
Software Modeling & Analysis

[OOPT stage 1000]

No.	T7
Subject	Software Modeling & Analysis
Professor	JUNBEOM YOO
Team Member	201410621 한상민
	201514208 박종건
	201615007 문기태

Chart.

Activity 1001. Define Draft Plan

- 1) Motivation
- 2) Project Objectives
- 3) Functional Requirements
- 4) Non-Functional Requirements
- 5) Resource Estimation

Activity 1002. Create Preliminary Investigation Report

- 1) Alternative Solutions
- 2) Project Justification (Business Needs)
- 3) Risk Management
- 4) Risk Reduction Plan
- 5) Market Analysis
- 6) Managerial Issue

Activity 1003. Define Requirements

- 1) Functional Requirements
- 2) Performance Requirements
- 3) Operating Environments

Activity 1004. Record Terms in Glossary

Activity 1006. Define Business Use Case

- 1) Define System Boundary
- 2) Identify and Describe Actors
- 3) Identify Use-Case
 - A. Actor-Based
 - B. Event-Based
- 4) Allocate system functions into related use cases
- 5) Categorize use cases
- 6) Draw a Use-Case Diagram
- 7) Describe Use-Case
- 8) Rank Use-Case

Activity 1007. Define Business Concept Model

Activity 1008. Define Draft System Architecture

Activity 1009. Define System Test Case

Activity 1010. Refine Plan

- 1) Project Scope
- 2) Project Objectives
- 3) Functional Requirements
- 4) Performance Requirements
- 5) Operating Environments
- 6) Resources
- 7) Plan Scheduling

Activity 1001. Define Draft Plan

1. Motivation

현시대 은행의 전산화 과정에서 카드 및 통장 등 가상에서 화폐를 취급하는 경우가 많아지고 있다. 그럼에도 불구하고 현금을 요구하는 매장들이 아직 많기 때문에 필요할 때 마다 꼭 자신의 은행을 가서 현금을 인출하는 것이 아닌 주위에 흔히 있는 편의점에 은행간 공용으로 이용할 수 있는 현금 지급기를 설치함으로써 일반 대중들의 편의를 지향하고자 ATM기기에 들어갈 프로그램을 만들게 되었다

2. Project Objectives

카드 나 통장을 이용하여 현금을 인출할 수 있으며 송금 혹은 입금까지 은행에서 제공되는 기본적인 은행 서비스를 제공할 수 있다. 다만 혹시 모를 범죄를 염두에 두어 인출한도와 송금한도를 설정하여 피해를 감소시킨다.

3. Functional Requirements

-Send Money

- ATM을 통하여 **고객의 계좌에서 다른 계좌로 돈을 보낼 수 있다.**
- 송금을 수행할 경우 **수수료**가 부과된다.
- 송금을 하려면 해당 계좌의 **비밀번호**를 입력해야 하고 **다른 계좌의 은행과 계좌번호**를 입력 해야 한다.
- 보낼 금액보다 잔액이 더 적을 경우 거래가 취소된다.
- 고객이 설정한 **송금 한도**보다 많은 금액을 보내려 할 경우 거래가 취소된다.
- 거래 도중에 언제든지 취소를 할 수 있다.
- 거래가 끝나면 거래 내용을 업데이트한다.
- 거래가 끝나고 송금에 대한 **명세서**를 출력할 수 있다.

-Withdraw Money

- 고객은 ATM을 통하여 **계좌에서 돈을 뺄 수 있다.**
- 인출을 수행할 경우 **수수료**가 부과된다.
- 인출을 하려면 해당 계좌의 **비밀번호**를 입력해야 한다.
- 인출할 금액보다 잔액이 더 적을 경우 거래가 취소된다.
- 고객이 설정한 **인출 한도**보다 더 많은 금액을 보내려 할 경우 거래가 취소된다.
- 거래 도중에 언제든지 취소를 할 수 있다.
- 거래가 끝나면 거래 내용을 업데이트한다.
- 거래가 끝나고 송금에 대한 **명세서**를 출력할 수 있다.

-Deposit Money

- ATM을 통하여 **계좌에 돈을 입금 할 수 있다.**
- 입금을 하려면 고객이 돈을 직접 넣어야 한다.
- 거래 도중에 언제든지 취소를 할 수 있다.
- 거래가 끝나면 거래 내용을 업데이트한다.
- 거래가 끝나고 송금에 대한 **명세서**를 출력할 수 있다.

-Check Remain Money

- ATM을 통하여 **고객의 계좌에 있는 잔액을 확인할 수 있다.**
- 잔액 조회를 수행하려면 **비밀번호**를 입력 받아야 한다.
- 거래 도중에 언제든지 취소를 할 수 있다.
- 거래가 끝나고 송금에 대한 **명세서**를 출력할 수 있다.

-Check Bank

-Select Bank

-Find Account

- Update Account
- Read Account Information
- Input password
- Check Password
- Count Commission
- Check Business Hours
- Limit Amount
- Check Amount
- Print Statement
- Return To Main

4. Non-Functional Requirement

- Effective UI
- Comfortable Control
- Fast Process Work (within 1min.)

5. Resource Estimation

- Human Efforts: 3man, 4month
- Human Resources: 3 programmers
- Duration: 4 month
- Budget

(단위:만원)

	부분 계	비고
인건비	0	
활동비	150	
계	150	

Activity 1002. Create Preliminary Investigation Report

1. Alternative Solutions

- 시중에 있는 ATM기의 소프트웨어를 사용한다
- 타 업체의 외주를 부탁한다.

2. Project Justification (Business Needs)

- cost: 인건비가 들지 않는다는 장점이 있다
- duration: 4month
- Risk: OOAD JAVA UML 등에 대한 숙달 부족, 타 실제 은행과의 협약 필요
- Effect: 유지보수에 용이하다. 관리자 권한으로 쉽게 접근할 수 있다.

3. Risk Management

Risk	Probability	Significance	Weight
OOPT Skill	3	5	15
Programming Skill	3	5	15
UML Skill	1	3	3
Mid&Final Exam	10	10	100

4. Risk Reduction Plan

Risk	Reduction Plan
OOPT Skill	교수님 및 조교님의 수업을 열심히 따라간다
Programming Skill	관련 서적을 참고한다
UML Skill	관련 서적과 조교님의 도움을 받는다
Mid&Final Exam	프로젝트와 수업을 골고루 병행하며 열심히 한다

5. Market Analysis

-현재 이미 많이 분포 되어 있고 독점적으로 운영하고 있는 기업이 있으므로 진입이 다소 어려울 수도 있다.

-여러 ATM의 소프트웨어의 디자인이 모두 비슷하기 때문에 디자인의 구성요소에 대해 생각할 필요는 없다.

6. Managerial Issue

-제한된 시간 안에 개발을 완료 해야 된다.

-개발언어에 대한 자문을 구하기 위한 계획을 추가적으로 세워야 한다.

Activity 1003. Define Requirements

1. Functional Requirements (Rev. Activity1001)

Function	Description
Send Money	송금을 실시한다
Withdraw Money	인출을 실시한다
Deposit Money	입금을 실시한다
Check Remain Money	잔액조회를 실시한다
Check Bank	입력한 계좌와 해당 은행을 조회한다.
Select Bank	송금할 대상의 은행을 선택한다.
Find Account	송금할 대상의 계좌를 찾는다
Update Account	은행 업무 뒤 계좌의 내역을 갱신하다
Read Account Information	현재 계좌정보를 불러온다
Input Password	비밀번호를 입력한다
Check Password	입력한 비밀번호가 맞는지 확인하다
Count Commission	수수료를 계산한다
Check Business-Hours	은행 별 영업시간을 확인한다.
Limited Amount	출금 송금 한도를 나타낸다
Check Amount	입금금액을 확인시켜준다
Print statement	명세서를 출력한다
Return To Main	메인 화면으로 돌아간다

Reference #	Function	Category
1-1	Check Bank	Event
1-2	Read Account Information	Hidden
1-3	Input Password	Event
1-4	Check Password	Hidden
1-5	Find Account	Event
1-6	Select Bank	Event
2-1	Send Money	Event
2-2	Withdraw Money	Event
2-3	Deposit Money	Event
2-4	Check Remain Money	Event
3-1	Check Amount	Event
3-2	Limited Amount	Hidden
4-1	Count Commission	Hidden
4-2	Check Business-hours	Hidden
5-1	Update Account	Event
5-2	Print statement	Event
5-3	Return to Main	Hidden

2. performance requirement

-송금 과정이 1분 이내로 되어야 한다

-계좌정보를 형식에 맞게 텍스트 파일로 저장해야 한다

-명시되어 있는 은행만을 사용하여야 한다

3. operating Environments

-OS: window7 & window10

-IDE: Eclipse

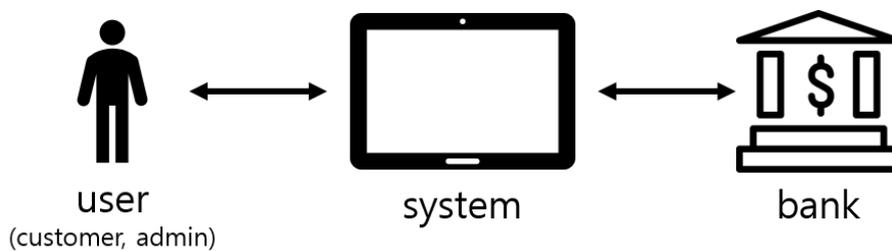
-programming language: Java

Activity 1004. Record term in Glossary

Glossary	Description
withdraw	(계좌에서 돈을) 인출하다
account	계좌
password	비밀번호
check	확인하다, 점검하다
limit	한계, 한도
deposit	예금, 입금
ID	신분증명서, 신분증 (identity 또는 identification의 약어)
remain	남아있는, 잔여의
statement	입출금 명세서
information	정보
admin	관리자(administrator의 약어)
business-hours	영업시간
input	투입, 입력
update	갱신하다,

Activity 1006. Define Business Use Case

1. Define System Boundary



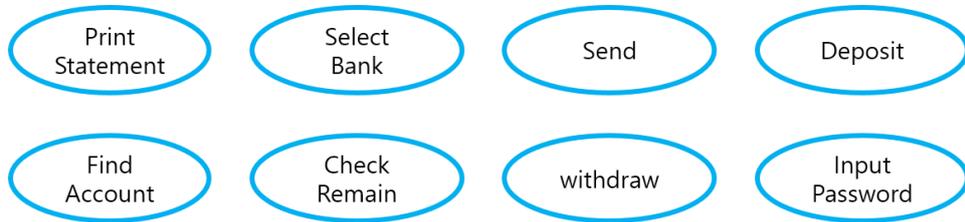
2. Identify and Describe Actors

-Customer: 은행의 일반 업무(입출금, 송금, 잔액확인 등)를 수행할 수 있다.

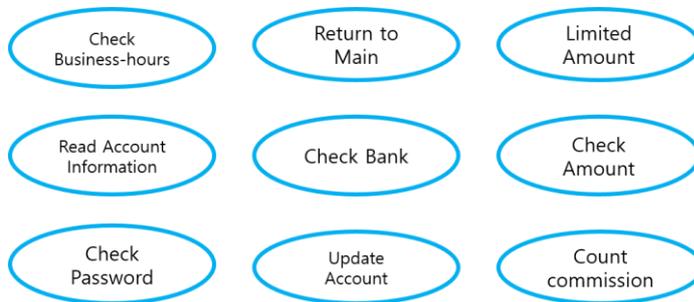
-Admin: ATM을 관리할 수 있는 권한(ATM 총액확인)을 가지고 있다.

3. Identify Use-Case

A. Actor-Based



B. Event-Based



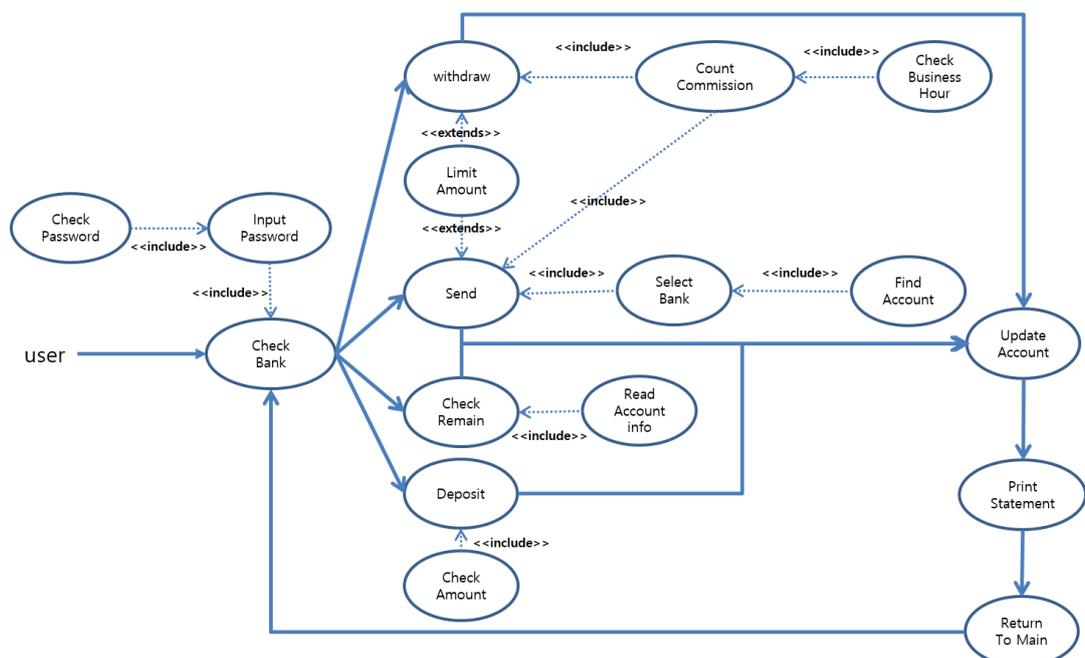
4. Allocate system functions into related use cases

Reference #	Function	Use case Number &Name	Category
1-1	Check Bank	3. Check Bank	
1-2	Read Account Information	4. Read Account Information	
1-3	Input Password	5. Input Password	
1-4	Check Password	6. Check Password	
1-5	Find Account	7. Find Account	
1-6	Select Bank	8. Select Bank	
2-1	Send Money	9. Send Money	
2-2	Withdraw Money	10. Withdraw Money	
2-3	Deposit Money	11. Deposit Money	
2-4	Check Remain Money	12. Check Remain Money	
3-1	Check Amount	13. Check Amount	
3-2	Limited Amount	14. Limited Amount	
4-1	Count Commission	15. Count Commission	
4-2	Check Business-hours	16. Check Business-hours	
5-1	Update Account	17. Update Account	
5-2	Print statement	18. Print statement	
5-3	Return to Main	19. Return to Main	

5. Categorize use cases

Reference #	Function	Use case Number &Name	Category
1-1	Check Bank	3. Check Bank	Primary
1-2	Read Account Information	4. Read Account Information	Primary
1-3	Input Password	5. Input Password	Primary
1-4	Check Password	6. Check Password	Primary
1-5	Find Account	7. Find Account	Primary
1-6	Select Bank	8. Select Bank	Primary
2-1	Send Money	9. Send Money	Primary
2-2	Withdraw Money	10. Withdraw Money	Primary
2-3	Deposit Money	11. Deposit Money	Primary
2-4	Check Remain Money	12. Check Remain Money	Primary
3-1	Check Amount	13. Check Amount	Primary
3-2	Limited Amount	14. Limited Amount	Primary
4-1	Count Commission	15. Count Commission	Primary
4-2	Check Business-hours	16. Check Business-hours	Primary
5-1	Update Account	17. Update Account	Primary
5-2	Print statement	18. Print statement	Primary
5-3	Return to Main	19. Return to Main	Primary

6. Draw a Use-Case Diagram



7. Describe Use-Case

Use Case	1. Withdraw
Actors	User
Descriptions	고객이 ATM기에서 현금을 인출한다.

Use Case	2. Check Remain
Actors	User
Descriptions	예금 조회를 실시 한다

Use Case	3. Deposit
Actors	User
Descriptions	통장에 돈을 입금한다

Use Case	4. Send
Actors	User
Descriptions	카드 혹은 통장을 사용해 돈을 이체한다

Use Case	5. Print Statement
Actors	User
Descriptions	모든 작업이 끝난 후 명세서를 출력한다

Use Case	6. Input Password
Actors	User
Descriptions	카드 혹은 통장 비밀번호를 입력한다

Use Case	7. Find Account
Actors	User
Descriptions	송금 대상의 계좌 번호를 입력하고 조회한다

Use Case	8. Select Bank
Actors	User
Descriptions	송금 대상의 은행을 선택한다

Use Case	9. Check Password
Actors	System
Descriptions	입력된 비밀번호를 검증한다

Use Case	10. Check Bank
Actors	System
Descriptions	해당 계좌의 은행을 조회하고 인증한다

Use Case	11. Return to Main
Actors	System
Descriptions	작업 종료 후 기본 화면으로 돌아간다

Use Case	12. Read Account Information
Actors	System
Descriptions	해당 계좌의 정보를 불러온다

Use Case	13. Limited Amount
Actors	System
Descriptions	출금 한도 이체 한도 등을 넘기는 지 확인한다.

Use Case	14. Check Amount
Actors	System
Descriptions	입금 시킨 금액을 확인 시켜준다

Use Case	15. Open File
Actors	System
Descriptions	계좌 정보가 적혀 있는 텍스트 파일을 연다

Use Case	16. Save File
Actors	System
Descriptions	갱신된 계좌의 정보를 다시 텍스트 파일에 저장한다

Use Case	17. Check Business Hour
Actors	System
Descriptions	야간과 주간의 수수료 부과를 위해 영업시간을 확인하다

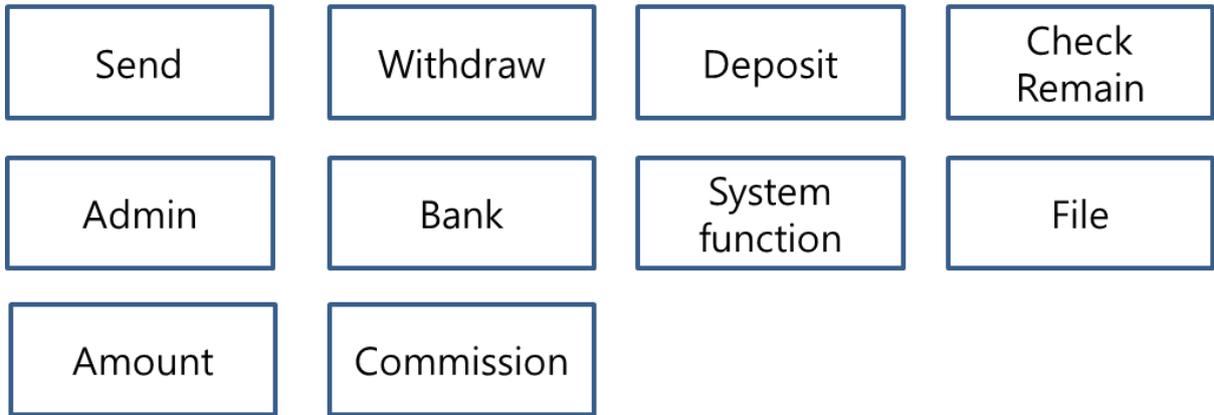
Use Case	18. Count Commission
Actors	System
Descriptions	송금 출금 등을 할 때 수수료를 부과 한다

Use Case	19. Update Account
Actors	System
Descriptions	계좌의 정보를 갱신한다

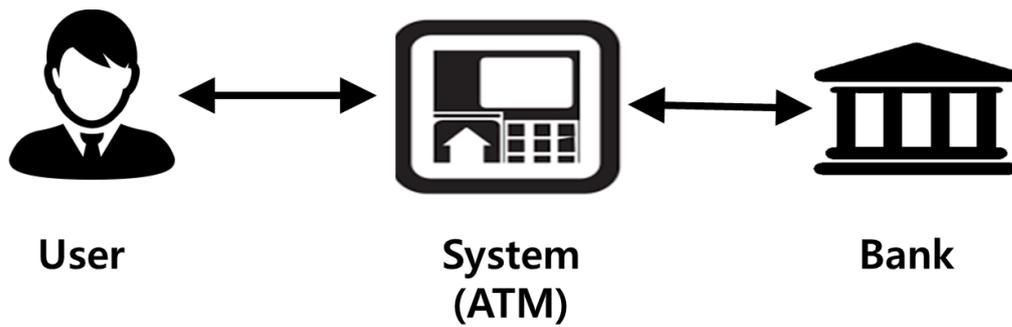
8. Rank Use-Case

Use-Case Number & Name	Description
1. Send	High
2. Withdraw	High
3. Deposit	High
4. Check Remain Money	High
5. Check Bank	High
6. Read Account information	High
7. Input Password	High
8. Check Password	High
9. Find Account	High
10. Check Total Amount	Low
11. Limited Amount	High
12. Count Commission	High
13. Check Business-hours	High
14. Update Account	High
15. Print statement	High
16. Return To Main	High
17. Select Bank	High

Activity 1007. Business Concept Model



Activity 1008. Define Draft System Architecture



Activity 1009. Define System Test Case

Test	Function	Description
클래스 간 변수가 제대로 넘어가는지 확인한다	Send	송금을 실시한다
	Withdraw	인출을 실시한다
	Deposit	입금을 실시한다
갱신된 계좌 정보가 잘 반영되는지 확인한다	Check Remain Money	잔액조회를 실시한다
계좌의 정보가 잘 넘어오는지 확인한다	Check Bank	해당계좌의 은행을 조회하고 인증한다.
텍스트 파일이 제대로 열리는지 확인한다	Read Account information	현재 계좌정보를 불러온다
비밀번호가 입력되는지 확인한다	Input Password	비밀번호를 입력한다

비밀번호를 잘 판별하는지 확인한다	Check Password	입력 비밀번호를 확인하다
해당 계좌를 텍스트 파일에서 찾을 수 있는지 확인한다	Find Account	송금할 대상 계좌를 찾는다
사용자가 입금한 금액을 제대로 보여주는지 확인한다	Check Amount	입금금액을 확인시켜준다
2가지의 경우에 맞게 출금한도가 나오는지 확인한다	Limited Amount	출금 송금 한도를 나타낸다
시간대별 수수료가 잘 책정되는지 확인한다	Count Commission	수수료를 계산한다
현실시간을 잘 반영하는지 확인한다	Check Business-hours	은행 별 영업시간을 나타낸다
모든 작업 종료 후 계좌 정보를 갱신했는지 확인한다	Update Account	은행 업무를 본 후 계좌의 정보를 갱신하다
명세서를 출력하는지 확인한다	Print statement	명세서를 출력한다
초기화면으로 돌아가는지 확인한다	Return To Main	메인 화면으로 돌아간다
해당은행 버튼이 잘 눌러지나 확인하고 그에 해당하는 은행에 연결되는지 확인한다	Select Bank	송금 대상 은행을 선택한다.

Activity 1010. Refine Plan

1. Project Scope

은행 직영점은 다양한 금융 업무를 맡을 수 있지만 다소 한정된 시간 동안만 서비스를 제공한다. 그리고 이러한 서비스 중 상당수가 출금 송금 입금 등 기본적인 것 들로 이루어져 있다. 이에 따라 각 은행마다 따로 있는 ATM기를 대신해 모든 은행이 공용으로 사용할 수 있는 ATM를 만들기로 목표하였으며 사용시간 또한 편의점 같은 24시간 영업지점에 기기 배치함으로써 시간의 제약을 받지 않도록 하는 것이 프로젝트의 목적이다.

2. Project Objectives

사용자는 시간에 구애를 받지 않고 원했을 때에 ATM기기에서 업무를 수행할 수 있다.

3. Functional Requirements

- Send Money
- Withdraw Money
- Deposit Money
- Check Remain Money
- Check Bank
- Select Bank
- Find Account
- Update Account
- Read Account Information
- Input password
- Check Password
- Count Commission
- Check Business Hours
- Limit Amount
- Check Amount
- Print Statement
- Return To Main

4. Performance Requirements

직관적인 UI와 빠른 작업 능력처리(1분 이내)를 갖고 있어야 한다.

5. Operating Environment

OS: Window7 & Window 10

6. Resources

-Human Efforts: 3man, 4month

-Human Resources: -3 programmers

-Duration: -4 month

-Budget

(단위: 만원)

	부분 계	비고
인건비	0	
활동비	150	
계	150	

7. Plan Scheduling

Stage	Phase(00X0)/Activity(001X)	Schedule(week)									
		1	2	3	4	5	6	7	8	9	10
1000. Plan & Elaboration	1001. Define Draft plan	■									
	1002. Create Preliminary Investigation Report	■									
	1003. Define Requirement	■									
	1004. Record Terms in Glossary	■									
	1005. Implement Prototype	■	■								
	1006. Define Business Use-Case		■								
	1007. 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 Case			■	■	■						
2032. Refine Use Case diagram			■	■	■						
2033. Define Domain Model			■	■	■						
2034. Refine Glossary			■	■	■						
2035. Define System Sequence Diagrams			■	■	■						
2036. Define Operation Contracts			■	■	■						
2038. Refine System Test Case			■	■	■						
2039. Analyze (2030) Traceability Analysis			■	■	■						
2040. Design			■	■	■						
2041. Design Real Use Case			■	■	■						
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 & Methods			■	■	■						
2052. Implement Windows			■	■	■						
2053. Implement Reports			■	■	■						
2054. Write Unit Test Code			■	■	■						
2060 Test			■	■	■						
2061 Unit Testing			■	■	■						
2062. Integration Testing			■	■	■						
2063. System Testing			■	■	■						
2064. Performance Testing			■	■	■						
2065. Acceptance Testing			■	■	■						
2066. Documentation Testing			■	■	■						
2067. Testing Traceability Analysis			■	■	■						