스마트 모빌리티 사용자 정보
통합 소프트웨어 프로젝트

산업정보시스템공학과 20172487 김현주
산업정보시스템공학과 20172568 홍인혁

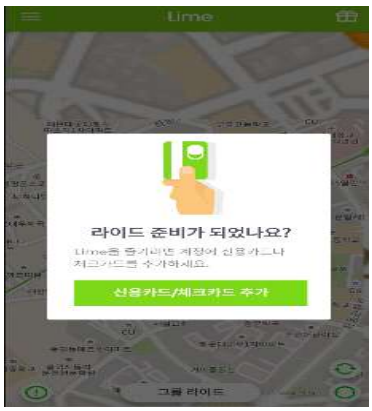
1. 개요

1.1 프로젝트 개요

• 배경

최근 스마트 기술의 발달은 교통수단에도 많은 변화를 일으키고 있다. 스마트 모빌리티는 전기로 움직이는 차세대 교통수단으로 최첨단 충전, 동력 기술이 융합된 소형 개인 이동 수단이 다. 보다 지능화되고 똑똑해진 교통 서비스를 말하기도 한다. 전기와 같은 친환경 연료를 사용하거나 1~2인승 개념의 소형 개인 이동 수단에 집중되어 있어 많은 대중들이 편리하게 애용하는 추세이다. 이런 추세에 발맞추어 간편하면서도 재미있게 즐길 수 있는 교통수단의 종류로 쏘카, 라임(Lime), 씽씽(xing xing) 등 다양한 모빌리티 쉐어링 서비스가 다수 등장하고 있다.

우리는 이러한 스마트 모빌리티를 이용하기 위해 고객의 면허정보와 결제를 진행하기 위한 카드 정보를 각각의 모빌리티 어플에 등록해야 사용이 가능하다. 기존의 스마트 모빌리티 소프트웨어들은 모두 같은 방식으로 결제를 위한 카드 정보와 면허 정보를 요구 함에도 불구하고 각각의 스마트 모빌리티 회사마다 결제하기 위한 카드 정보와 면허 정보 등록을 매번 요구한다. 이에 고객들은 번거로운 과정의 반복이 발생하여 스마트 모빌리티 이용에 불편함을 겪고 있다.



따라서, 본 프로젝트는 각각의 어플리케이션에서 공통으로 요구하는 정보 항목들을 통합하여 한 번의 입력을 통해 정보를 저장하여 소비자들에게 더욱 효율적이고 실용적이게 스마트 모빌리티의 편리한 이용이 가능하도록 제공하는 것을 목적으로 한다. 여기서 통합된 카드정보와 고객의 면허 정보는 각각의 회사와 상관없이 본 프로젝트의 개인정보 통합 어플리케이션으로 호환할 수 있도록 한다. 이를 구현하기 위해 다양한 스마트 모빌리티 회사에 흩어져있는 고객들의 면허 정보와 결제 진행을 위해 수집한 카드 등록 정보를 하나의 DB에 통합시켜 관리하도록 한다. 축적된 고객 정보를 바탕으로 정확한 고객DB를 구축하여 회원 개개인의 면허 정보와 결제를 위한 카드 정보를 수집하고, 이를 바탕으로 수집한 회원정보를 수준 높은 보안 시스템으로 관리한다.

• 사용자 요구 및 대책 <User requirement>
각 회사들은 공통으로 요구하는 카드정보와 면허정보와 같은 고객의 정보를 하나의 플랫폼에 구축하여, 이를 기반으로 고객에게 결제 및 사용이 가능하도록 서비스를 제공한다.
이에 따른 신규 소프트웨어의 최종 목표는 사용의 편리성에 중심을 둔 새로운 개인정보 통합 소프트웨어의 개발이다.

• 프로젝트 기간
2020년 5월 1일~ 2021 3월 31일

• 프로젝트 목적
- 고객들의 사용 편의를 제공하기 위한 통합 개인정보 시스템 개발
- DBMS 플랫폼 구축
- 제공기능: 통합된 정보 제공, 통합된 어플리케이션 신규 등록, 개인정보 조회 서비스, 공통된 요구 항목 입력을 위한 튜토리얼 기능
- 제약: 각 회사마다 수집된 고객 정보를 통합하고 관리할 시스템 및 인력 필요가 발생

1.2 프로젝트의 산출물
• 개발 단계별 주요 산출물

개발 단계	산출물

계획	프로젝트 수행 계획서
요구분석	사용자, 시스템 요구사항 명세서
설계	- 구조(시스템), 상세(DB) 설계 명세서 - 사용자 어플리케이션 알고리즘 설계서
구현	- DBMS 플랫폼 구축을 통한 프로그램 - 프로그램 소스 코드 - 개발자 시험 코드
테스트	테스트 보고서
출시	통합 시스템

1.3 정의, 약어

아웃소싱, CPM, DBMS, 고객정보시스템, Top down(하향식)

* DBMS: 데이터베이스(DB) 형태로 저장된 방대한 양의 각종 정보를 체계적으로 관리하는 기업용 소프트웨어.

* 고객 정보 시스템: 기업에서 고객의 이름, 주소, 가족 사항, 구매 사항 등의 정보를 데이터베이스화하여 전표 처리와 광고 우편 발행 등 사무 작업의 효율화나 고객 서비스의 향상 등을 목적으로 하는 정보 시스템.

* Top down: 프로그램의 규모를 예측하고 과거 경험을 바탕으로 예측한 규모에 대한 소요인력과 기간을 추정하는 프로젝트 비용 예측 방법.

• 타당성 분석

계약적 타당성, 법적 타당성, 기술적 타당성

- 계약적 타당성: 발주자와 외주업체와의 계약의 문서화를 통해 구체적이고 명확하게 기술하도록 한다. 이 때 구두 약속이 아닌 서면을 통해 계약을 최종 완료한다.
- 법적 타당성: 신용정보조회동의서를 작성함으로써 개인의 신용 정보를 정확한 절차를 거쳐 이용할 수 있으며, 신용 정보 조회 및 활용을 정확하게 진행할 수 있다. 따라서 본 프로젝트는 사용 도구들에 대한 정당한 법적 권한

을 갖는다.

- 기술적 타당성: 많은 고객의 데이터를 하나의 플랫폼으로 구축하는 기술이 필요하다. 이는 전문 IT업체에 외주를 맡겨 중앙 집중식의 시스템을 활용하여 분산된 데이터 및 정보 시스템을 연결하고 통합하여 데이터의 중복을 제거하고 양질의 데이터로 유지 및 관리가 수행될 수 있도록 한다.

2.1 자원

전체적인 자원과 규모를 예측하기 위해 Top Down (하향식) 방식을 이용한다. 이 방법을 통해 프로그램의 규모를 예측하고 과거 경험을 바탕으로 예측한 규모에 대한 소요 인력과 기간을 추청할 수 있다. 프로젝트를 개발할 때 필요한 소프트웨어의 LOC가 따로 주어지지 않았고 알 수 없기 때문에 기능점수특성을 고려하는 Function Point Attribute Estimation, COCOMO 방식을 이용한다. 이 때 추정되는 값은 모두 불확실성이 존재한다.

* Function Point Attribute Estimation

attribute	count	low	average	high	total
user input	6	5	6.5	8	39
user output	5	4	5	6	25
user inquiries	5	2	4.5	7	22.5
file	9	3	6	9	54
External Interfaces	3	5	7.5	10	22.5

가중치를 고려한 총 FP=163

프로그래밍 언어: java 사용 (53LOC/FP)

[1] CAV를 이용한 Line, Effort, Cost 계산

[1-1] Complexity is not considered(복잡성을 고려하지 않을 경우)
Suppose) java=53LOC/FP, Productivity=680LOC/MM, COST=2,000원/LOC

Total Lines = 163FP*53LOC/FP=8,639LOC
Efforts = 8,639LOC/680LOC/MM=12.70MM

Total COST=8,639LOC*2000원/LOC=17,278,000원

*복잡도 조정값(Complexity Adjustment Value)

영향없음	우연	적절	평균	중요	필수
0	1	2	3	4	5

1. 시스템이 신뢰성 있는 백업과 복구를 요구하는가?	4
2. 데이터 통신이 요구되는가?	3
3. 분산처리기능이 있는가?	3
4. 성능이 중요한가?	2
5. 현존하는 많은 운영환경에서 시스템이 수행되는가?	5
6. 시스템이 온라인 데이터 entry를 요구하는가?	3
7. 온라인 데이터 entry는 다중 스크린에서 혹은 다중 조작을 통해 구축된 입력 트랜잭션을 요구하는가?	1
8. 마스터 파일들이 온라인으로 갱신되는가?	3
9. 입력, 출력, 파일 또는 질의들이 복잡한가?	1
10. 내부처리가 복잡한가?	3
11. 코드는 재사용 가능하게 설계되었는가?	4
12. 변경(Conversion)과 설치(Installation)가 설계에 포함되었는가?	4
13. 시스템은 서로 다른 조작들에 다중 설치되도록 설계되었는가?	5
14. 어플리케이션은 사용자에 의해 쉽게 변경되고 사용될 수 있게 설계되었는가?	4

CAV total =45

[1-2] Complexity is considered(복잡성을 고려할 경우)

Suppose) java=53LOC/FP, Productivity=680LOC/MM, COST=2000원/LOC

따라서 680/53= 12.83(FP/MM)
Cost=2,000(원/LOC)*53(LOC/FP)=106,000(원/FP)

AFP= 163FP*[0.65+0.01*45] =179.3 FP
COST=163FP*106,000(원/FP)=17,278,000원
EFFORT= 163FP/12.83(FP/MM)=12.70MM
따라서 총 개발비용은 17,278,000원이고 , 총 투입노력은 12.70MM 이다.

[2]COCOMO를 활용한 PM, TDEV, N 계산

Total Lines=163FP*53LOC/FP=8,639LOC

예상 Lines 시스템 예상 규모가 8.639KLOC<50KLOC 이므로 organic mode 유형으로 계산한다.

Basic PM = $2.4*(8.639)^{1.05}$=23.09MM≈23MM

PM=23*(1.40)(0.94)(1.11)(1.15)(1.10)(1.07)(1.04)=47.29MM≈47MM

TDEV=$2.5*(47)^{0.38}$=10.79 Month≈11Month

FSP=MM/TDEV=47/11=4.27명≈약 4명

앞에서 구하였던 총 개발비용이 17,278,000원이기 때문에

Cost=2.6시간*(11*16)일*4명*9000원=16,632,000원

시급=9000원, 주 4회 근무 , 2.6시간으로 하였다.

비용 승수 요소	승수					
	매우 낮음	낮음	정상	높음	매우높음	극히 매우높음
요구되는 신뢰도	0.75	0.88	1.00	1.15	1.40	
데이터 베이스 크기		0.94	1.00	1.08	1.16	
제품의 복합도	0.70	0.85	1.00	1.15	1.30	1.65
실행 시간의 제약			1.00	1.11	1.30	1.66
주기억장치의 제약			1.00	1.06	1.21	1.56
S/W와 H/W의 조합의 안정성		0.87	1.00	1.15	1.30	
처리 시간		0.87	1.00	1.07	1.15	
분석가의 능력	1.46	1.19	1.00	0.86	0.71	
응용 경험	1.29	1.13	1.00	0.91	0.82	
컴퓨터와의 친속성	1.21	1.10	1.00	0.90		
프로그래머 능력	1.42	1.17	1.00	0.86	0.70	
프로그래밍 언어 경험	1.14	1.07	1.00	0.95		
소프트웨어 공학원리의 사용	1.24	1.10	1.00	0.91	0.82	
소프트웨어 도구의 사용	1.24	1.10	1.00	0.91	0.83	
요구되는 개발 일정	1.23	1.08	1.00	1.04	1.10	

2.2 일정

프로젝트의 진행은 계획-분석-설계-개발-테스트 순으로 진행되며 근무자들 근무 시간은 주4회이므로 4일을 1주일로 계산하여 PERT/CPM차트와 GANTT 차트를 작성한다.

[1] CPM을 이용한 일정수립

1-1) CPM 소작업 리스트

소작업 (activity)	추진내용	평균 일수
A	기획 목표, 범위 설정	4일
B	계획서 작성	6일
C	목표 제약 및 정의	7일
D	요구사항 분석	10일
E	현 시스템 분석	20일
F	시스템 구조 설계	30일
G	UI 수정 및 설계	25일
H	카드 정보 인식 시스템 개발	40일
I	현 프로세스와의 연동	25일
J	개발 S/W 의 호환성 테스트	15일
K	카드 번호 입력 시스템 개발	30일
L	카드 정보 입력 시스템 테스트	25일
M	카드 정보 인식 테스트	25일
N	전체 시스템 통합 테스트	20일
O	프로젝트 종료	0

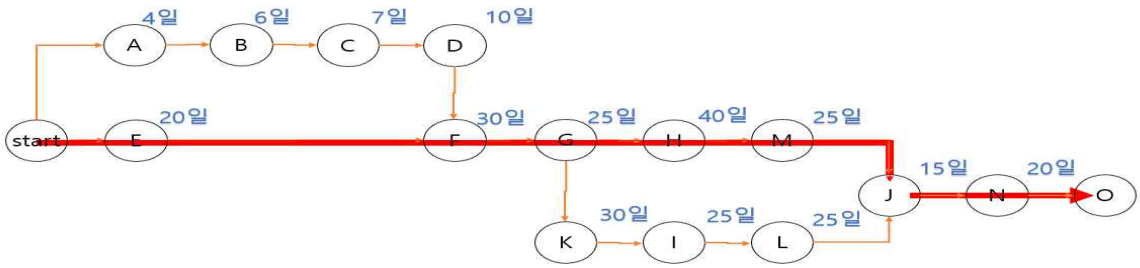
***간트 차트**

소작업	작업 내용	추진 일정										
		5월	6월	7월	8월	9월	10월	11월	12월	1월	2월	3월
A	기획 목표, 범위 설정	5/9										
B	계획서 작성	5/22										
C	목표 제약 및 정의		6/1									
D	요구사항 분석		6/12									
E	현 시스템 분석		6/5									
F	시스템 구조 설계			7/24								
G	UI 수정 및 설계					9/11						
H	카드 정보 인식 시스템 개발							11/20				
M	카드 정보 인식 테스트									01/01		
K	카드 번호 입력 시스템 개발						10/30					
I	현 프로세스와의 연동								12/18			
L	카드 정보 입력 시스템 테스트									1/29		
J	개발 S/W 의 호환성 테스트										02/12	
N	전체 시스템 통합 테스트											03/05

*CPM선행작업

소작업	선행작업	소요기간(일)
A	-	4일
B	A	6일
C	B	7일
D	C	10일
E	-	20일
F	E	30일
G	F	25일
H	G	40일
M	H	25일
K	G	30일
I	K	25일
L	I	25일
J	M,L	15일
N	J	20일

1-2) PERT/CPM 적용



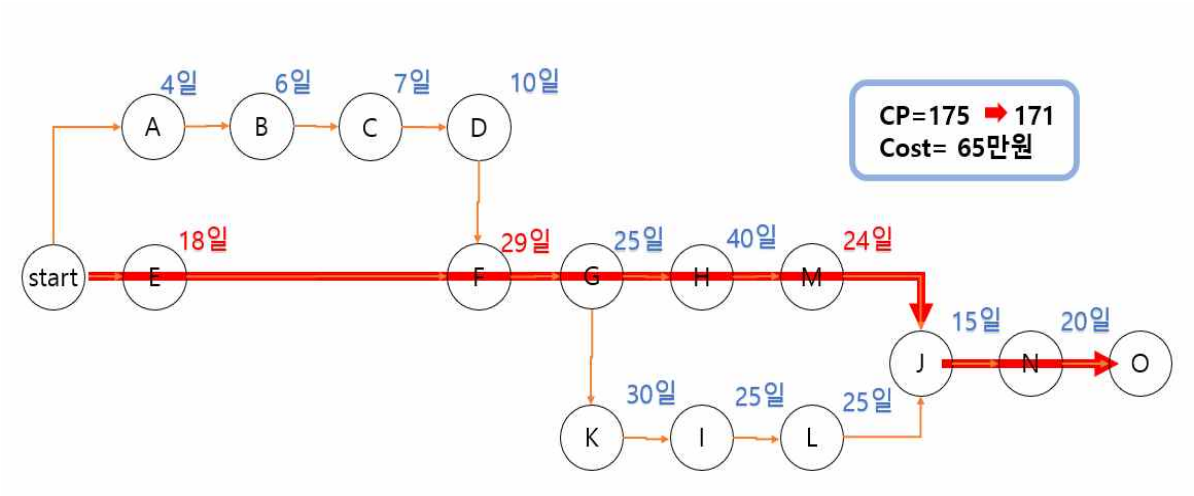
가능 경로	소 요 기 간 (일)
1. A-B-C-D-F-G-K-L-J-N	1.197
2. A-B-C-D-F-G-H-M-J-N	2.182
3. E-F-G-K-I-L-J-N	3.190
4. E-F-G-H-M-J-N	4.175

따라서 CP = 175(일)이다.

[2] Crashing

총 개발비용이 17,278,000원이고 개발자 Cost가 16,632,000이기 때문에 Crashing비용은 최대 646,000원으로 가정하고 계산한다.

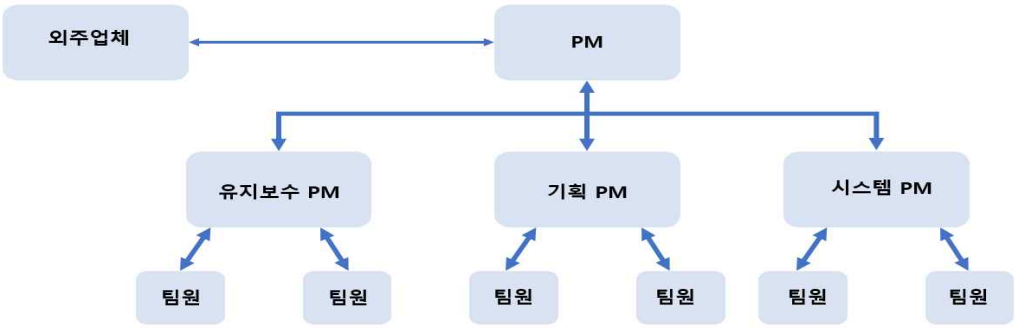
활동(activity)	예상 소요 일수	최소 소요 일수	1일 단축비용(만원)
A	4일	3일	10
B	6일	4일	10
C	7일	5일	10
D	10일	-	
E	20일	18일	15
F	30일	29일	15
G	25일	-	
H	40일	30일	25
I	25일	-	
J	15일	-	
K	30일	28일	25
L	25일	-	
M	25일	23일	20
N	20일	-	



최대비용이 대략 65만원이기 때문에 Cashing결과 최종 CP=171(일)이다.

3. 조직 구성 및 인력 배치

3.1 조직 구성



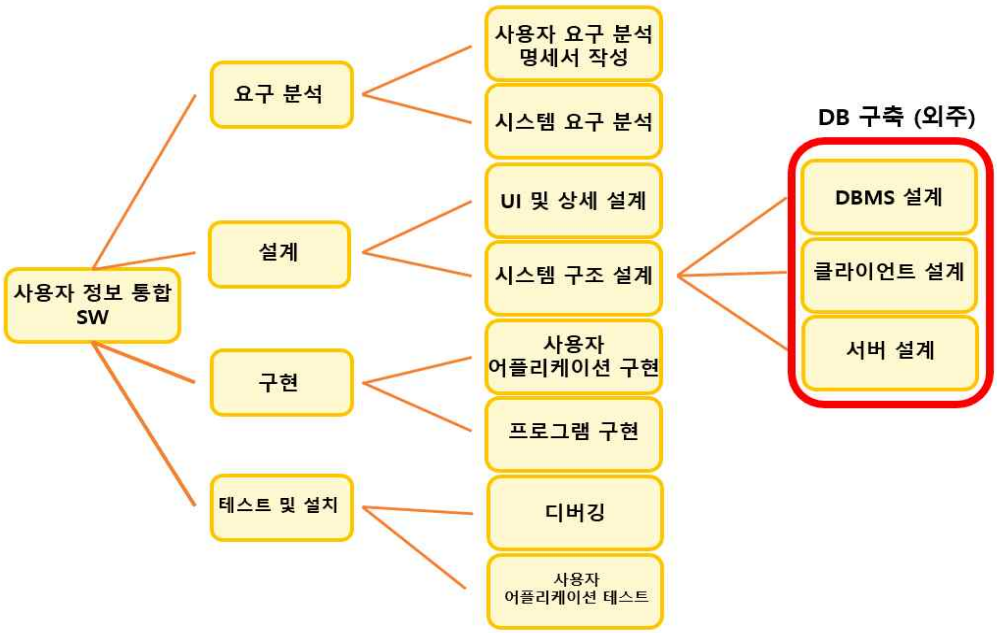
< 프로젝트 조직도 >

- 총괄 PM과 유지보수, 기획, 시스템 PM으로 혼합형 팀조직을 구성.

3.2 직무 기술

팀(조직)	직무기술
PM	• 프로젝트의 총괄 책임 • 산출물 관리, 품질 관리 지도 • 소프트웨어 개발 계획 수립 및 일정 관리
유지·보수	• 프로그램 지속적으로 유지 및 관리 보수 • 통합된 데이터베이스 관리
시스템	• 시스템 설계 및 구현 • UI 디자인
외주업체	• 시스템 테스트 • 통합된 고객 정보 DBMS 구축
기획	• 수집된 요구사항을 바탕으로 사용자 요구사항 명세서(User requirements) 작성 • 최종 산출물 디자인 고안

4. WBS



5. 기술관리 방법

5.1 변경 관리

개발 단계의 산출물들에 대하여 변경관리를 실시한다. 산출물의 변경이 동결된 후 일어나는 변경은 보고에 의하여 심사한 후 변경하고 통보한다. 변경관리에서 사용하는 변경 요청서는 별도로 정하며 변경 심사 및 관리 절차는 품질 담당자가 추후에 정한다.

• 고객 요구사항 변경: 기획팀에 변경사항을 요구사항명세서에 기입하여 수정하고 전체 선별 회의를 통해 요구사항 변경을 승인하거나 국지적인 변경 요청은 각 모듈 담당 파트별 선별회의를 개최한다. 조직 내 다른 팀들과 원활한 의사소통을 바탕으로 내용을 전달하도록 하고 요구사항 변경의 영향 분석을 진행해 승인 여부 결정 후 결정 사항을 변경관리대장에 기록한다.
• 예산의 변경: 변경되는 예산에 적합하도록 프로젝트 설계를 부분적으로 혹은 전체적으로 예산에 맞추어 변경한다.
• 일정의 변경: 일정을 계획에 맞게 변경, 조직원 모두가 변경된 일정에 대한 사항을 인식 하도록 한다.
• 운영체제의 변경: 변경된 운영 체제에 맞도록 개발 및 재설계를 한다.

5.2 위험 관리(Risk Control)

● **위험 확인 (Risk 식별)**

Risk Type	Risk
People	- 프로젝트 완료 전 숙련된 직원이 이탈할 경우.
Requirement	- 고객의 요구사항 변경이 자주 발생할 경우.
Organisational	- 조직의 재정적인 문제로 인한 프로젝트 예산 축소가 발생할 경우.
People	- 핵심 직원의 병가할 경우.
Technology	- 잘못된 인터페이스 개발.
Estimation	- 개발기간을 짧게 추정한 경우.

● 위험 분석

Risk	Probability	Effects
- 프로젝트 완료 전 숙련된 직원이 이탈할 경우.	Low	Tolerable
- 고객의 요구사항 변경이 자주 발생할 경우.	Tolerable	Serious
- 조직의 재정적인 문제로 인한 프로젝트 예산 축소가 발생할 경우.	Moderate	Serious
- 핵심 직원의 병가할 경우.	Low	Tolerable
- 잘못된 인터페이스 개발.	Serious	Catastrophic
- 개발기간을 짧게 추정한 경우.	Serious	Catastrophic

● 위험 관리

Risk	Startegy
- 프로젝트 완료 전 숙련된 직원이 이탈할 경우.	작업들이 일부 겹치도록 구성하여 다른 직원들이 그 빈자리를 채울 수 있도록 한다
- 고객의 요구사항 변경이 자주 발생할 경우.	요구사항 변동에 의한 비용발생과 영향을 고객이 알 수 있도록 하여 요구변동을 최소화한다.
- 조직의 재정적인 문제로 인한 프로젝트 예산 축소가 발생할 경우.	조직의 고위직에게 이 프로젝트의 달성 여부가 비즈니스에 끼치는 영향을 설명한다.
- 핵심 직원의 병가할 경우.	팀을 재조직하여 다른 책임자가 프로젝트를 임시로 담당 가능하도록 한다.
- 잘못된 인터페이스 개발.	철저한 검토 회의를 통해서 발주 정당이 요구하는 바를 정확히 충족시킬 수 있도록 한다.
- 개발기간을 짧게 추정한 경우.	비용을 더 들이더라도 일정 계획에 맞출 수 있도록 하거나 발주자의 양해를 구하여 기간을 조정한다.

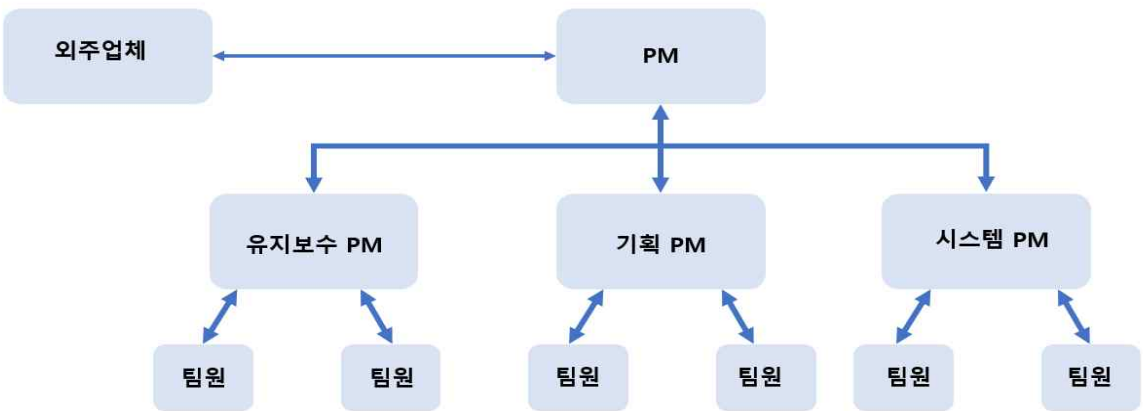
5.3 비용 및 진도관리

• 비용

* 추가적인 비용 발생의 경우 : 발주 정당에서 초기 선정 지불 비용과 별개로 지원을 받는다.

* 계속적인 요구사항의 변화의 경우 : 엄격한 선별 과정을 통해 요구 사항을 반영하여 추가 비용을 최소화 시킨다.

• 진도



불필요한 문서들은 최소화하는 방향으로 작업을 진행.
- 진도의 흐름은 Top-Down 방식
- 진도 관리 흐름은 팀원들과 각 책임PM간의 교류와 의사결정을 바탕으로 최종 산출물을 총괄 PM에게 전달하는 흐름방식이다.

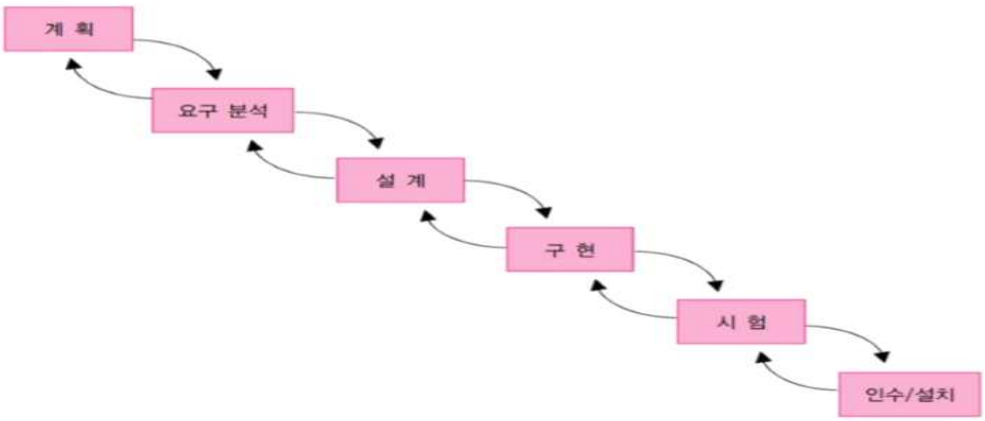
5.4 문제점 해결 방안

프로젝트 진행 과정에서 발생 가능한 위험요소들을 식별하고 발생 가능성이 높은 위험요소들을 주의하여 선별한다. 모든 위험요소들의 제거는 어렵기 때문에 위험요소들의 영향을 최소화 시킬 수 있는 위험 최소화 전략, Risk 관리 전략을 세워 위험에 대처한다.

6. 표준 및 개발 절차

6.1 개발 방법론

• 폭포수 개발 모형



프로세스가 단순하여 초보자가 쉽게 적용 가능하고 중간 산출물을 명확히 확인하며 관리가 가능하다.
각 단계 수행 전에 충분한 연구와 분석이 가능해 적절한 피드백을 진행하며 개발이 수행될 수 있도록 폭포수 모형을 사용한다.

7. 검토 회의

7.1 검토회 일정

검토회의	일시	참여자
계획	계획서 검토 시작하는 첫날	프로젝트 매니지들과 사용자
요구분석	요구명세서 작성 상세	프로젝트 매니저들과

	면담 마지막 날	사용자
설계	상세 설계 시작하는 첫날	프로젝트 매니저들과 시스템, 기획 팀원들
개발	프로그램 실행 첫날	프로젝트 매니저들과 시스템, 기획 팀원들
테스트	유지 및 보수작업 시작날	프로젝트 매니저들과 총 팀원들

7.2 검토회 진행 방법

• 검토회는 계획-요구분석-설계-개발-테스트-종료 단계에 따라 진행된다.
• 요구 분석 검토 회의 과정에선 프로젝트 매니저들은 시스템 사용자의 요구를 중심으로 검토회를 진행한다.
• 모든 검토회 과정의 회의록을 작성한다.
• 각 팀의 PM 주관 아래 각 팀원들이 참여하여 최종 산출물의 일정에 맞추어 개발 사항을 체크하고 확인 및 점검하여 설계나 기술적 부분의 문제를 보완한다.

7.3 검토회 후속 조치

작성된 회의록을 프로젝트 매니저들과 팀원들 전원이 반드시 확인하도록 한다. 검토회의 종료 후 발견된 오류 리스트를 작성한다. 그리고 지속적으로 진행될 수 있는 해결방안을 구상하여 각 부서에 공유하고 후속 조치를 실시한다. 적절한 과업 분배 과정을 바탕으로 오류들의 해결방안이 프로젝트에 적절히 반영 될 수 있도록 진행한다.

8. 개발 환경

서버 서비스: 아파치 서버, 웹서비스지원: HTML, PHP4
DB: MySQL DB관리자: SAP, 개발언어: Java언어
OS: Window10, Android OS , IOS
* 성공적인 프로젝트 수행을 위해선 최적의 개발환경이 필요하다. 최적의 개발환경은 개발비용과 기간 증가를 방지하는데 도움을 줄 것이다.
• People
개인의 능력과 특성을 잘 고려하여 팀을 선택하고 조직을 구성한다.
팀원들 내 원활한 의사소통을 바탕으로 프로젝트를 진행한다.
• Process

위험요소를 파악하고 품질을 높이기 위해 노력한다.
• Product
산출물을 가시적을 확인 가능하도록 하고 기능적 비기능적 요구들을 반영한다.
• Technology
적절한 지원도구 및 개발언어를 선택하여 사용한다.

9. 성능 시험 방법

• Validation(유효성) - 요구사항에 맞는 프로그램인지 확인
• Verification(검증) - 개발 방법론 평가가 제대로 진행되었는지 확인
• 화이트 박스 시험(White box test) - 프로그램 내부 구조의 타당성 여부를 시험, 처리 루틴을 검증하기 위한 시험 데이터를 작성하여 시험을 실시한다.

10. 문서화

각 시스템별 요구사항 분석 및 해결방법을 제시하고 자료 사전 및 소단위로 명세서를 작성해 유지보수를 위한 코드의 문서화를 실시한다.
각 프로세스를 단계별로 구분하여 이후 프로젝트 진행하는 데 참고 자료로도 활용 가능하도록 문서화한다.
- 계획단계 : 프로젝트 계획서, 기획 목표서 등
- 분석 단계 : 예비 예산안 작성, DFD 등
- 설계 단계 : 시스템 구조 설계서, UI 설계, 프로그램 설계 등
- 개발 단계 : 코드 개발 등
- 테스트, 출시 단계: 사용자 만족도 평가서 등

11. 유지보수

• 품질 분야는 일정 기간을 간격으로 문제점을 지속적으로 점검하여 관리한다.
• 프로그램 오류가 발생한 경우에는 프로그램 수정을 통해 오류가 해결되도록 실행한다.
• 별도의 유지보수 팀원들을 조직하여 오류사항에 대해 문서화를 실행한다.
• 일정 기간을 간격으로 사용자의 요구사항을 확인한다.

12. 설치, 인수

• 프로젝트 총괄 매니저가 최종 검토 진행 후 설치를 실행한다.
• 사이클 반복 과정을 거칠 때마다 최적의 프로젝트 산출물의 결과를 인수할 수 있도록 한다.

• 각 단계별 산출물을 상호 공유한다.
• 프로그램 사용자에게 프로그램 사용 및 주의 사항에 대한 튜토리얼을 교육한다.

13. 참고문헌 및 부록

한국정보통신기술협회 (www.ttta.or.kr), 국제표준화기구(www.iso.org), Code Complete, 정보문화사(Steve Mcconnell, 서우석 역, 2005), 소프트웨어 공학론, 정익사(최은만 저,1995), 네이버 지식백과 IT 용어사전, 전기전자기술자협회