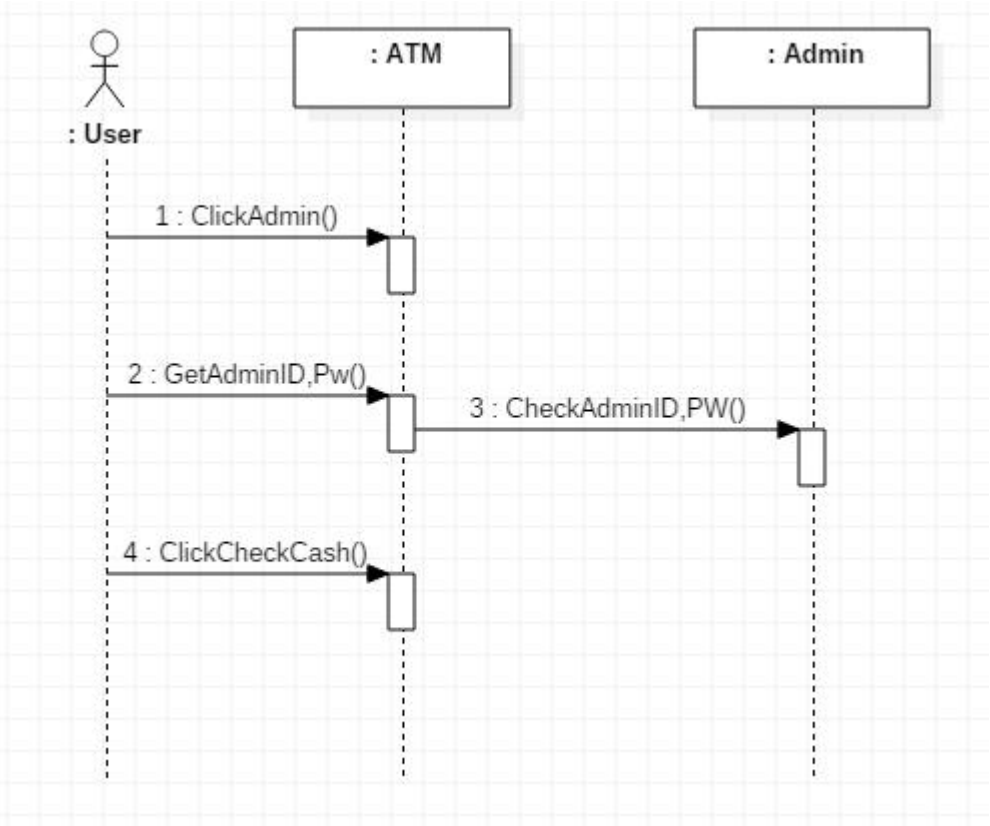
- RobberyReport



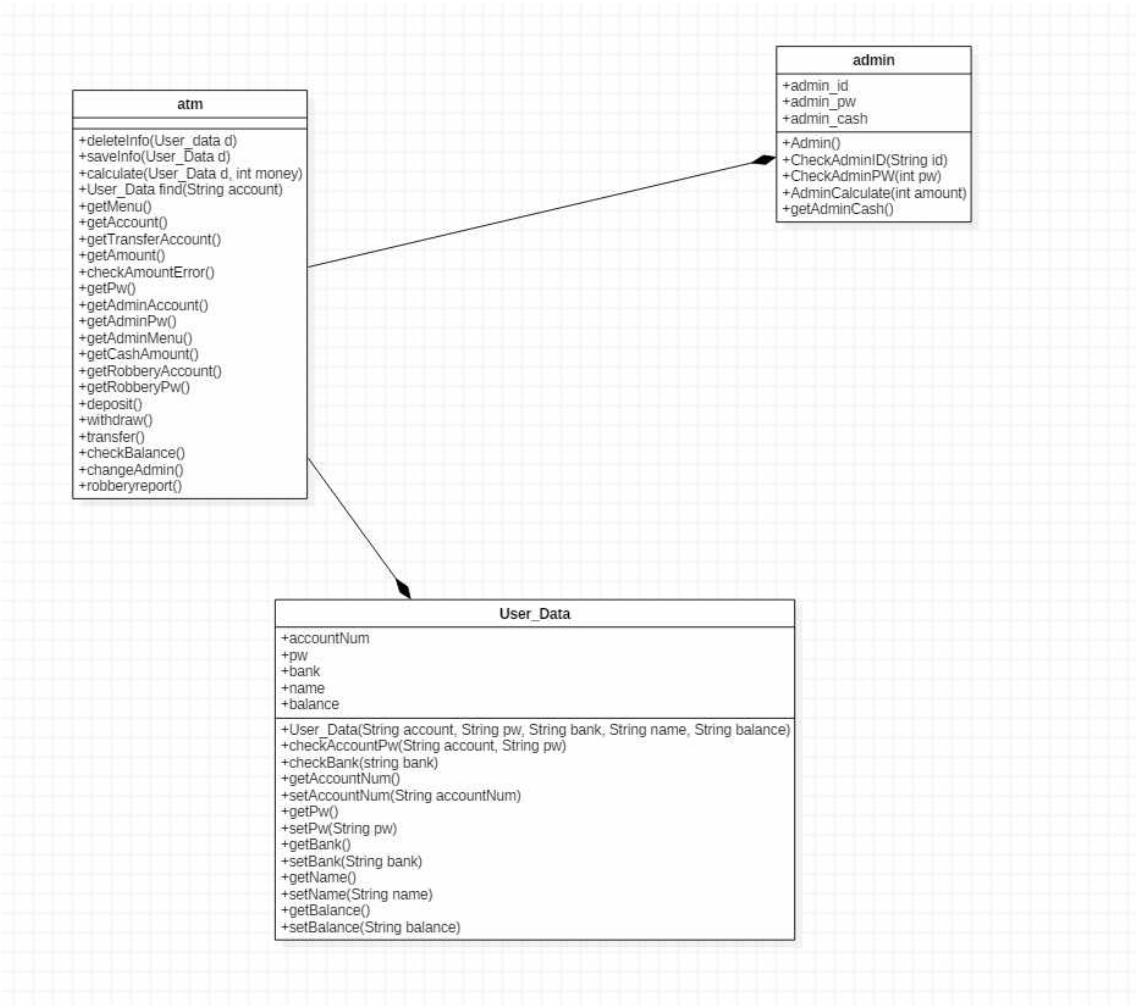
- AddCash


: ATM
: Admin
: User
1 : ClickAdmin()
2 : GetAdminID,Pw()
3 : CheckAdminID,PW()
4 : ClickAddCash()
5 : InsertMoney()
6 : AdminCalculate()

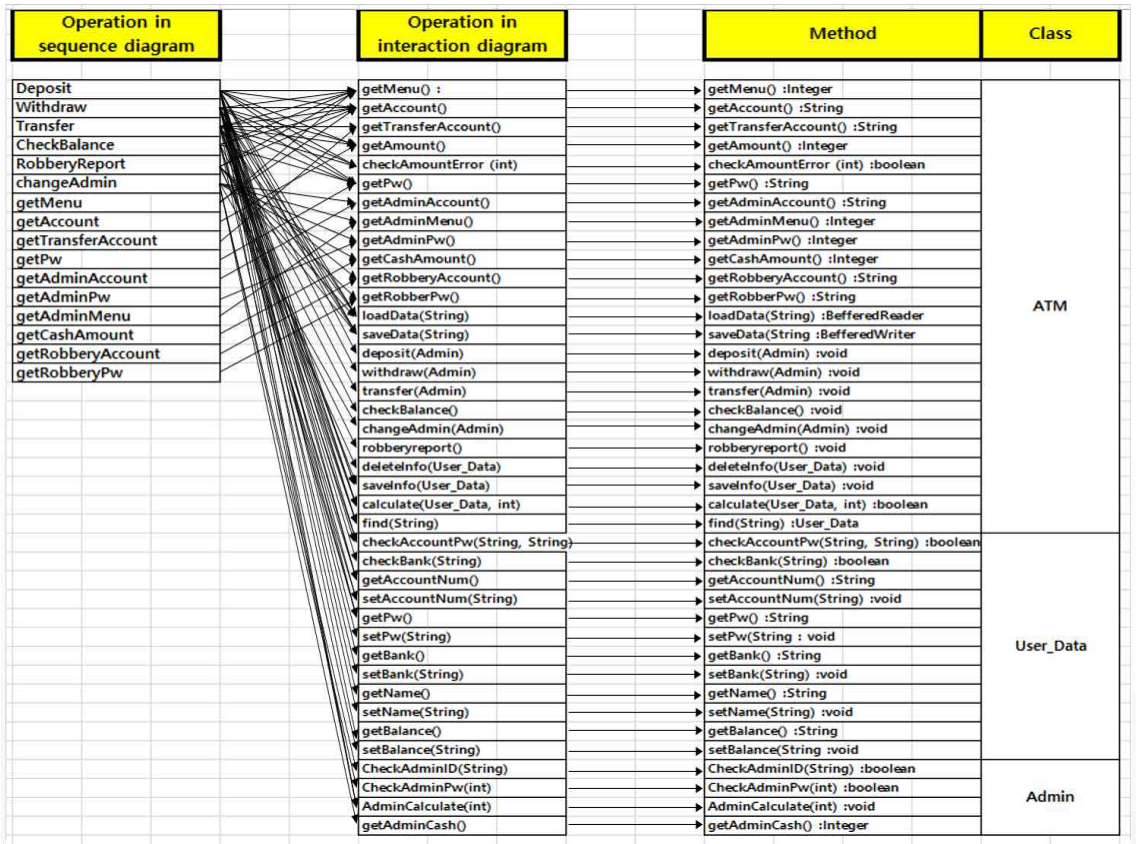
- CheckCash


: ATM
: Admin
: User
1 : ClickAdmin()
2 : GetAdminID,Pw()
3 : CheckAdminID,PW()
4 : ClickCheckCash()

5. Activity 2045. Define Design Class Diagrams



6. Activity 2046. Define Traceability Analysis



Operation in sequence diagram	Operation in interaction diagram	Method	Class
Deposit	getMenu() :	getMenu() :Integer	ATM
Withdraw	getAccount()	getAccount() :String	
Transfer	getTransferAccount()	getTransferAccount() :String	
CheckBalance	getAmount()	getAmount() :Integer	
RobberyReport	checkAmountError (int)	checkAmountError (int) :boolean	
changeAdmin	getPw()	getPw() :String	
getMenu	getAdminAccount()	getAdminAccount() :String	
getAccount	getAdminMenu()	getAdminMenu() :Integer	
getTransferAccount	getAdminPw()	getAdminPw() :Integer	
getPw	getCashAmount()	getCashAmount() :Integer	
getAdminAccount	getRobberyAccount()	getRobberyAccount() :String	
getAdminPw	getRobberPw()	getRobberPw() :String	
getAdminMenu	loadData(String)	loadData(String) :BefferedReader	
getCashAmount	saveData(String)	saveData(String :BefferedWriter	
getRobberyAccount	deposit(Admin)	deposit(Admin) :void	
getRobberyPw	withdraw(Admin)	withdraw(Admin) :void	
	transfer(Admin)	transfer(Admin) :void	
	checkBalance()	checkBalance() :void	
	changeAdmin(Admin)	changeAdmin(Admin) :void	
	robberyreport()	robberyreport() :void	
	deleteInfo(User_Data)	deleteInfo(User_Data) :void	
	saveInfo(User_Data)	saveInfo(User_Data) :void	
	calculate(User_Data, int)	calculate(User_Data, int) :boolean	
	find(String)	find(String) :User_Data	
	checkAccountPw(String, String)	checkAccountPw(String, String) :boolean	User_Data
	checkBank(String)	checkBank(String) :boolean	
	getAccountNum()	getAccountNum() :String	
	setAccountNum(String)	setAccountNum(String) :void	
	getPw()	getPw() :String	
	setPw(String)	setPw(String : void	
	getBank()	getBank() :String	
	setBank(String)	setBank(String) :void	
	getName()	getName() :String	
	setName(String)	setName(String) :void	
	getBalance()	getBalance() :String	
	setBalance(String)	setBalance(String :void	
	CheckAdminID(String)	CheckAdminID(String) :boolean	Admin
	CheckAdminPw(int)	CheckAdminPw(int) :boolean	
	AdminCalculate(int)	AdminCalculate(int) :void	
	getAdminCash()	getAdminCash() :Integer	