JackPot ATM

OOPT Stage 1000_ver.3

<Plan and Elaboration>

Team 5

Date

2018-04-16

201312259 백만일

201112052 방민석

201211383 조영래

Activity 1001. Define Draft Plan

Activity 1002. Create Preliminary Investigation Report

Activity 1003. Define Requirements

Activity 1004. Record Terms in Glossary

Activity 1005. Implement Prototype

Activity 1006. Define Draft System Architecture

Activity 1007. Define Business Use Case

Activity 1008. Define Business Concept Model

Activity 1009. Define System Test Case

Activity 1010. Refine Plan

Activity 1001. Define Draft Plan

1. Motivation

경제가 발전함에 따라 많은 사람들이 금융 거래를 많이 하게 되었고, 모든 금융 거래를 은행의 은행원이 모두 처리 할 수 없기에 은행원의 일부 업무를 대신 할 수 있는 시스템을 개발하게 되었다. 하지만 금융시장에서 ATM 기기는 현재 포화상태이며 특별한 기능을 하지 않는 ATM 대신 전자송금 어플리케이션 등을 통한 거래가 증가 추세에 있다. ATM기기를 통한 수수료 수입은 기업에게 있어서도 주요 수익원 중 하나이기 때문에 이미 막대한 ATM 설치에 막대한 설비투자를 마친 상황에서 고객들의 ATM 기기 사용 감소는 기업에게 위협의 요인이 될 수 있다. 이에 따라 팀 ¾ 은 사람들이 직접 와서 거래를 하고싶도록 만들기 위해 특별한 ATM 기기를 제작하고자 하였으며 게이미피케이션의 원리를 ATM 에 도입하여 본 ATM 거래자 중 1일 1명 랜덤으로 5만원을 추가 지급하는 Jackpot ATM 을 제작하게 되었다.

2. Project Objective

본 프로젝트의 목표는 기존 포화상태인 ATM 기기 시장에서 새로운 형태의 독특한 ATM 기기를 도입함으로써 해당 ATM 소프트웨어를 사용하는 금융기업의 이윤을 극대화시키는 것에 있다. Jackpot ATM 은 기존 ATM 이 가지고 있는 '신속하고 간편한 금융거래' 뿐만아니라 1일 1 Jackpot 기능을 현실성 있게 구현하여 ATM 기기로서의 차별성을 확보하고자 한다.

- 신속 정확한 금융 서비스
- 차별성을 가지는 Jackpot 기능 구현
- Jackpot 기능의 악용 가능성(ex, 한사람이 만원씩 지속적으로 출금하는 등)을 제거

3. Functional Requirements

- 조회,출금,송금,입금 메뉴 선택
- 카드번호 혹은 계좌번호(통장) 입력
- 계좌 비밀번호 입력
- 출금/입금/송금 금액 입력
- 명세서 출력 기능 제공
- 수수료 계산
- ATM 잔고 확인 및 추가
- Jackpot 계산

4. Non Functional Requirements

- 사용자가 보기 편한 화면을 제공해야 한다.
- 금융 거래 프로그램이므로 보안이 잘 되어야 한다.
- 신속한 금융 거래가 가능해야 한다.

5. Resource Estimation

- Human Efforts (M/M) : 3M / 3M
- Human Resource : 1 컴퓨터 공학 / 2 다전공
- Project Duration : 3개월
- Cost : 식비 (5,000) * 주당 미팅 횟수(2) * 학업주(12) * 전체인원(3) =360,000

6. Other Information

- None

Activity 1002. Create Preliminary Investigation Report

1. Alternative Solutions

- 이미 존재하는 ATM S/W 구매한다.
- 개발 전문 업체에 프로그램 제작을 요청한다.

2. Project Justification(Business Demands)

- Cost : 5000(인당 식비) * 2(주당 미팅 횟수) * 3(전체인원) * 12주 = 360,000원
- Duration : 3 months
- Risk : Plan management , OOPT skill, Java skill, UML skill, 시험 및 타 과목 공부
- Effect :

3. Risk Management

Risk	Probability	Significance	Weight
Plan management	**5**	**5**	**25**

OOPT skill 부족	**4**	**5**	**20**
Java sill 부족	**3**	**4**	**12**
UML skill 부족	**3**	**3**	**9**
시험 및 타 과목 공부	**5**	**5**	**25**

4. Risk Reduction Plan

Risk	Reduction plan
Plan Management	관련 서적이나 검색을 통해 도움을 받는다.
OOPT skill 부족	수업자료와 사이트를 참고해서 보완한다.
Java sill 부족	관련 서적이나 검색을 통해 도움을 받는다.
UML skill 부족	수업자료와 관련사이트, 질문을 통해 보완한다.
시험 및 타 과목 공부	서로 도와가며 시험에 대비하고, 타 과목 공부로 프로젝트에 영향을 미치는 일이 없도록 한다.

5. Market Analysis

- 현재 각 은행별 ATM은 물론 편의점에도 설치되어 있는 상황이다.
- 핸드폰을 이용해서 간단히 일을 처리할 수 있는 Application들로 인해 이용률이 낮아지고 있다.

6. Other Managerial issues

- 2018년 6월전까지 제작이 완료되어야 한다.

Activity 1003. Define Requirements

1. Functional Requirements

1) 조회

- 조회버튼을 터치
- 카드 혹은 통장을 넣으라는 메시지를 출력
- 카드 혹은 통장을 입력받으면 비밀번호를 입력하라는 메시지를 출력
- 비밀번호를 입력 받으면 현재 잔고와 거래내역을 보여줌

2) 출금

- 출금 버튼 터치
- 카드 혹은 통장을 넣으라는 메시지를 출력
- 카드 혹은 통장을 넣으면 출금 금액을 입력하라는 메시지를 출력
- 출금금액을 입력받으면 수수료를 더한 금액을 통장잔고와 비교
- 잔고보다 많은 금액이면 오류
- 출금금액을 ATM 잔고와 비교
- 잔고보다 많은 금액이면 오류
- 통장 잔고와 ATM 잔고보다 적은금액이면 출금을 위한 비밀번호 입력 메시지를 보여줌
- 비밀번호가 틀리면 오류
- 비밀번호가 맞으면 명세서 출력 확인 맞으면 출력 아니면 생략
- 이후 출금금액과, 출금 후 잔고를 보여줌
- 출금금액을 ATM 잔고에서 빼줌
- 거래후 잔고를 출력

3) 입금

- 입금 버튼을 터치
- 카드 혹은 통장을 넣으라는 메시지를 출력
- 카드 혹은 통장을 넣으면 입금 금액을 입력하라는 메시지를 출력
- 입금금액과 잔고금액을 더함
- 이후 입금금액과 입금 후 잔고를 보여줌
- 입금 금액을 ATM 잔고에 더해줌
- 거래 후 잔고를 출력

4) 송금

- 송금 버튼을 누르면 카드 혹은 통장을 넣으라는 메시지를 보여줌

- 카드 혹은 통장을 넣으면 계좌번호를 입력하도록 함
- 계좌번호가 없으면 오류
- 계좌번호가 맞으면 송금 금액을 입력하라는 메시지를 보여줌
- 송금 금액을 입력받으면 수수료를 더해서 계좌 잔고와 비교
- 잔고가 부족하면 에러 출력
- 잔고가 충분하면 비밀번호 입력
- 비밀번호가 다르면 오류
- 비밀번호가 같으면 카드 잔고에서 송금금액만큼을 빼줌
- 송금금액 만큼을 입력 계좌에 더해줌
- 거래후 잔고를 출력

5) Jackpot

- 랜덤으로 N번째 출금을 한 고객에게 5만원을 추가적으로 지급한다

6) manage_ATM

- ATM 잔고를 추가하거나 뺄 수 있다.
- ATM 잔고가 일정수준(10,000원권 30매, 50,000권 30매) 이하가 되면 잔고 메인화면 하단부에 잔고 부족 메시지를 출력한다
- 각 화폐별 최대 수량은 500매이다.

Ref.#	Function	Category
R.1.1	jackPot	Hidden
R.1.2	withdraw	Evident
R.2	deposit	Evident
R.3	remittance	Evident
R.4	view_account_detail	Hidden
R.5	manage_ATM	Evident

2. Performance Requirements

1) 모든 입력에 대한 기기의 반응은 1초 이내로 이루어진다.

3. Operating Environments

1) OS : Windows 7, Windows 10
2) 개발언어: JAVA

4. Develop Environments

1) OS: Windows7, Windows 10, Mac
2) IDE : Eclipse
3) 개발언어: JAVA

5. Interface Requirements

1) 입금/출금/송금/조회 버튼
2) 숫자 입력 버튼
3) 거래 결과 화면

6. Other Requirements

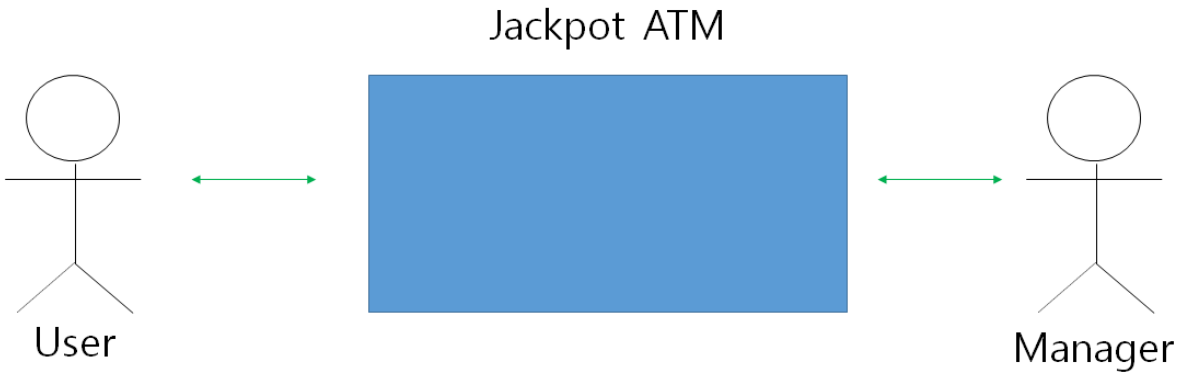
1) 공동 문서 편집을 위한 구글 계정

Activity 1004. Record Terms in Glossary

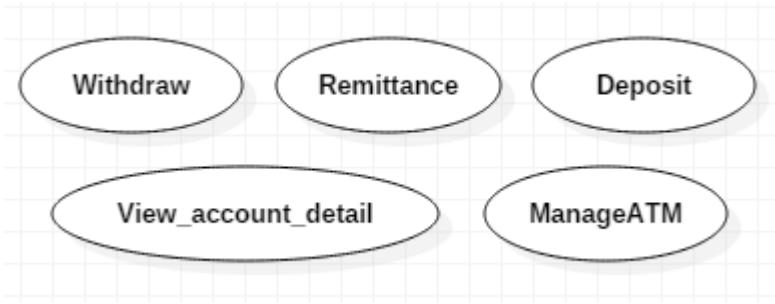
Term	Description	Remarks
withdraw	이용자가 통장에서 돈을 빼기 위한 작업	
deposit	이용자가 통장에 돈을 넣기 위한 작업	
remittance	이용자가 다른 계좌에 돈을 보내기 위한 작업	
account	계좌	
manage	관리	
jackPot	임의의 이용자에게 5만원을 추가로 지급하는 기능	

Activity 1006. Define Draft System Architecture

1. **Define System Boundary**



2. **Identify and Describe Actors**
    a. User : ATM을 이용하는 사람
    b. Manager : ATM의 잔고를 확인하고 관리하는 사람
3. **Identify Use-Case**
    a. Use-Cases by Actor-Based



b. Use-Cases by Event-Based

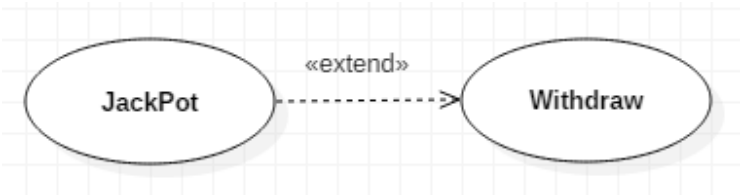
4. Allocate System Functions into Related Use-Cases

Ref.	Function	Use-Case Number & Name	Remarks
R.1.1	jackPot	1.jackPot	
R.1.2	withdraw	2.withdraw	
R.2	deposit	3.deposit	
R.3	remittance	4.remittance	
R.4	view_account_detail	5.view_account_detail	
R.5	manage_ATM	6.manage_ATM	

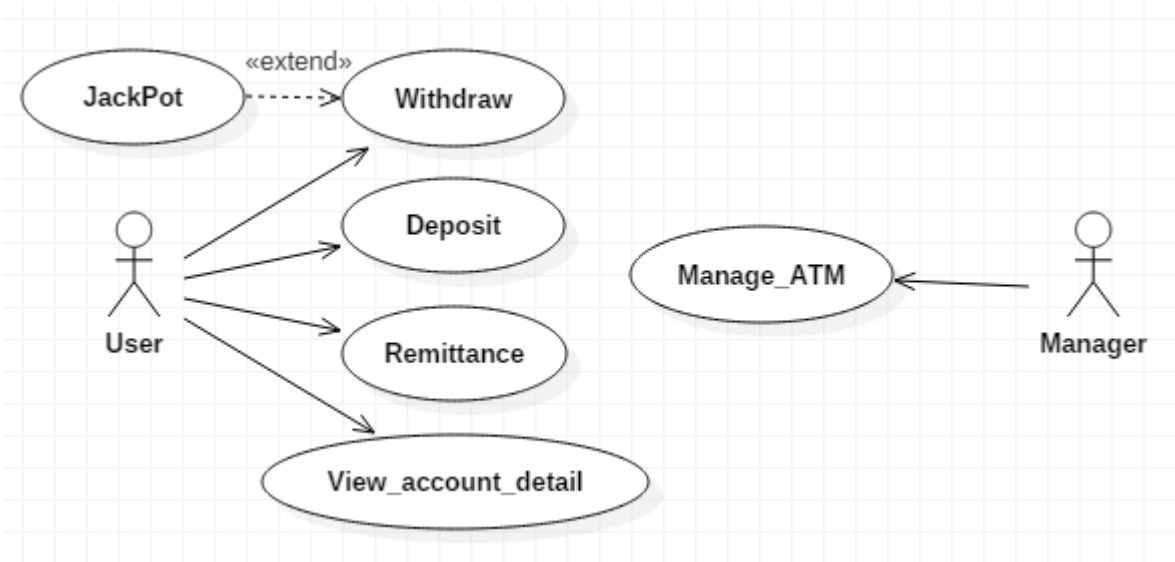
5. Categorize Use-Cases

Ref.	Function	Use-Case Number & Name	Category
R.1.1	jackPot	1.jackPot	Primary
R.1.2	withdraw	2.withdraw	Primary
R.2	deposit	3.deposit	Primary
R.3	remittance	4.remittance	Primary
R.4	view_account_detail	5.view_account_detail	Primary
R.5	manage_ATM	6.manage_ATM	Primary

6. Identify Relationships between Use-Cases



7. Draw a Use-Case Diagram



8. Describe Use-Cases

Use-Case Name	Description
1.withdraw	- 유저가 출금 버튼을 누르면 시작되는 Use Case - 카드 혹은 통장을 넣으라는 메시지를 출력 - 카드 혹은 통장을 넣으면 출금 금액을 입력하라는 메시지를 출력 - 출금금액을 입력받으면 수수료를 더한 금액을 통장잔고와 비교 - 잔고보다 많은 금액이면 오류 - 출금금액을 ATM 잔고와 비교
Actor	
User	

	- 잔고보다 많은 금액이면 오류 - 통장 잔고와 ATM 잔고보다 적은금액 이면 출금을 위한 비밀번호 입력 메시지 출력 - 비밀번호가 틀리면 오류 - 비밀번호가 맞으면 명세서 출력 확인 맞으면 출력 아니면 생략 - 이후 출금금액과, 출금 후 잔고를 보여줌 - 출금금액을 ATM 잔고에서 빼줌

Use-Case Name	**Description**
2.deposit	- 유저가 입금 버튼을 터치하면 시작되는 Use Case - 카드 혹은 통장 넣으라는 메시지 출력 - 카드 혹은 통장 넣으면 입금할 금액 입력하라는 메시지 출력 - 입금금액과 잔고금액 더함 - 이후 입금금액과 입금 후 잔고 보여줌 - 입금 금액 ATM 잔고에 더함 - 명세서 출력 확인 맞으면 출력 아니면 생략
Actor	
User	

Use-Case Name	**Description**
3.remittance	- 유저가 송금 버튼을 누르면 시작하는 Use Case - 카드 혹은 통장을 넣으라는 메시지를 보여줌 - 카드나 통장을 넣으면 계좌번호를 입력하도록 함
Actor	
User	

	- 계좌번호가 없으면 오류 - 계좌번호가 맞으면 송금 금액을 입력하라는 메시지를 보여줌 - 송금 금액 입력받으면 수수료를 더해서 계좌 잔고와 비교 - 잔고가 부족하면 에러 출력 - 잔고가 충분하면 비밀번호 입력 - 비밀번호가 다르면 오류 - 비밀번호가 같으면 카드 잔고에서 송금액만큼을 빼줌 - 송금액 만큼을 입력 계좌에 더해줌 - 성공한 거래의 송금액/입금계좌/이름을 보여줌 - 명세서 출력 확인 맞으면 출력 아니면 생략

Use-Case Name	**Description**
4.view_account_detail	- 유저가 조회버튼을 터치하면 시작되는 Use Case - 카드 혹은 통장을 넣으라는 메시지를 출력 - 카드 혹은 통장을 입력받으면 비밀번호를 입력하라는 메시지를 출력 - 비밀번호를 입력 받으면 현재 잔고와 거래내역을 보여줌
Actor	
User	

Use-Case Name	**Description**

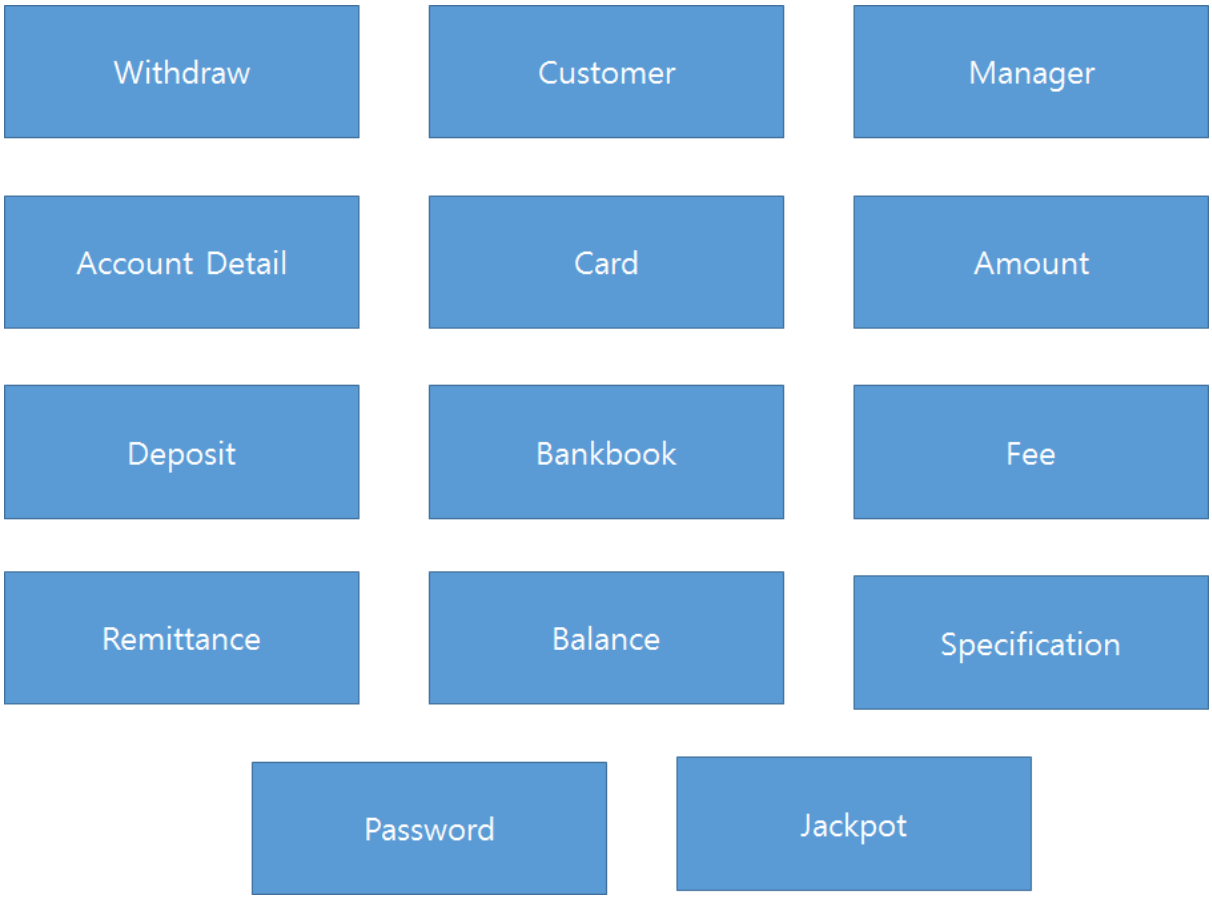
5.jackPot	- 랜덤으로 N번째 출금을 한 고객에게 5만원을 추가적으로 지급하는 Use case
Actor	
User	

Use-Case Name	**Description**
6.manage_ATM	- 관리자가 ATM 잔고에 문제가 있음을 알았을때 관리하기 위한 Use case - ATM 잔고를 추가하거나 뺄 수 있다.
Actor	
None	

9. Rank Use-Cases

Use-Case Name	**Rank**
jackPot	High
withdraw	High
deposit	High
remittance	High
view_account detail	High
manage_ATM	High

Activity 1007. Define Business Use Case



Activity 1008. Define Business Concept Model

Activity 1009. Define System Test Case

Test Number	Test 항목	Description	Use Case	System Function
1	출금	유효한 계좌를 입력받으면 이용자가 입력한 금액을 계좌의 잔고와 ATM 잔고를 확인해서 제대로 이용자에게 지급하는지 확인하고, 명세서를 출력하는지 확인	withdraw	
2	입금	유효한 계좌를 입력받으면 이용자가 넣은 금액을 통장 잔고에 제대로 더하고, ATM 잔고를 확인하고 명세서를 출력하는지 확인	deposit	
3	송금	유효한 계좌를 입력받으면 이용자가 입력한 금액을 알맞은 계좌에 입금하는지 확인하고, 이용자의 계좌잔고가 줄어드는지 확인	remittance	
4	조회	유효한 계좌를 입력받으면 계좌의 현재 잔고와 거래내역을 보여줌	view_account_detail	
5	ATM 잔고 증가 시험	관리자가 입력한 금액만큼 ATM 잔고가 제대로 증가하는지 시험한다	manage_ATM	

6	Jackpot 출금 시험	랜덤 N번째 고객의 출금이 성공적으로 이루어질 때 추가로 5만원이 출금되는지 확인	Random_Jackpot	

Activity 1010. Refine Plan

1. Project Scope

팀 ¾은 기존 ATM 기기 시장의 포화라는 위협요인에서 케이미피케이션의 요소를 활용한 Jackpot ATM 을 제작하여 차별화된 형태의 ATM 기기 시장의 문을 열고 이를 통하여 고객들이 같은 조건이라면 Jackpot ATM 기기를 이용하도록 하여 사용자를 늘려 궁극적으로는 수수료 수익을 극대화 하는것에 있다.

2. Project Objectives

Jackpot ATM 기기를 사용하는 고객으로 하여금 기본적으로 ATM 기기가 가지고 있는 신속하고 정확한 금융거래뿐만 아니라 추가적인 보상을 제공하여 고객의 이용률을 증대시킴에 있어 오남용을 막을수 있도록 소프트웨어를 구축하고, Jackpot 이라는 보상을 제공할 때에도 그 투명성을 지키기 위해 랜덤하게 당첨자가 도출되도록 한다.

- 사용자가 금융거래를 신속하고 정확하게 할 수 있도록 한다.
- Jackpot 이라는 추가적인 보상을 제공한다.
- Jackpot 의 오남용이 일어나지 않을 수 있도록 제어한다.

3. Functional Requirements

Ref.#	Function	Category
R.1.1	jackPot	Hidden
R.1.2	withdraw	Evident
R.2	deposit	Evident
R.3	remittance	Evident

R.4	view_account_detail	Evident
R.5	manage_ATM	Evident

4. Performance Requirements

- 기기조작에 대한 반응은 1초 이내로 이루어 져야한다.
- 금융 거래는 1분 이내로 이루어 져야한다

5. Operating Environment

- OS : Windows 7, Windows 10
- IDE : Eclipse
- 개발언어: JAVA

6. User Interface Requirements

- 메뉴 선택을 통한 접근방식
- 터치스크린에 기반을 둔 개발

7. Resource Estimation

- Human Efforts (M/M) : 3M / 3M
- Human Resource : 1 컴퓨터공학 / 2 다전공
- Project Duration : 3개월
- Cost : 식비 (5,000) * 주당 미팅 횟수(2) * 학업주(12) * 전체인원(3) =360,000

8. Scheduling

Stage	Phase(00X0)/Activity(000X)	Schedule(Week)											
		1	2	3	4	5	6	7	8	9	10	11	12
1000. Plan & Elaboration	1001. Define Draft Plan												
	1002. Create Preliminary Investigation												
	1003. Define Requirements												
	1004. Record Terms in Glossary												

	1005. Implement Prototype												
	1006. Define Business Use Case												
	1007. Define Business Concept Model												
	1008. Define Draft System Architecture												
	1009. Define System Test Case												
	1010. Refine plan												
2000. Build	2010. Revise Plan												
	2020. Synchronize Artifacts												
	2030. Analyze												
	2031. Define Essential Use Cases												
	2032. Refine Use Case Diagrams												
	2033. Define Domain Model												
	2034. Refine Glossary												
	2035. Define System Sequence Diagrams												
	2036. Define Operation Contracts												
	2037. Define State Diagrams												
	2040. Design												
	2041. Design Real Use Cases												
	2042. Define Reports, UI, and Storyboards												
	2043. Refine System Architecture												
	2044. Define Interaction Diagrams												
	2045 Define Design Class Diagrams												

	2046. Design Traceability Analysis												
	2050. Construct												
	2051. Implement Class & Interface Definition												
	2052. Implement Methods												
	2053. Implement Windows												
	2054. Implement Reports												
	2055. Implement DB Schema												
	2056. Write Test Code												
	2060. Test												
	2061. Unit Testing												
	2062. Integration Testing												
	2063. System Testing												
	2064. Performance Testing												
	2065. Acceptance Testing												
	2066. Documentation Testing												