A New Reliable ATM

OOPT Phase 2030

Analyze

Project Team T6

201411140 권성완
201511247 김선정
201510436 허윤아
201510285 조수빈

Date

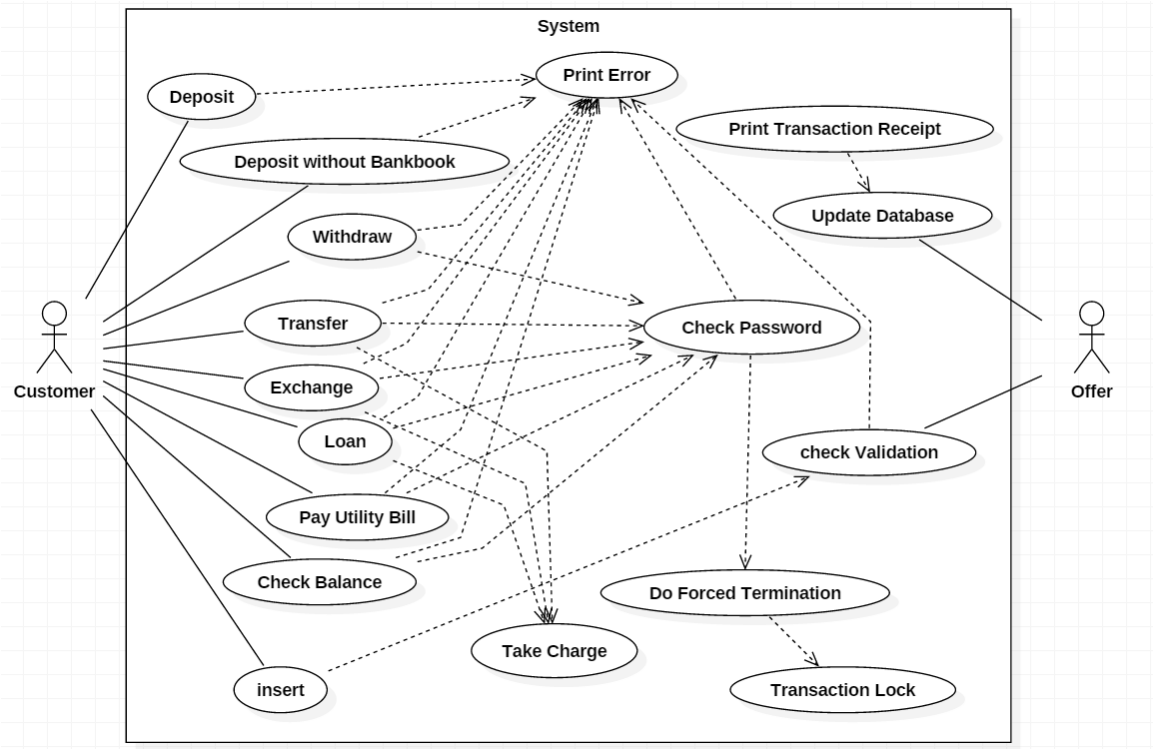
2018-06-01

Activity 2010. Revise Plan

1. OOPT 1000 version 3 중 Activity 1003의 functional requirements

Ref. #	Function
R1.1	Deposit
R1.2	Deposit Without bankbook
R1.3	Withdraw
R1.4	Transfer
R1.5	Exchange
R1.6	Loan
R1.7	Pay Utility Bill
R1.8	Check Balance
R2.1	Insert
R2.2	Print Transaction Receipt
R2.3	Print Error
R2.4	Do Forced Termination
R3.1	Take Charge
R4.1	Check Password
R5.1	Transaction Lock
R6.1	Check Validation
R6.2	Update Database

2. OOPT 1000 version 3 중 Activity 1006의 use case diagram(추가)



Activity 2020. Synchronize Artifacts

- OOPT Stage 1000이 version 4로 수정되면서 Functional requirement와 use case, glossary 가 수정되었다.

Activity 2031. Define Essential Use Cases

Use Case	1. Deposit
Actor	Customer
Purpose	Deposit cash into account or credit card
Overview	Customer deposits cash into account or credit card.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.1, R2.1, R2.2, R2.3, R2.4, R6.1, R6.2 Use case : Insert, Print Transaction Receipt, Print Error, Do Forced Termination, Check Validation, Update Database
Pre-Requisites	(N/A)
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) Customer selects deposit menu from the basic screen. 2. (S) Prints out "Input account or credit card number". 3. (A) Customer inputs credit card or account(Use case "Insert"). 4. (S) Invoke "Check Validation". If valid, ask customer to input cash. 5. (A) Customer Inputs cash in unit of 10000₩ and 50000₩. 6. (S) Check amount of cash inputted. 7. (S) If total amount of money is correct, invoke "Update Database". 8. (S) If DB is successfully updated, invoke "Print Transaction Receipt".
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 4 : If customer insert credit card, view loan record. Line 5 : If cash is not in unit of 10000₩ and 50000₩, print error. Line 7 : If incorrect, invoke "Print Error". If error occurs 3 times, invoke "Do Forced Termination". If loan record exists, loan is automatically repaid.

Use Case	2. Deposit without Bankbook
Actor	Customer
Purpose	Deposit cash into account without bankbook or check card
Overview	Customer deposits cash into default bank's account.

Type	Primary and Essential
Cross Reference	Functional Requirements : R1.2, R2.2, R2.3, R2.4, R6.1, R6.2 Use Case : Print Transaction Receipt, Print Error, Do Forced Termination, Check Validation, Update Database
Pre-Requisites	Customer should know exact account number to deposit.
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) Customer chooses Deposit Without Bankbook menu from basic screen. 2. (S) Ask Customer to input bank account number to deposit. 3. (A) Customer inputs bank account number. 4. (S) If bank account number is valid, ask customer to input cash in unit of 10000₩, 50000₩. 5. (S) Check total amount of cash. 6. (S) If counted right, invoke "Update Database". 7. (S) If DB is successfully updated, invoke "Print Transaction Receipt".
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 4 : if invalid, invoke "Print Error". If error occurs 3 times, invoke "Do Forced Termination"

Use Case	3. Withdraw
Actor	Customer
Purpose	Withdraw cash from bank account
Overview	Customer withdraws cash from bank account.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.3, R2.1, R2.2, R2.3, R2.4, R4.1, R5.1, R6.1, R6.2 Use Case : Insert, Print Transaction Receipt, Print Error, Do Forced Termination, Check Password, Transaction Lock, Check Validation, Update Database
Pre-Requisites	Customer should know password for the account, and balance should be enough to withdraw.
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) Customer selects withdraw menu from the basic screen. 2. (S) Prints out "Input account". 3. (A) Customer inputs account("Insert"). 4. (S) Invoke "Check Validation". If valid, ask customer to input amount of money to withdraw.

	5. (A) Customer inputs amount of money in unit of 10000₩, 50000₩ to withdraw from account. 6. (S) Ask for password for the account. 7. (A) Customer inputs password for the account. 8. (S) Invoke "Check Password". If password is correct, count numbers of bills. 9. (S) If balance is enough, withdraw cash and invoke "Update Database". 10. (S) If DB is successfully updated, invoke "Print Transaction Receipt".
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 4, 8, 9 : If incorrect or invalid, or if balance is not enough, invoke "Print Error". If error occurred more than three times, invoke "Do Forced Termination". Especially if password is incorrect 3 times, additionally invoke "Transaction Lock", "Update Database". Line 5 : If total amount of money to withdraw is over 50000₩, customer inputs number of 50000₩ bill. Line 9 : If customer's account does not belong to default bank, invoke "Take Charge".

Use Case	4. Transfer
Actor	Customer
Purpose	Transfer money from customer's account to another account
Overview	Customer transfers money from own account to another.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.4, R2.1, R2.2, R2.3, R2.4, R3.1, R4.1, R5.1, R6.1, R6.2 Use Case : Insert, Print Transaction Receipt, Print Error, Do Forced Termination, Take Charge, Check Password, Transaction Lock, Check Validation, Update Database
Pre-Requisites	Customer should know password for the account, and account to transfer money.
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) Customer selects transfer menu from the basic screen. 2. (S) prints out "Input account". 3. (A) Customer inputs account("Insert"). 4. (S) Invoke "Check Validation". If valid, ask customer to input

	bank and account number to transfer money. 5. (A) Customer inputs bank and account number to transfer money. 6. (S) If inputted information is valid, ask customer to input amount of money to transfer. 7. (A) Customer inputs amount of money to transfer. 8. (S) If balance is enough, ask for password for the account. 9. (A) Customer inputs password for the account. 10. (S) Invoke "Check Password". If password is correct, invoke "Update Database". 11. (S) If DB is successfully updated, invoke "Print Transaction Receipt".
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 4, 10 : If incorrect or invalid, or if balance is not enough, invoke "Print Error." If error occurs 3 times, invoke "Do Forced Termination". Especially, if password is incorrect 3 times, additionally invoke "Transaction Lock", "Update Database". Line 7 : If customer inputted different bank's account, or if customer's account does not belong to default bank, invoke "Take Charge".

Use Case	5. Exchange
Actor	Customer
Purpose	Exchange KRW into foreign currency
Overview	Customer exchanges KRW in account into foreign currency.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.5, R2.1, R2.2, R2.3, R2.4, R3.1, R4.1, R5.1, R6.1, R6.2 Use Case : Insert, Print Transaction Receipt, Print Error, Do Forced Termination, Take Charge, Check Password, Transaction Lock, Check Validation, Update Database
Pre-Requisites	Customer should know password for the account. ATM only handles USD, JPY, CNY, EUR.
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) Customer selects exchange menu from basic screen. 2. (S) Print out "Input account". 3. (A) Customer inputs account("Insert"). 4. (S) Invoke "Check Validation". If valid, print out list of countries

	available. 5. (A) Customer selects country to exchange money. 6. (S) Print out "input amount of money to exchange". 7. (A) Customer inputs amount of money to exchange. 8. (S) Calculate total amount of money based on exchange rate and ask for password. 9. (A) Customer inputs password. 10. (S) Invoke "Check Password". If password is correct and balance is enough, withdraw cash in foreign currency. 11. (S) If withdrawn correctly, invoke "Update Database". 12. (S) If DB is successfully updated, invoke "Print Transaction Receipt".
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 4, 9 : If incorrect or invalid, or if balance is not enough, invoke "Print Error". If error occurs 3 times, invoke "Do Forced Termination". Especially, if password is incorrect 3 times, additionally invoke "Transaction Lock", "Update Database". Line 10 : Invoke "Take Charge". Charge is deducted from balance.

Use Case	6. Loan
Actor	Customer
Purpose	Loan cash by credit card
Overview	Customer loans cash using credit card. There is loan limit.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.6, R2.1, R2.2, R2.3, R2.4, R3.1, R4.1, R5.1, R6.1, R6.2 Use Case : Loan, Insert, Print Transaction Receipt, Print Error, Do Forced Termination, Take Charge, Check Password, Transaction Lock, Check Validation, Update Database
Pre-Requisites	Customer should know password for the credit card.
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) Customer chooses loan menu from basic screen. 2. (S) Print out "Input credit card". 3. (A) Customer inputs credit card("Insert"). 4. (S) Invoke "Check Validation". If valid, print out "input amount of money to loan". 5. (A) Customer inputs amount of money to loan in unit of 10000₩, 50000₩.

	6. (S) print out "input password for the credit card". 7. (A) Customer inputs password for the credit card. 8. (S) Invoke "Check Password". If password is correct, count number of bills. 9. (S) If total amount of money to loan is under credit card limit, withdraw cash and invoke "Update Database". 10. (S) If DB is successfully updated, invoke "Print Transaction Receipt".
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 5 : If total amount of money to withdraw is over 50000₩, customer inputs number of 50000₩ bill. Line 4, 8 : If incorrect or invalid, invoke "Print Error". If error occurs 3 times, invoke "Do Forced Termination". Especially, if password is incorrect 3 times, additionally invoke "Transaction Lock", "Update Database". Line 9 : Invoke "Take Charge".

Use Case	7. Pay Utility Bill
Actor	Customer
Purpose	Pay utility bill by giro bill
Overview	Customer pay utility bill by giro bill and account.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.7, R2.1, R2.2, R2.3, R2.4, R4.1, R5.1, R6.1 R6.2 Use Case : Insert, Print Transaction Receipt, Print Error, Do Forced Termination, Check Password, Transaction Lock, Check Validation, Update Database
Pre-Requisites	Customer should know password for the account, and have giro bill.
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) Customer chooses pay utility bill menu from basic screen. 2. (S) Print out "input account written in giro bill". 3. (A) Customer inputs account number which belongs to Korean bank("Insert"). 4. (S) Invoke "Check Validation". If valid, print out "input account". 5. (A) Customer inputs account("Insert"). 6. (S) Invoke "Check Validation". If valid, print out "input password for the account".

	7. (A) Customer inputs password for the account. 8. (S) Invoke "Check Password". If password is correct and balance is enough to pay utility bill, invoke "Update Database". 9. (S) If DB is successfully updated, invoke "Print Transaction Receipt".
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 4, 6, 8 : If incorrect or invalid, or if balance is not enough, invoke "Print Error". If error occurs 3 times, invoke "Do Forced Termination". Especially, if password is incorrect 3 times, additionally invoke "Transaction Lock", "Update Database".

Use Case	8. Check balance
Actor	Customer
Purpose	Check balance of account
Overview	Customer checks balance of account.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.8, R2.1, R2.3, R2.4, R4.1, R5.1, R6.1 Use Case : Insert, Print Error, Do Forced Termination, Check Password, Check Validation
Pre-Requisites	Customer should know password for the account.
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) Customer chooses check balance menu from basic screen. 2. (S) Print out "Input account". 3. (A) Customer inputs account("Insert"). 4. (S) Invoke "Check Validation". If valid, print out "input password for the account". 5. (A) Customer inputs password for the account. 6. (S) Invoke "Check Password". If password is correct, print out recent transactional information(under 100) and balance after each transactional process. 7. (A) Customer input 'OK' button.(GUI level) 8. (S) Return to basic screen.
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 4, 6 : If incorrect or invalid, invoke "Print Error". If error occurs 3 times, invoke "Do Forced Termination". Especially, if password is incorrect 3 times, additionally invoke "Transaction Lock", "Update Database".

Use Case	9. Insert
Actor	Customer
Purpose	Insert method for transaction
Overview	Customer inserts method for transaction
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.1, R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.1, R2.3, R6.1 Use Case : Deposit, Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Check Balance, Print Error, Check Validation
Pre-Requisites	N/A
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) Customer inserts method for transaction, such as account or credit card.
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 1 : After, invoke "Check Validation".

Use Case	10. Print Transaction Receipt
Actor	(None)
Purpose	Check if transaction is successfully finished by printing out Transaction Receipt
Overview	System prints out transaction receipt to check if transaction is successfully finished.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.1, R1.2, R1.3, R1.4, R1.5, R1.6, R1.7, R2.2, R2.3, R6.2 Use Case : Deposit, Deposit Without Bankbook, Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Print Error, Update Database
Pre-Requisites	Transaction ended, DB updated
Typical Courses of Events	(A) : Actor, (S) : System 1. (S) Transaction ended successfully, and if DB is successfully updated, this use case is invoked.
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 1, 2 : If not successful, invoke "Print Error".

Use Case	11. Print Error
Actor	(None)

Purpose	Print error during transaction
Overview	System prints out various error messages during transaction.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.1, R1.2, R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.1, R2.2, R2.3, R2.4, R4.1, R5.1, R6.1, R6.2 Use Case : Deposit, Deposit Without Bankbook, Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Check Balance, Insert, Print Transaction Receipt, Do Forced Termination, Check Password, Transaction Lock, Check Validation, Update Database
Pre-Requisites	Error occurred during transaction
Typical Courses of Events	(A) : Actor, (S) : System 1. (S) If error occurred during certain transaction, this use case is invoked. 2. (S) If this use case occurs 3 times, invoke "Do Forced Termination".
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 2 : If password is incorrect 3 times, additionally invoke "Transaction Lock", "Update Database".

Use Case	12. Do Forced Termination
Actor	(None)
Purpose	Immediately end transaction when error occurs 3 times
Overview	System ends transaction automatically and immediately when error occurs 3 times.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.1. R1.2, R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.1, R2.3, R2.4, R5.1, R6.2 Use Case : Deposit, Deposit Without Bankbook, Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Check Balance, Insert, Print Error, Transaction Lock, Update Database
Pre-Requisites	Use case "Print Error" occurred 3 times
Typical Courses of Events	(A) : Actor, (S) : System 1. (S) This use case occurs if "Print Error" occurred 3 times.
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 1 : If password error occurs 3 times, additionally invoke "Transaction Lock", "Update Database".

Use Case	13. Take Charge
Actor	(None)
Purpose	Take charge during transaction
Overview	System takes charge when customer transfers, exchanges, loans.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.3, R1.4, R1.5, R1.6, R3.1 Use Case : Withdraw, Transfer, Exchange, Loan
Pre-Requisites	N/A
Typical Courses of Events	(A) : Actor, (S) : System 1. (S) During withdrawing, transferring, exchanging, and loaning, take charge.
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 1 : If credit rating is high, this use case is ignored.

Use Case	14. Check Password
Actor	(None)
Purpose	Check password of account or credit card
Overview	System checks password of account or credit card inputted by Customer.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.3, R2.4, R4.1, R5.1, R6.2 Use Case : Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Check Balance, Print Error, Do Forced Termination, Transaction Lock, Update Database
Pre-Requisites	Customer inputs password for the account or credit card.
Typical Courses of Events	(A) : Actor, (S) : System 1. (S) Compare password inputted between information stored in DB. 2. (S) If incorrect, invoke “Print Error”.
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 2 : If incorrect more than 3 times, invoke “Do Forced Termination”, “Transaction Lock”, “Update Database”,

Use Case	15. Transaction Lock
Actor	(None)
Purpose	Lock transaction of account or credit card when password error

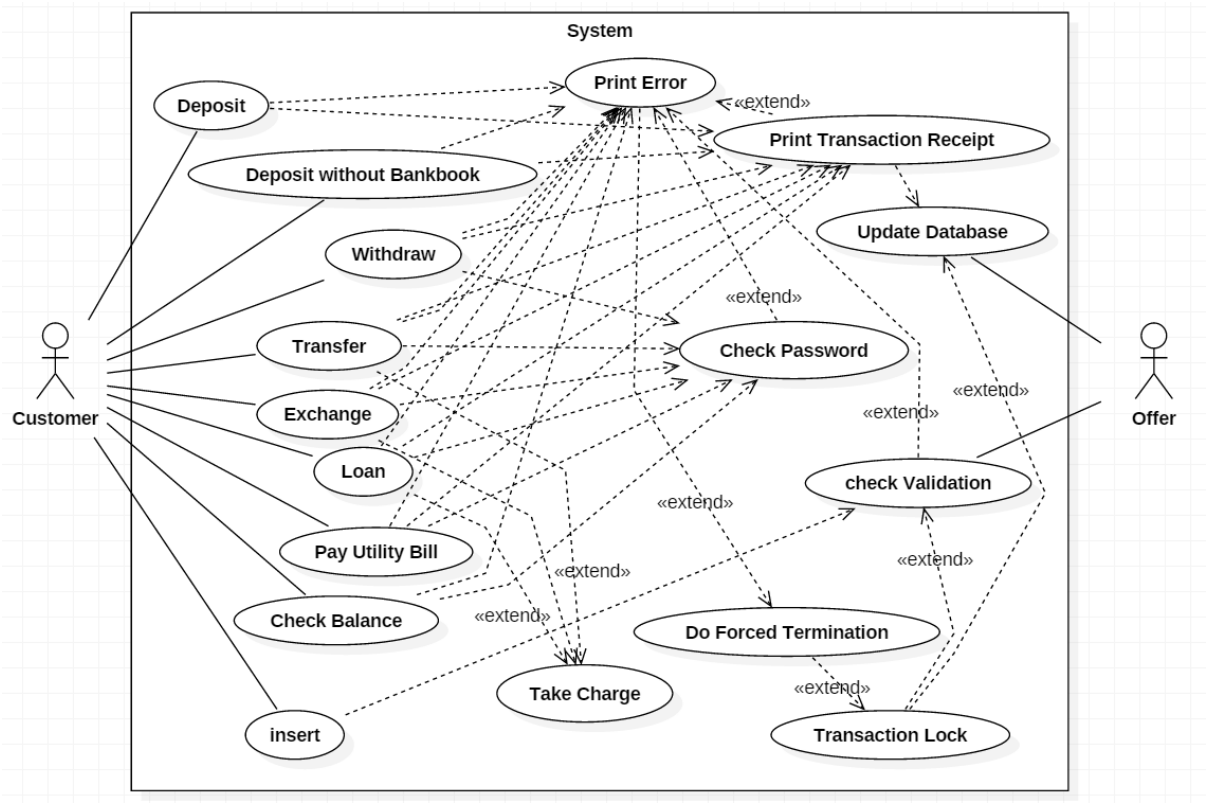
	occurs 3 times
Overview	System locks transaction of account or credit card when password error occurs 3 times.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.1, R1.2, R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.1, R2.3, R2.4, R4.1, R5.1, R6.2 Use Case : Deposit, Deposit Without Bankbook, Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Check Balance, Insert, Print Error, Do Forced Termination, Check Password, Update Database
Pre-Requisites	In use case "Check Password", use case "Print Error" occurred 3 times.
Typical Courses of Events	(A) : Actor, (S) : System 1. (S) This use case occurs if password error occurred 3 times. 2 (S) If this use case occurred, invoke "Update Database".
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 1 : In each error, invoke "Print Error". After error occurred 3 times, invoke "Do Forced Termination". Line 2 : Customer cannot unlock transaction by System, and cannot use account or credit card.

Use Case	16. Check Validation
Actor	Offer
Purpose	Check validation of inserted method
Overview	System checks validation of inserted method after selecting menu from basic screen.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.1, R1.2, R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.1, R2.3, R2.4, R5.1, R6.1 Use Case : Deposit, Deposit Without Bankbook, Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Check Balance, Insert, Print Error, Do Forced Termination, Transaction Lock
Pre-Requisites	Use case "Insert" invoked.
Typical Courses of Events	(A) : Actor, (S) : System 1. (S) Requests to check validation of inserted method to Offer. 2. (A) Offer compares information of inserted method between information stored in Database. 3. (A) If valid, Offer gives "valid" to System, and system goes on

	a process.
Alternative Courses of Events	N/A
Exceptional Courses of Events	Line 3 : If invalid or already has "Transaction Lock", invoke "Print Error". If error occurs 3 times, invoke "Do Forced Termination".

Use Case	17. Update Database
Actor	Offer
Purpose	Update Database after transaction
Overview	System requests Offer to update database after every transaction.
Type	Primary and Essential
Cross Reference	Functional Requirements : R1.1, R1.2, R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.2, R5.1, R6.2 Use Case : Deposit, Deposit Without Bankbook, Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Check Balance, Print Transaction Receipt, Transaction Lock
Pre-Requisites	Transaction ended successfully
Typical Courses of Events	(A) : Actor, (S) : System 1. (S) Request Offer to update database after every transaction. 2. (A) Offer updates database.
Alternative Courses of Events	Line 1 : If account already has "Transaction Lock", this use case is ignored.
Exceptional Courses of Events	N/A

Activity 2032. Refine Use Case Diagrams



Activity 2033. Define Domain Model

1. List concepts(domain class) from use-cases

- Guideline 1

Concept Category	Examples
Physical or tangible objects	Credit Card, Check Card, Bankbook ,Cash, Utility Bill, Exchange Rate, Balance, Receipt
Customer's Data	Name, Bank, Rating, Account number, Password, Dept(loan), Limit
Exchange Rate	USD, JPY, EUR, CNY, KRW
Basic function	Deposit, Withdraw, Deposit Without Bankbook, Transfer, Exchange, Loan, Check balance, Pay utility bill
Dealing Error	Print Error, Do Forced Termination, Transaction Lock
Dealing Validation & Update	Check Password, Check Credit, Insert, Update Server Information
Account State	Lock, Normal
Actor	Customer, Offer
Need to take charge	Transfer, Exchange, Loan

- Guideline 2 ; Using Noun Phrases

Deposit	Withdraw	Deposit Without Bankbook	Transfer	Loan
Transaction Receipt	Charge	Utility Bill	Balance	Exchange
Cash	Credit Card	Check Card	Bank	Customer
Offer	Customer Information	Validation	Limit	Password
Account Number	Giro	receiver(transfer)	sender(transfer)	Error
Charge	ATM	Lock	Update Server	Payment

2. Assign class name into a concept

Offer	Customer	Basic function	Error
Update Server	Account	Validation	

3. Draw a conceptual class diagram

Offer	Customer	Basic function	Error
Update Server	Account	Validation	

4. Identify associations according to association categories

A selects B	Customer – Basic Function
A requests to B	Update Server – Offer
A refers to B	Basic function – Error Basic function – Validation Error – Validation
A occurs after B	Update server - Basic function Update Server – Error
A handles B	Offer – Account
A has B	Customer – Account

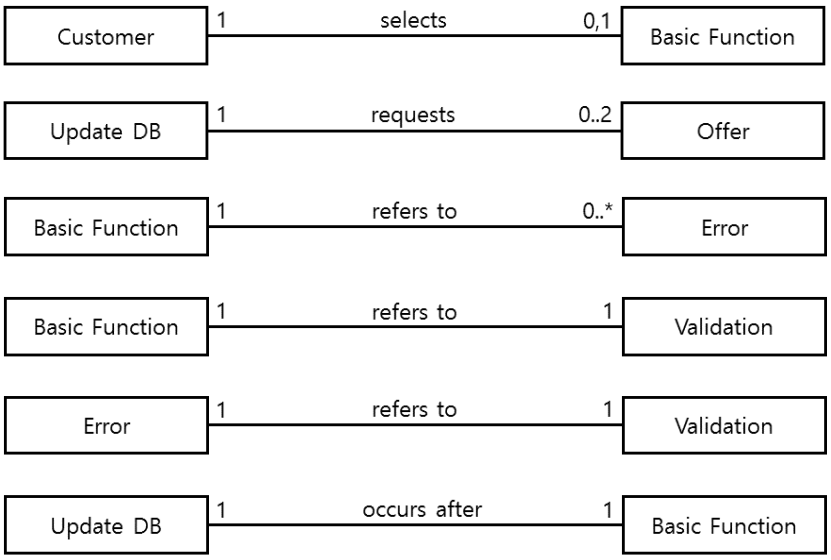
5. Assign priority into associations

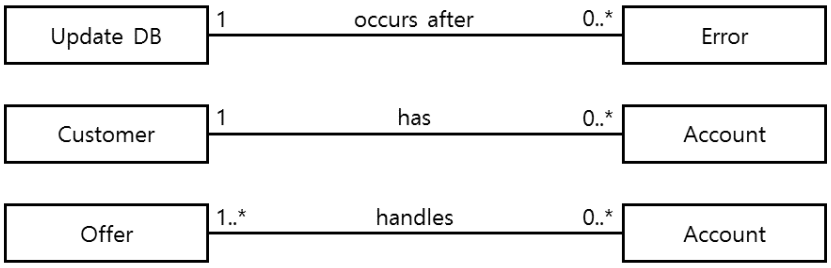
Customer – Basic Function	High
Basic Function – Error	High
Basic Function – Validation	High

6. Assign names into associations

- Customer *selects* Basic Function
- Update Server *requests* to Offer
- Basic function *refers to* Error
- Basic function *refers to* Validation
- Error *refers to* Validation
- Update server *occurs after* Basic function
- Update Server *occurs after* Error
- Offer *handles* Account

7. Add roles and Multiplicity





8. Add Attributes

Customer
Name: String Account: Account Rating: Enum Dept: int Limit: int credit card num: int check card num: int

Basic function
giro: int payments: enum Exchange Rate: int

Update server
Bank: enum Current Account: Account

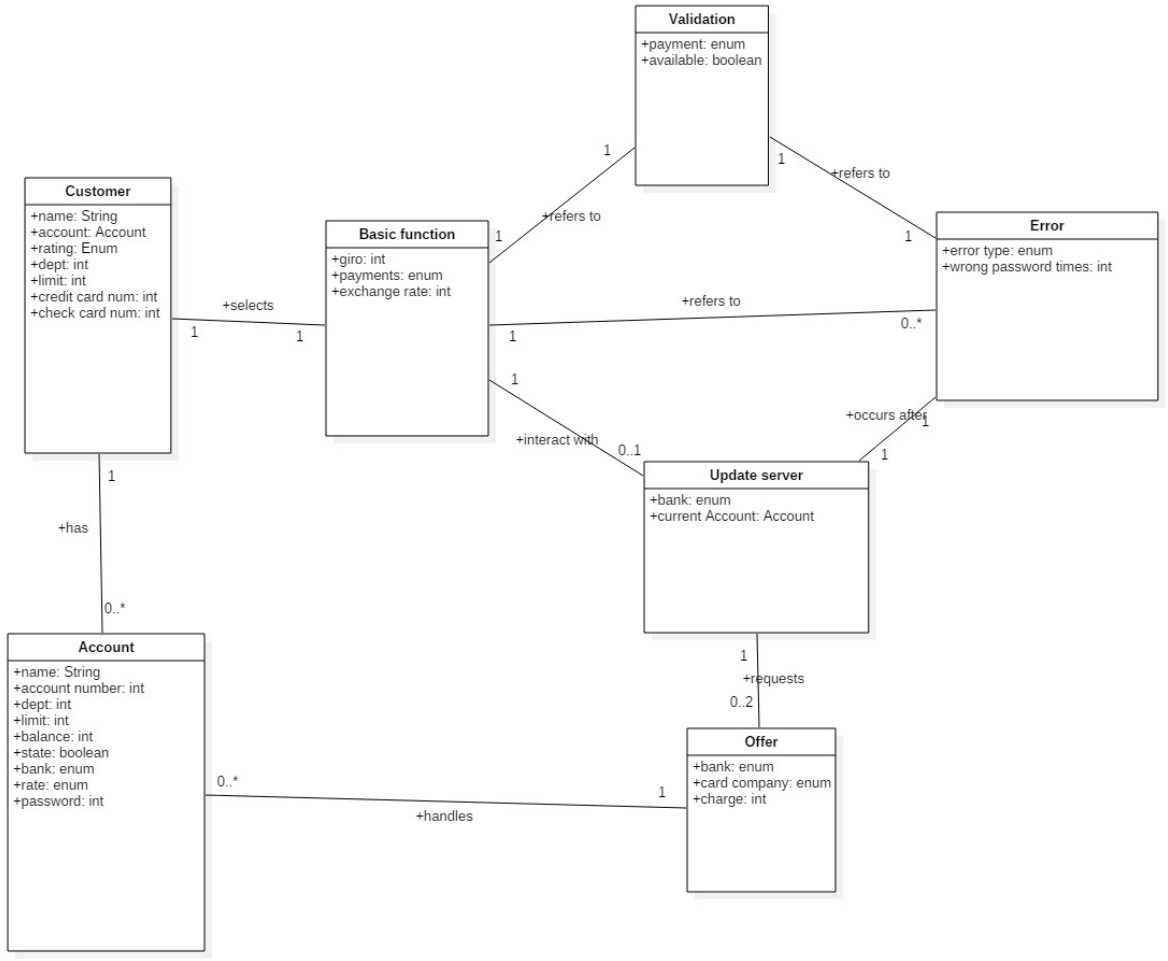
Validation
payment: enum available: boolean

Offer
Bank: enum Card company: enum charge: int

Error
Error type: enum

Account
name: String Account number: int dept: int limit: int balance: int state: boolean bank: enum rate: enum password: int

9. Define Domain Model

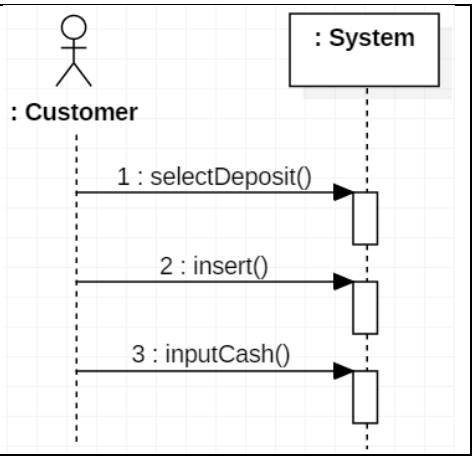


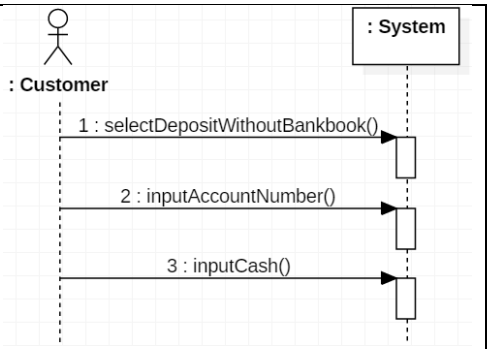
Activity 2034. Refine Glossary

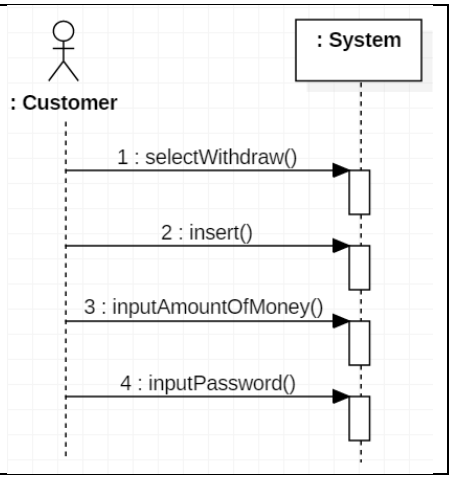
Term	Category	Remarks
Customer	Class	The person who has accounts
Basic function	Class	functions that customer can uses(select)
Account	Class	Account information
Offer	Class	Deal with all customer's accounts and credit card information
Update server	Class	ATM requests Offer to update customer's information which has changed by basic function
Error	Class	Deal with errors
Validation	Class	Deal with validations
account num	Attribute	account number(bankbook number)
check card num	Attribute	check card number

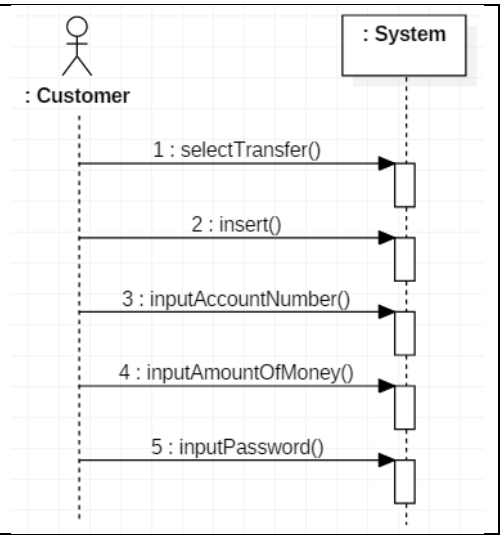
credit card num	Attribute	credit card number
rating	Attribute	credit rating(VIP, Gold, Silver)
Dept	Attribute	amount of dept(occurs by loan)
Limit	Attribute	loan limit
giro	Attribute	has virtual account number
payments	Attribute	credit card/ check card/ cash
exchange rate	Attribute	USD, JPY, EUR, CNY, KRW
bank	Attribute	bank company name
card company	Attribute	card company name
charge	Attribute	commission
error type	Attribute	password inconsistency, transaction lock, over limit, not enough balance, not available transfer receiver…
available	Attribute	check account is available
name	Attribute	customer or account owner's name
balance	Attribute	amount of money that account has
password	Attribute	password
State	Attribute	Account lock/ normal

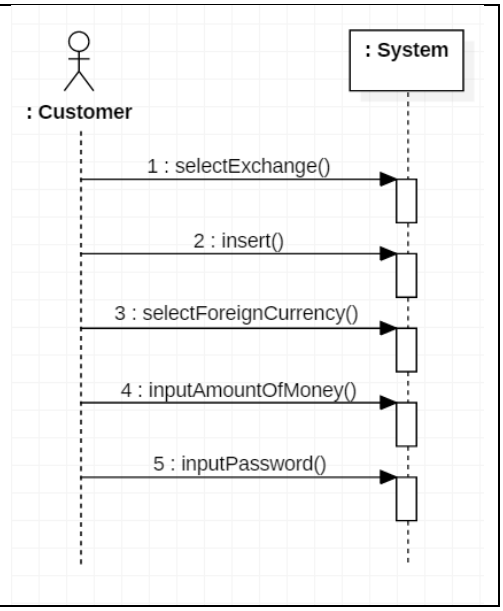
Activity 2035. Define System Sequence Diagrams

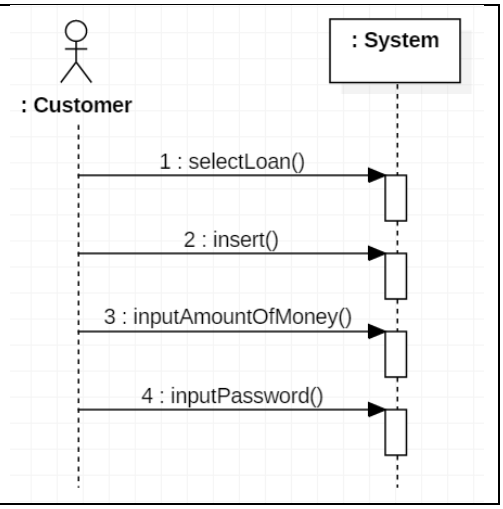
Use Case : Deposit	
1. Customer selects deposit menu from the basic screen. 2. Customer inputs account or credit card number. 3. Customer Inputs cash in unit of 10000₩, 50000₩. 4. System prints out transaction receipt.	

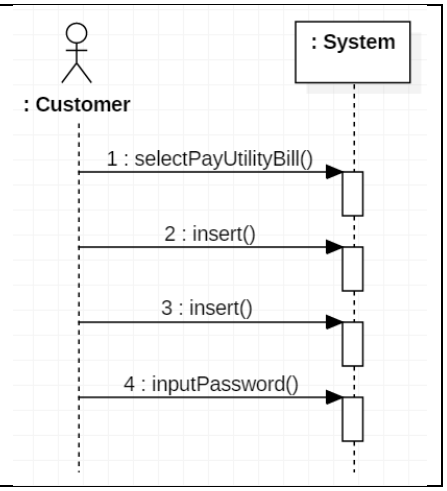
Use Case : Deposit without Bankbook	
1. Customer selects Deposit Without Bankbook menu from screen. 2. Customer inputs bank account number. 4. Customer inputs cash in unit of 10000₩, 50000₩. 4. system prints out transaction receipt.	

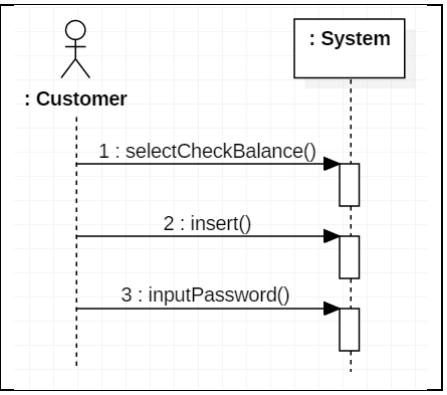
Use Case : Withdraw	
1. Customer selects withdraw menu from the basic screen. 2. Customer inputs account. 3. Customer inputs amount of money in unit of 10000₩, 50000₩ to withdraw from account. 4. Customer inputs password for the account. 5. If balance is enough, System withdraws cash and prints out transaction receipt.	

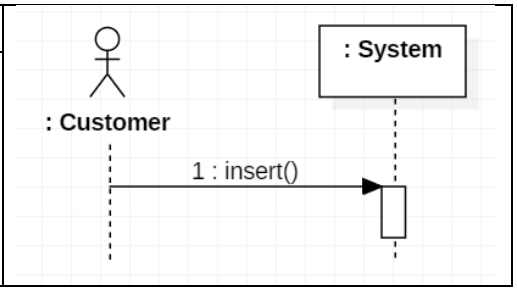
Use Case : Transfer	
1. Customer selects transfer menu from the basic screen. 2. Customer inputs account. 3. Customer inputs account number to transfer money. 4. Customer inputs amount of money to transfer. 5. Customer inputs password for the account. 6. If password is correct, system prints out transaction receipt.	

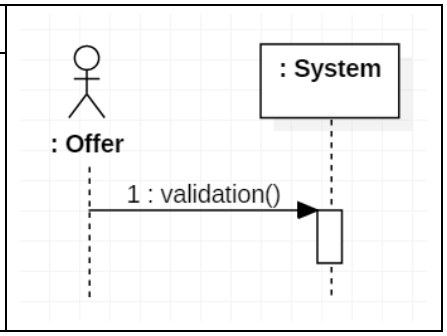
Use Case : Exchange	
1. Customer selects exchange menu from basic screen. 2. Customer inputs account. 3. Customer selects foreign currency to exchange. 4. Customer inputs amount of money to exchange. 5. Customer inputs password. 6. If password is correct, and If withdrawn correctly, system prints out transaction receipt.	

Use Case : Loan	
1. Customer chooses loan menu from basic screen. 2. Customer inputs account. 3. Customer inputs amount of money to loan in unit of 10000₩, 50000₩. 4. Customer inputs password for the credit card. 5. If password is correct, money to loan is under loan limit, withdrawn correctly, system prints out transaction receipt.	

Use Case : Pay Utility Bill	
1. Customer chooses pay utility bill menu from basic screen. 2. Customer inputs account written in giro bill. 3. Customer inputs own account. 4. Customer inputs password for the account. 5. If password is correct, balance is enough, system prints out transaction receipt.	

Use Case : Check Balance	
1. Customer chooses check balance menu from basic screen. 2. Customer inputs account. 3. Customer inputs password for the account. 4. If password is correct, system prints out recent transactional information and balance after each transactional process.	

Use Case : Insert	
1. Customer inserts method for transaction, such as account of credit card.	

Use Case : Check Validation	
1. System requests to check validation of inserted method to Offer. 2. Offer compares information of inserted method between information stored in Database. 3. If inserted method is valid, Offer gives "valid" to System, and system goes on a process.	

Activity 2036. Define Operation Contracts

- System Operations

Use Case	System Operations
1. Deposit	1. selectDeposit 2. insert 3. inputCash
2. Deposit Without Bankbook	1. selectDepositWithoutBankbook 2. inputAccountNumber 3. inputCash
3. Withdraw	1. selectWithdraw 2. insert 3. inputAmountOfMoney 4. inputPassword
4. Transfer	1. selectTransfer 2. insert 3. inputAccountNumber 4. inputAmountOfMoney 5. inputPassword
5.Exchange	1.selectExchange 2. insert 3. selectForeignCurrency 4. inputAmountOfMoney 5. inputPassword
6. Loan	1. selectLoan 2. insert 3. inputAmountOfMoney 4. inputPassword
7. Pay Utility Bill	1. selectPayUtilityBill 2. insert 3. insert 4. inputPassword
8. Check Balance	1. selectCheckBalance 2. insert 3. inputPassword
9. Insert	1. insert
10. Check Validation	1. validation

- Operation Contracts

Name	selectDeposit
Responsibilities	Customer가 입금 기능을 선택한다
Type	System
Cross References	Functional Requirements : R1.1, R2.1, R2.2, R2.3, R2.4, R6.1, R6.2 Use Cases : Deposit, Insert, Print Transaction Receipt , Print Error, Do Forced Termination, Check Validation, Update Database
Exceptions	N/A
Output	입금 과정을 진행한다.
Pre-conditions	초기 화면
Post-conditions	매체(신용카드, 계좌)를 입력하는 창을 띄운다.

Name	selectDepositWithoutBankbook
Responsibilities	무통장입금 기능을 선택한다
Type	System
Cross References	Functional Requirements : R1.2, R2.2, R2.3, R2.4, R6.2 Use Cases : Deposit Without Bankbook, Print Transaction Receipt, Print Error, Do Forced Termination, Update Database
Exceptions	Default로 설정된 한 가지 은행에 대해서만 처리할 수 있다.
Output	무통장입금 과정을 진행한다.
Pre-conditions	초기 화면
Post-conditions	입금할 계좌를 입력하는 창을 띄운다.

Name	selectWithdraw
Responsibilities	출금 기능을 선택한다
Type	System
Cross References	Functional Requirements : R1.3, R2.1, R2.2, R2.3, R2.4, R4.1, R5.1, R6.1, R6.2 Use Cases : Withdraw, Insert, Print Error, Print Transaction Receipt, Check Password, Transaction Lock, Check Validation, Update Database
Exceptions	N/A
Output	출금 과정을 진행한다.
Pre-conditions	초기 화면
Post-conditions	매체(계좌)를 입력하는 창을 띄운다.

Name	selectTransfer
Responsibilities	송금 기능을 선택한다

Type	System
Cross References	Functional Requirements : R1.4, R2.1, R2.2, 2.3, R2.4, R3.1, R4.1, R5.1, R6.1, R6.2 Use Cases : Transfer, Insert, Take Charge, Print Transaction Receipt, Print Error, Do Forced Termination, Take Charge, Check Password, Transaction Lock, Check Validation, Update Database
Exceptions	N/A
Output	송금 과정을 진행한다.
Pre-conditions	초기 화면
Post-conditions	매체(계좌)를 입력하는 창을 띄운다.

Name	selectExchange
Responsibilities	환전 기능을 선택한다
Type	System
Cross References	Functional Requirements : R1.5, R2.1, R2.2, R2.3, R2.4, R3.1, R4.1, R5.1, R61, R6.2 Use Cases : Exchange, Insert, Print Transaction Receipt, Print Error, Do Forced Termination, Take Charge, Check Password, Transaction Lock, Check Validation, Update Database
Exceptions	N/A
Output	환전 과정을 진행한다.
Pre-conditions	초기 화면
Post-conditions	매체(계좌)를 입력하는 창을 띄운다.

Name	selectLoan
Responsibilities	대출 기능을 선택한다
Type	System
Cross References	Functional Requirements : R1.6, R2.1, R2.2, R2.3, R2.4, R3.1, R4.1, R5.1, R6.1, R6.2 Use Cases : Loan, Insert, Print Transaction Receipt, Print Error, Do Forced Termination, Take Charge, Check Password, Transaction Lock, Check Validation, Update Database
Exceptions	N/A
Output	대출 과정을 진행한다.
Pre-conditions	초기 화면
Post-conditions	매체(신용카드)를 입력하는 창을 띄운다.

Name	selectPayUtilityBill
Responsibilities	공과금 납부 기능을 선택한다
Type	System
Cross References	Functional Requirements : R1.7, R2.1, R2.2, R2.3, R2.4, R4.1, R5.1, R6.1, R6.2 Use Cases : Pay Utility Bill, Insert, Print Transaction Receipt, Print Error, Do Forced Termination, Check Password, Transaction Lock, Check Validation, Update Database
Exceptions	N/A
Output	공과금 납부 과정을 진행한다.
Pre-conditions	초기 화면
Post-conditions	매체(계좌)를 입력하는 창을 띄운다.

Name	selectCheckBalance
Responsibilities	거래 조회 기능을 선택한다
Type	System
Cross References	Functional Requirements : R1.8, R2.1, R2.3, R2.4, R4.1, R5.1, R6.1 Use Cases : Check Balance, Insert, Print Error, Do Forced Termination, Check Password, Transaction Lock, Check Validation
Exceptions	N/A
Output	거래 조회 과정을 진행한다.
Pre-conditions	초기 화면
Post-conditions	매체(신용카드, 계좌)를 입력하는 창을 띄운다.

Name	selectForeignCurrency
Responsibilities	환전 과정에서 외화의 종류를 선택한다
Type	System
Cross References	Functional Requirements : R1.5, R3.1 Use Cases : Exchange, Take Charge
Exceptions	선택 가능한 외화의 종류는 USD, JPY, CNY, EUR 총 네 종류이다.
Output	환전 과정을 진행한다.
Pre-conditions	환전 과정에서 매체(계좌)를 입력한다.
Post-conditions	환전할 금액을 입력하는 창을 띄운다.

Name	insert
Responsibilities	거래에 필요한 매체(신용카드, 계좌, 지로용지)를 입력한다
Type	System
Cross References	Functional Requirements : R1.1, R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.3, R6.1

	Use Cases : Deposit, Withdraw, Transfer, Exchange, Pay Utility Bill, Check Balance, Print Error, Check Validation
Exceptions	ATM에서 처리 가능한 매체(이미 존재하는, 잠겨 있지 않은 계좌 혹은 신용카드)를 입력해야 거래가 진행된다.
Output	입력한 매체의 정보를 받아, Database에 저장된 정보와 비교하도록 한다.
Pre-conditions	Customer가 무통장입금을 제외한 각 기능(메뉴)을 선택한다.
Post-conditions	System은 Customer가 입력한 매체(신용카드, 계좌, 지로용지)의 정보를 받아서, Database에 저장되어 있던 정보와 비교하도록 한다.

Name	inputCash
Responsibilities	입금, 무통장입금 과정에서 현금을 투입한다.
Type	System
Cross References	Functional Requirements : R1.1, R1.2, R2.3, R5.1, R6.2 Use Cases : Deposit, Deposit Without Bankbook, Print Error, Transaction Lock, Update Database
Exceptions	현금은 만원 단위로만 입력 되어야 한다. 만원 단위가 아닐 경우 Error message를 출력한다.
Output	입력한 현금을 확인하고, Database를 업데이트한다.
Pre-conditions	입금 : 입금 기능 선택 이후, 계좌를 입력한다. 무통장입금 : 무통장입금 기능 선택 이후, 입금할 계좌를 입력한다.
Post-conditions	입력된 현금을 확인하고, Database에 정보를 업데이트한다.

Name	inputAccountNumber
Responsibilities	무통장입금, 송금 과정에서 입금 혹은 송금 받을 계좌번호를 입력한다.
Type	System
Cross References	Functional Requirements : R1.2, R1.4, R2.3, R5.1, R6.1 Use Cases : Deposit Without Bankbook, Transfer, Print Error, Transaction Lock, Check Validation
Exceptions	존재하지 않거나 거래가 잠겨 있는 계좌를 입력한 경우 Error message를 출력한다.
Output	N/A
Pre-conditions	무통장입금 : 무통장입금 기능을 선택한다. 송금 : 송금 기능 선택 이후, 송금할 계좌를 입력 후 송금 받을 계좌의 은행을 고른다.
Post-conditions	무통장입금 : 현금을 입금하는 창을 띄운다. 송금 : 송금할 금액을 입력하는 창을 띄운다.

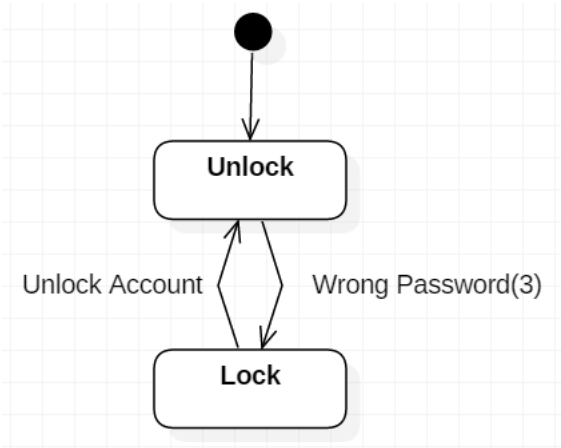
Name	inputAmountOfMoney
Responsibilities	출금, 송금, 환전, 대출 과정에서 거래할 금액을 입력한다
Type	System
Cross Reference	Functional Requirements : R1.3, R1.4, R1.5, R1.6, R2.3, R2.4, R3.1, R6.2 Use Cases : Withdraw, Transfer, Exchange, Loan, Print Error, Do Forced Termination, Take Charge, Update Database
Exception	거래 가능 한도까지만 입력할 수 있다.
Output	입력한 금액을 확인하고, 비밀번호를 입력하는 창을 띄운다.
Pre-Conditions	출금 : 출금 기능을 선택하고, 계좌를 입력한다. 송금 : 송금 받을 계좌번호를 입력한다. 환전 : 계좌를 입력한 후, 환전할 외화의 종류를 선택한다. 대출 : 대출 기능을 선택하고, 신용카드를 입력한다.
Post-Conditions	입력한 금액을 확인하고, 계좌 혹은 신용카드의 비밀번호를 입력하는 창을 띄운다.

Name	inputPassword
Responsibilities	출금, 송금, 환전, 대출, 공과금 납부, 잔액 확인 과정에서 비밀번호를 입력한다
Type	System
Cross Reference	Functional Requirements : R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.3, R2.4, R4.1, R5.1, R6.1 Use Cases : Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Check Balance, Print Error, Do Forced Termination, Check Password, Transaction Lock, Update Database
Exception	비밀번호를 잘못 입력한 경우 Error message를 출력한다. 비밀번호가 3번 잘못 입력될 경우 계좌 및 카드는 거래가 불가능하다.
Output	올바른 비밀번호를 입력한 경우 거래를 완료하고, Database를 업데이트한다.
Pre-Conditions	출금 : 계좌번호 입력 후, 인출할 금액을 입력한다. 송금 : 송금 받을 계좌를 입력한 후, 송금할 금액을 입력한다. 환전 : 외화의 종류를 선택하고, 환전할 금액을 입력한다. 대출 : 신용카드 입력 후, 대출할 금액을 입력한다. 공과금 납부 : 지로 고지서 투입 이후, 계좌번호를 입력한다. 잔액 확인 : 잔액 확인 기능 선택 후, 계좌번호를 입력한다.
Post-Conditions	올바른 비밀번호를 입력했을 경우, 거래를 완료하고 Database를 업데이트한다.

Name	Validation
Responsibilities	System을 통해 받은 정보의 유효성을 확인해 값을 반환한다.
Type	System
Cross Reference	Functional Requirements : R1.1, R1.2, R1.3, R1.4, R1.5, R1.6, R1.7, R1.8, R2.1, R2.3, R4.1, R5.1, R6.1 Use Cases : Deposit, Deposit Without Bankbook, Withdraw, Transfer, Exchange, Loan, Pay Utility Bill, Check Balance, Insert, Print Error, Check Password, Transaction Lock, Check Validation
Exception	유효하지 않은 경우 Error message를 출력한다.
Output	Customer에게 입력한 매체 혹은 계좌번호가 유효한지, 아닌지 알린다.
Pre-Conditions	Customer가 거래 과정에서 매체 혹은 계좌번호를 입력한다.
Post-Conditions	유효한 경우 거래를 계속 진행한다.

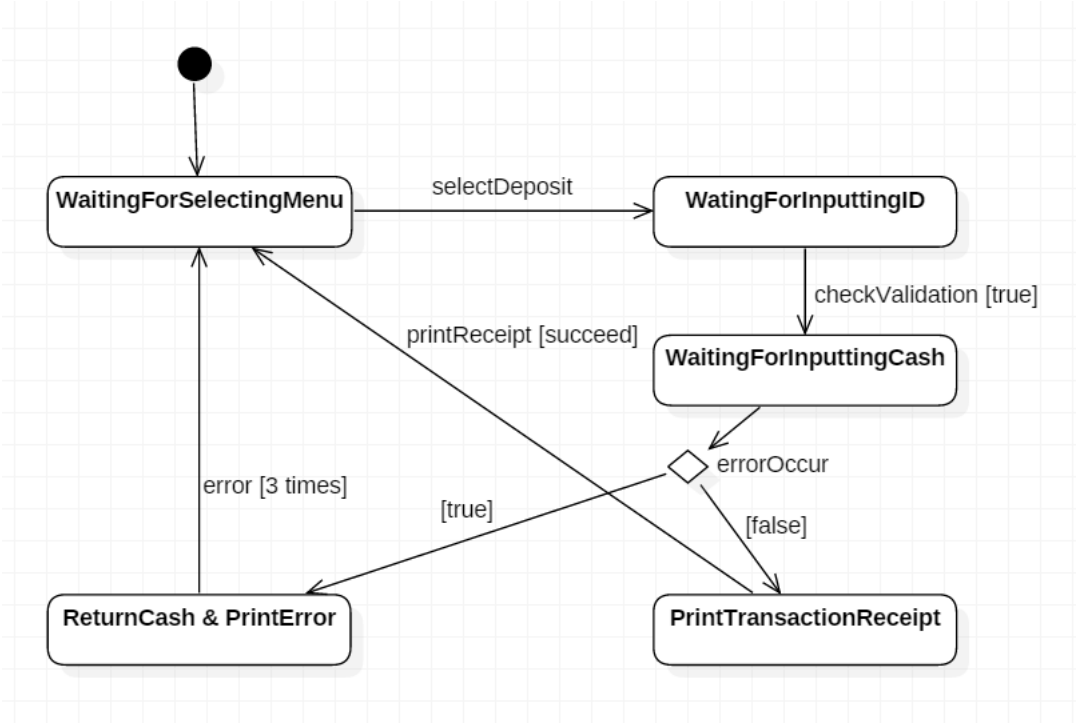
Activity 2037. Define State Diagrams

<Class State Diagram for "Account">

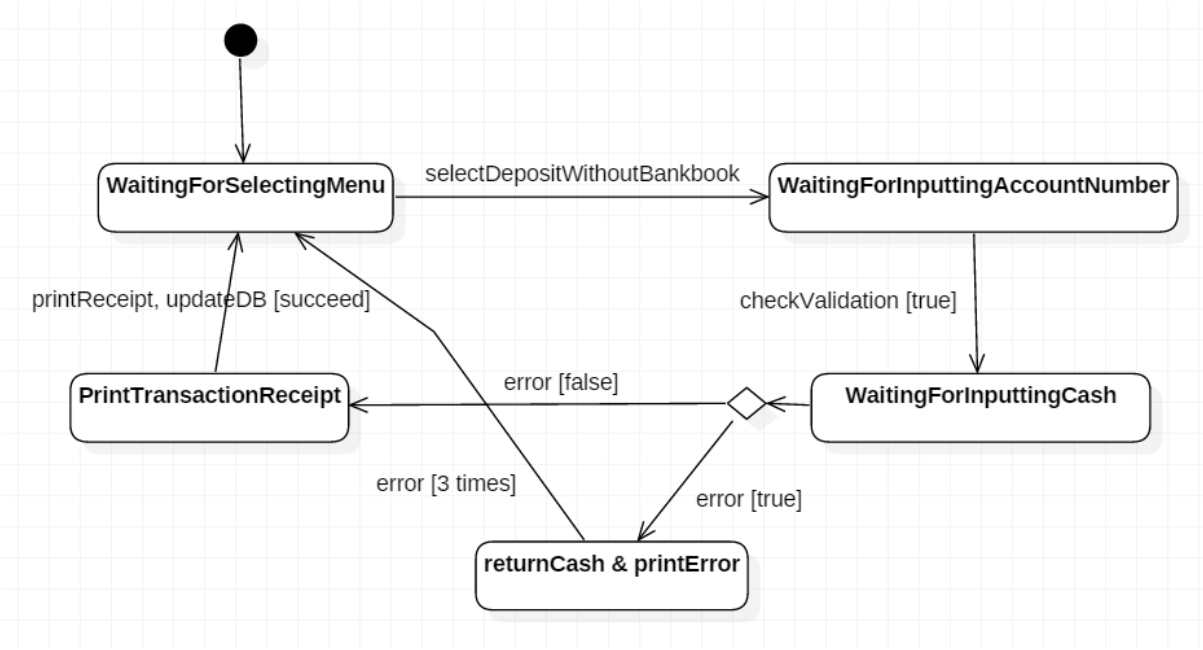


<Use Case State Diagram>

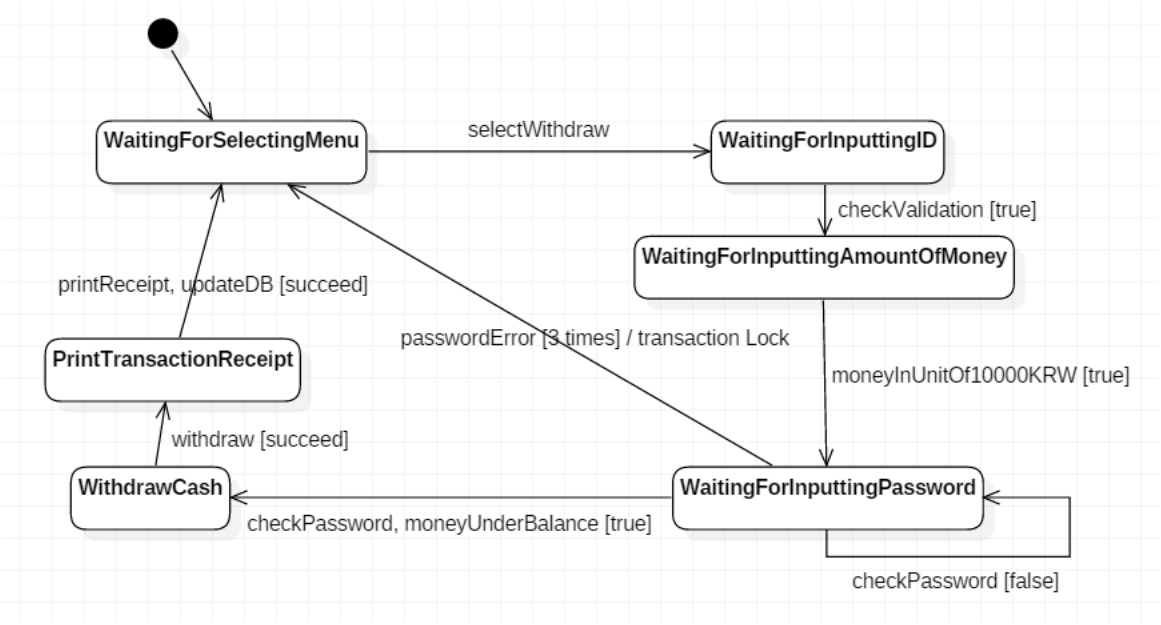
- Deposit



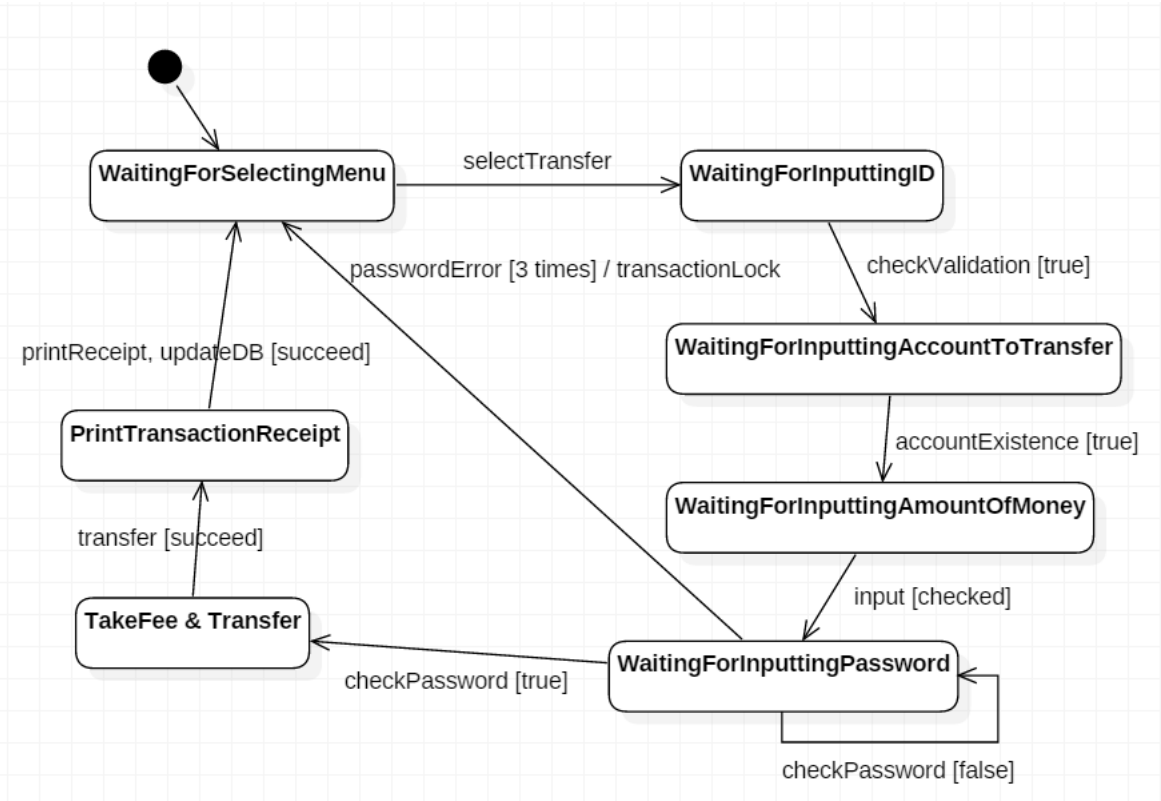
- Deposit Without Bankbook



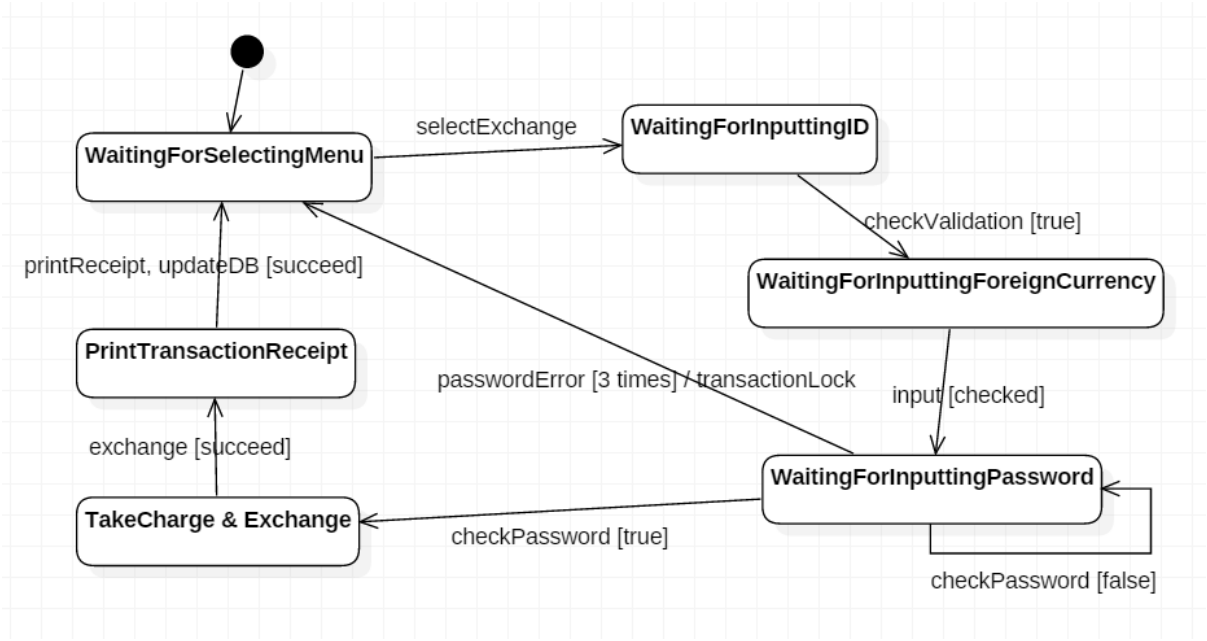
- Withdraw



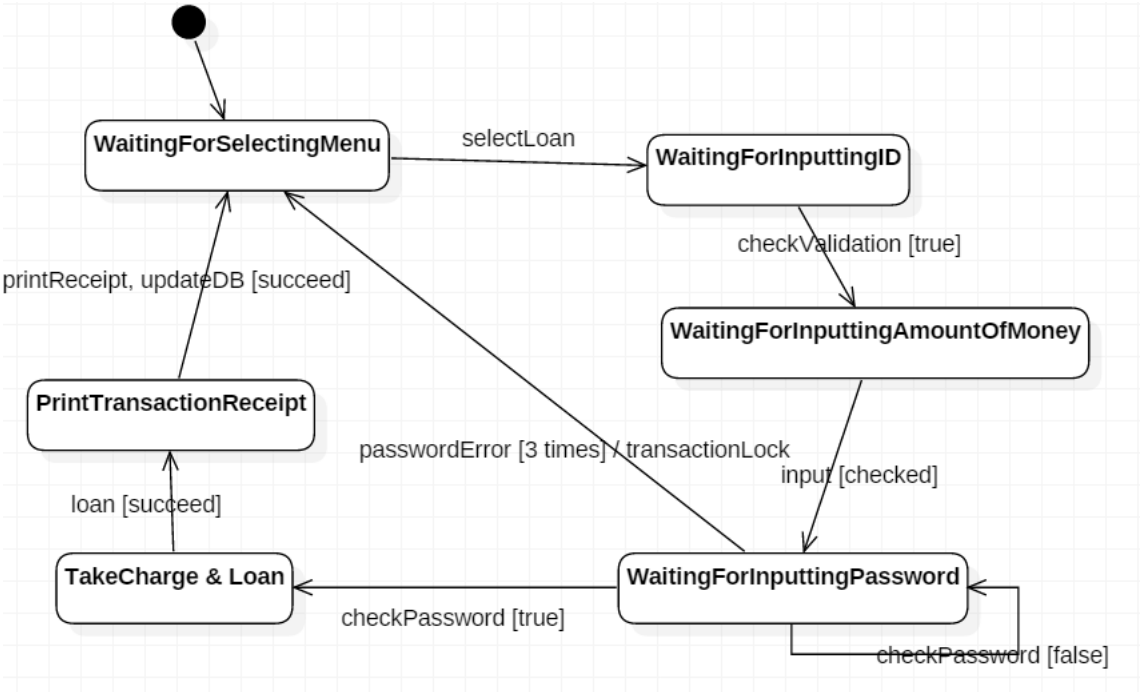
- Transfer



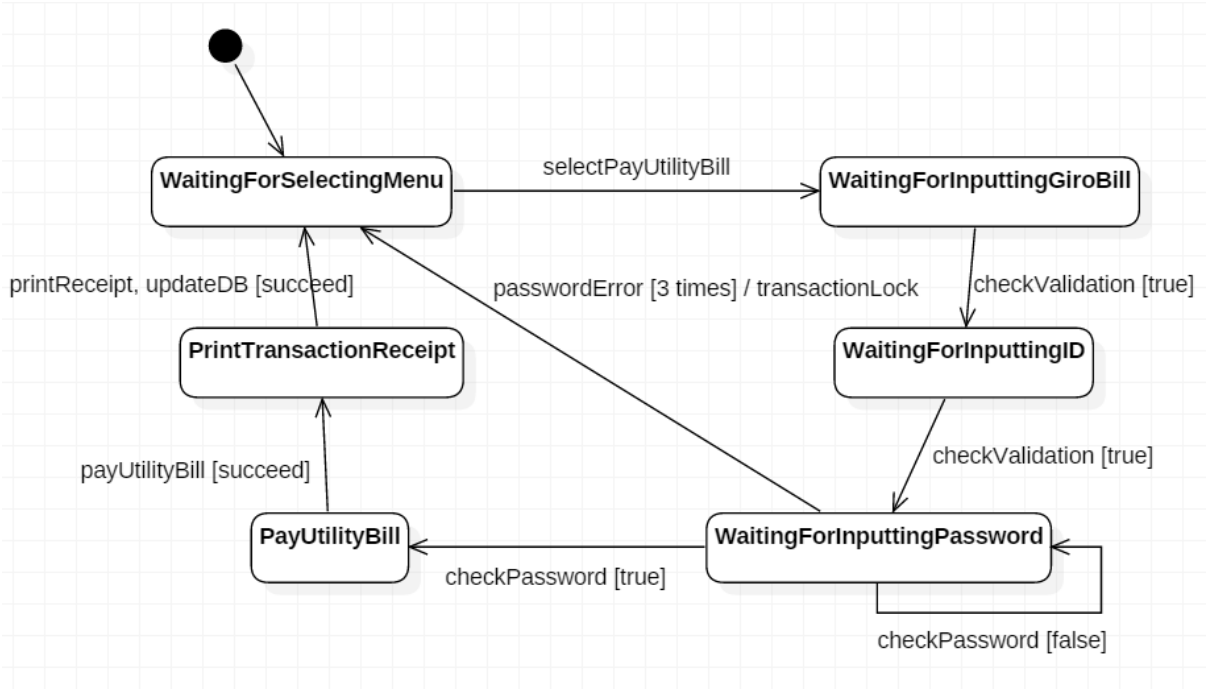
- Exchange



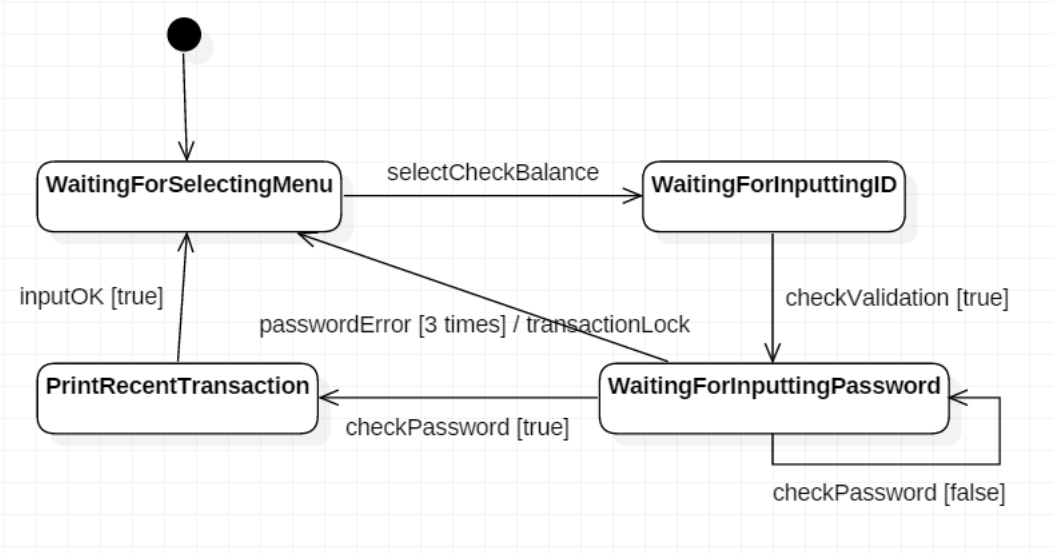
- Loan



- Pay Utility Bill



- Check Balance



Activity 2038. Refine System Test Case

<OOPT Stage 1000 v3 Activity 1008 참고>

Ref. #	Function	Use Case Number & Name	Test	Test description
R1.1	Deposit	1. Deposit	입금 TEST	-고객이 올바른 계좌를 입력했을 때만 입금이 가능한지 TEST - 고객이 입력한 계좌에 입금한 금액만큼 입금이 되는지 TEST - 만원 단위로 입금되는지 TEST - 신용카드를 이용할 시, 대출금이 상환되는지 TEST
R1.2	Deposit Without bankbook	2.Deposit Without Bankbook	무통장입금 TEST	- 고객이 올바른 계좌번호를 입력했을 때만 입금이 가능한지 TEST - ATM을 관리하는 은행과 동일한 은행의 계좌에만 입금이 되는지 TEST - 만원 단위로 입금되는지 TEST - 해당 계좌에 돈이 입금되었는지 TEST
R1.3	Withdraw	3. Withdraw	출금 TEST	- 고객이 올바른 계좌를 입력했을 때만 인출이 가능한지 TEST - 비밀번호가 맞아야 인출되는지 TEST - 만원 단위로만 인출되는지 TEST - 고객이 선택한 오만원권과 만원권의 장수대로 출금되는지 TEST - 계좌에 남아있는 잔고보다 더 많은 금액을 출금하려고 할 경우 출금을 할 수 없는지 TEST
R1.4	Transfer	4. Transfer	송금 TEST	- 고객이 본인의 계좌에 있는 돈을 보낼 수 있는지 TEST - 고객의 계좌에서 송금하려는 금액 + (타행의 경우)수수료가 인출되는지 TEST - 수신자의 계좌에 돈이 입금되었는지 TEST - 계좌의 비밀번호가 맞는 경우만 돈을 보내는지 TEST - 수신인의 정보가 정확해야 송금이 되는지 TEST
R1.5	Exchange	5. Exchange	한화에서 외화로 환전 TEST	- 고객이 올바른 계좌를 입력했을 때만 환전이 가능한지 TEST - 비밀번호가 맞아야 환전이 되는지 TEST - 나라마다 환율이 알맞게 적용되는지

				TEST - 고객의 계좌에서 환전하는 금액 + 수수료만큼 인출되는지 TEST
R1.6	Loan	6. Loan	대출 TEST	- 고객이 올바른 신용카드 번호를 입력했을 때만 대출이 되는지 TEST - 만원 단위로만 인출되는지 TEST - 한도를 넘지 않았을 경우에만 대출이 되는지 TEST - 비밀번호가 맞을 경우에만 대출이 되는지 TEST - 카드사에서 대출 금액 + 수수료만큼 대출했다는 정보가 제대로 처리되는지 TEST
R1.7	Pay Utility Bill	7. Pay Utility Bill	공과금 납부 TEST	- 고객이 올바른 계좌를 입력했을 때만 공과금을 납부 할 수 있는지 TEST - 지로고지서를 입력했을 때, 잔고가 충분한 경우 지로고지서에 입력되어 있는 금액만큼 납부가 되는지 TEST - 국가 계좌로 납부할 수 있는지 TEST - 비밀번호가 맞아야 납부할 수 있는지 TEST
R1.8	Check Balance	8.Check Balance	계좌 조회 TEST	- 고객이 올바른 계좌를 입력했을 때만 계좌 조회를 할 수 있는지 TEST - 비밀번호가 맞을 경우에만 계좌 조회가 되는지 TEST - 계좌의 최근 거래 내역(계좌의 경우 일시, 거래 종류, 거래 금액, 잔고, 신용카드의 경우 일시, 거래 종류, 대출 금액, 남은 한도)을 제대로 출력하는지 TEST
R2.1	Insert	9. Insert	매체 삽입 TEST	- 매체(계좌번호/신용카드/지로고지서)가 제대로 삽입되는지 TEST
R2.2	Print Transaction Receipt	10.Print Transaction Receipt	거래명세서 출력 TEST	- 각각의 프로세스가 끝난 후 거래명세서가 올바르게 출력되는지 TEST - 입금 : (계좌번호 이용 시) 거래 일시/입금 금액/잔고, 혹은 (신용카드 이용 시) 거래 일시/입금 금액/남은 대출 금액 - 무통장입금: 거래 일시/입금액 - 출금 : 거래 일시/출금 금액/남은 잔고 - 송금 : 거래 일시/송금 금액 + (타행일 경우) 수수료/송금 계좌/잔고 - 환전 : 거래 일시/환전 금액 + 수수료/잔고 - 대출 : 거래 일시/대출 금액/남은 한도

				- 공과금 납부: 거래 일시/납부 금액/잔고
R2.3	Print Error	11. Print Error	에러 출력 TEST	- 각종 에러들이 발생했을 경우 그에 알맞은 에러 메세지를 출력하는지 TEST (존재하지 않는 계좌/카드번호를 입력한 경우, 비밀번호를 잘못 입력한 경우, 한도/잔고 이상의 금액을 거래한 경우 등)
R2.4	Do Forced Termination	12. Do Forced Termination	강제 종료 TEST	- 오류가 3번 발생했을 경우 강제 종료가 되는지(처음 화면으로 돌아가는지) TEST
R3.1	Take Charge	13.Take Charge	수수료 부과 TEST	- 송금(타행의 경우)/대출/환전 시 수수료가 붙는지 TEST - ATM 내부에서 등급에 따라 수수료를 계산해 부과하는지 TEST
R4.1	Check Password	14.Check Password	비밀번호 확인 TEST	- 각 프로세스에서 올바른 비밀번호를 입력했을 때만 프로세스가 진행되는지 TEST - 필요한 프로세스 : 출금/송금/환전/대출/공과금 납부/계좌 조회
R5.1	Transaction Lock	15. Transaction Lock	거래 잠금 TEST	- 비밀번호를 3번 틀렸을 시 해당 계좌/카드로 거래를 할 수 없도록 계좌/카드를 잠그는지 TEST
R6.1	Check Validation	16.Check Validation	삽입된 매체의 유효성 TEST	- 삽입된 매체(계좌번호/신용카드/지로고지서)가 유효할 때만 프로세스가 진행되는지 TEST
R6.2	Update Database	17.Update Database	DB 업데이트 TEST	- 계좌 조회를 제외한 각 프로세스에서, 프로세스가 끝난 후 DB가 제대로 업데이트 되는지 TEST - 입금 : (계좌번호 이용 시) 거래 일시/입금 금액/잔고, 혹은 (신용카드 이용 시) 거래 일시/입금 금액/남은 대출 금액 - 무통장입금: 거래 일시/입금 금액/잔고 - 출금 : 거래 일시/출금 금액/남은 잔고 - 송금 : (송금한 계좌의 경우)거래 일시/송금 금액 + (타행일 경우) 수수료/송금 계좌/잔고, (송금 받은 계좌의 경우)거래 일시/입금 금액/송금한 계좌/잔고 - 환전 : 거래 일시/환전 금액 + 수수료/잔고 - 대출 : 거래 일시/대출 금액/남은 한도 - 공과금 납부: 거래 일시/납부 금액/잔고

Activity 2039. Analyze (2030) Traceability Analysis

Ref.#	Function	Use Case	Operation
R1.1	Deposit	Deposit	1. selectDeposit
R1.2	Deposit Without Bankbook	Deposit Without Bankbook	2. selectDepositWithoutBankbook
R1.3	Withdraw	Withdraw	3. selectWithdraw
R1.4	Transfer	Transfer	4. selectTransfer
R1.5	Exchange	Exchange	5. selectExchange
R1.6	Loan	Loan	6. selectLoan
R1.7	Pay Utility Bill	Pay Utility Bill	7. selectPayUtilityBill
R1.8	Check Balance	Check Balance	8. selectCheckBalance
R2.1	Insert	Insert	9. selectForeignCurrency
R2.2	Print Transaction Receipt	Print Transaction Receipt	10. insert
R2.3	Print Error	Print Error	11. inputCash
R2.4	Do Forced Termination	Do Forced Termination	12. inputAccountNumber
R3.1	Take Charge	Take Charge	13. inputAmountOfMoney
R4.1	Check Password	Check Password	14. inputPassword
R5.1	Transaction Lock	Transaction Lock	15. Validation
R6.1	Check Validation	Check Validation	
R6.2	Update Database	Update Database	