# 제 1회 SW Architect 성과 공유회 -
# Situation-based Contents Share/Notification Service

1. Overview
2. Requirement Specifications
3. Elements & Roles
4. Architecture Description (Lv. 01)
5. Architecture Description (Lv. 02)
6. Service Scenario for Validating Architecture
7. Lessons Learned & Future Works / 사업적 기여 및 성과 항목

Appendix.

2015

CTO부문 Software공학연구소

Agile개발팀 정민규 수석

Section 01.

# Overview

- 1.1. 상황기반 Content 공유/알림 서비스 개요
- 1.2. 사업동향 및 경쟁사 현황 분석
- 1.3. 서비스 방향성 분석

# 상황기반 Content(메모로 Prototyping : Memo.Rise) 공유 및 알림 서비스


**Write**
기억해야 할 메모를
작성한다

**Write with Situation**
날짜,장소,날씨와 함께
기억해야 할 메모를
작성한다

**Send**
친구에게 메모를 보내거나
함께 공유한다

**Receive**
친구가 보낸 메모를 받는다

**Remind**
특정 시간, 장소, 날씨가
도래하면
해당 메모를 알림 받는다

# 1.2. 사업동향 및 경쟁사 현황 분석

| | 미리알림 | Google Keep | 솜노트 | 어썸노트 | Memo.Rise |
|---|---|---|---|---|---|
| 컨셉 | <br>할 일을 등록하고 날짜/위치 기반으로 알림을 주는 To-do앱 |  <br>할일/메모를 등록하고 날짜/위치 기반으로 알림을 주는 메모 앱 |  <br>클라우드 기반의 긴 메모와 문서파일 첨부가 가능한 노트 앱 |  <br>할일, 일정/스케줄 관리에 용이한 통합형 다이어리 |  <br>날짜/위치/날씨 기반으로 알림을 주고 친구와 공유/협업하는 메모 앱 |
| 특징 | • iOS이 기본 앱(연락처, 전화)과 연동하여 다양하게 사용할 수 있음 | • 구글 음성인식 기술을 바탕으로 음성메모를 텍스트로 변환해서 저장<br>• 구글 드라이브와 동기화, Google Now를 통해 알림을 받을 수 있음 | • 카카오톡/facebook/메일로 공유<br>• 다양한 문서 파일 첨부 및 뷰어 기능 | • SMS/메일/facebook이벤트 등록으로 공유<br>• 앱 사용자 간 블루투스를 이용해 전송 가능 | • 날씨 상황 입력을 통해 감성적인 상황메모 작성 및 공유 가능<br>• 메모뿐만 아니라 다양한 콘텐츠(음성,사진,파일)을 공유하는 SNS 플랫폼으로 확장 가능 |
| 제약사항 | • 위치 알림 설정 시, 한국 지도에는 특화되지 않아 위치 검색/설정의 사용성이 떨어짐<br>• 친구와 공유하기 위해서는 icloud 웹으로 접속하여 친구의 이메일로 전송해야 하는 번거로움 | • 날짜,위치알림 2개 중 하나만 선택하여 알림 받을 수 있음<br>• 카테고리 분류가 없음 | • 저장 공간이 한정적임(100MB), 저장 공간을 추가 하기 위해서 유료 회원의 전환이 필요함<br>• 알림 기능 없음 | • PC버전은 제공하지 않음<br>• 동기화 기능의 버그가 많음 | • 현재 모바일 앱 서비스만 지원하지만 PC버전 및 동기화 기술 고려 필요 |

# 1.3. 서비스 방향성 분석

적절한 상황에 자신의 기억을 돕거나 타인에게 말을 전달하기 위해

**Situation(time, place, weather…) 정보**와 **Communication Channel**을 이용하여

손쉽게 짧은 글을 남기고 공유하는 서비스

## Collaboration

캠핑 떠나는데
각자 준비할 수 있는
것들 체크해볼까?

## Caring

카레 만들어서 냉장고
에 넣어놨으니 집에
오면 데워먹어

## Mission

집에 먼저 도착하는
사람이 택배를
받아놨으면 좋겠어

## Serendipity

헉! 여기는 부모님과
내가 2살 때
여행한 장소?

Section 02.

# Requirement Specification

- 2.1. Feature Identification
- 2.2. Functional Requirements
- 2.3. Non-Functional Requirements

- 01. 사용자 생명주기 관리 및 관련 정보 확장 지원
  - 사용자 계정 및 접근 권한 관리 (등록)
  - 사용자 로그인/아웃
  - 사용자 추가정보 입력 및 친구 관리 지원
- 02. Multi-context 기반 Content 작성
  - Time, Location, Weather 등 유연한 context 지원
  - Simple한 Content 작성
- 03. Content 수신/보기
  - Event-Hooking 지원
- - Compact한 Content 수신 알림
- 04. Uniform화 된 View Interface 지원

- **Func_Req01. 사용자 계정 및 접근권한 관리**
  - 사용자 계정 등록
    - Cloud API를 통한 사용자 Request 기반의 등록
  - 사용자 계정 관련 정보 및 접근권한 생성
    - 이벤트 기반의 ID, 패스워드, 전화번호, 접근권한 등 계정 관련 정보 생성 처리
- **Func_Req02. 사용자 로그인/아웃**
  - Cloud API를 통한 사용자 로그인/아웃 처리
  - 로그인 상태 유지 지원
- **Func_Req03. 사용자 추가정보 입력 및 친구 관리**
  - 사진, 나이, 성별 등 추가정보 입력 지원
  - 친구 리스트 및 추가/삭제 지원

- **Func_Req04. 유연한 Context 지원 구조**
  - 시간/위치/날씨 등 다양한 Context 설정 지원
  - Context 설정 인식 및 모니터링 기능
  - 추가 Context 지원 가능
- **Func_Req05. Simple한 Content 작성**
  - Content 작성창의 접근용이성 지원
  - 추가 Content 확장 지원

- **Func_Req06. Event Hooking 지원**
  - Content에 대한 Event Hooking으로 알림 지원
- **Func_Req07. Compact한 Content 수신 알림**
  - 최소의 정보로 Content 수신 알림
  - 알림에서 Content 정보로 연동
- **Func_Req08. Uniform화된 View Interface 제공**
  - 클라이언트SW의 View와 서비스SW의 Model 분리
  - Uniform Interface 제공

- **품질속성 (Quality Attributes)**
  - Platform 설계 시 고려하여야 할 공통 품질 속성

| 품질 속성 | 설명 |
|---|---|
| Reusability | ■ 플랫폼은 여러 시스템에서 활용하도록 재사용 가능한 기능/비기능을 커버할 수 있도록 설계/개발되어야 함<br>■ 라이브러리 (공용화된 인터페이스)를 통하여 플랫폼 기능을 재사용 가능하여야 함 |
| Extendibility | ■ 플랫폼은 공통적인 기능/비기능은 제공하고, 가변적이거나 특정 시스템에 특화된 기능/비기능은 실행될 수 있도록 준비되어 있어야 함<br>■ Extension Point를 감안하고, 추가될 기능/비기능성을 위하여 유연한 구조이어야 함 |
| Standardization | ■ 재사용성과 관련있는 품질 속성<br>■ 재사용성을 보장하기 위해서는, 플랫폼 내에 컴포넌트 간의 통신, 플랫폼과 외부 시스템과의 통신의 표준화가 보장되어야 함<br>■ 이를 지원하기 위하여 표준화된 인터페이스 설계가 필요함 |
| Performance | ■ 플랫폼은 기본적으로 일부 성능의 저하를 가져올 수 있음<br>그러므로 이를 최소화할 수 있도록 설계되어 있어야 함 |

- **품질속성 (Quality Attributes)**
  - 상황 기반 Content 공유/알림 서비스 설계 시 고려하여야 할 품질 속성

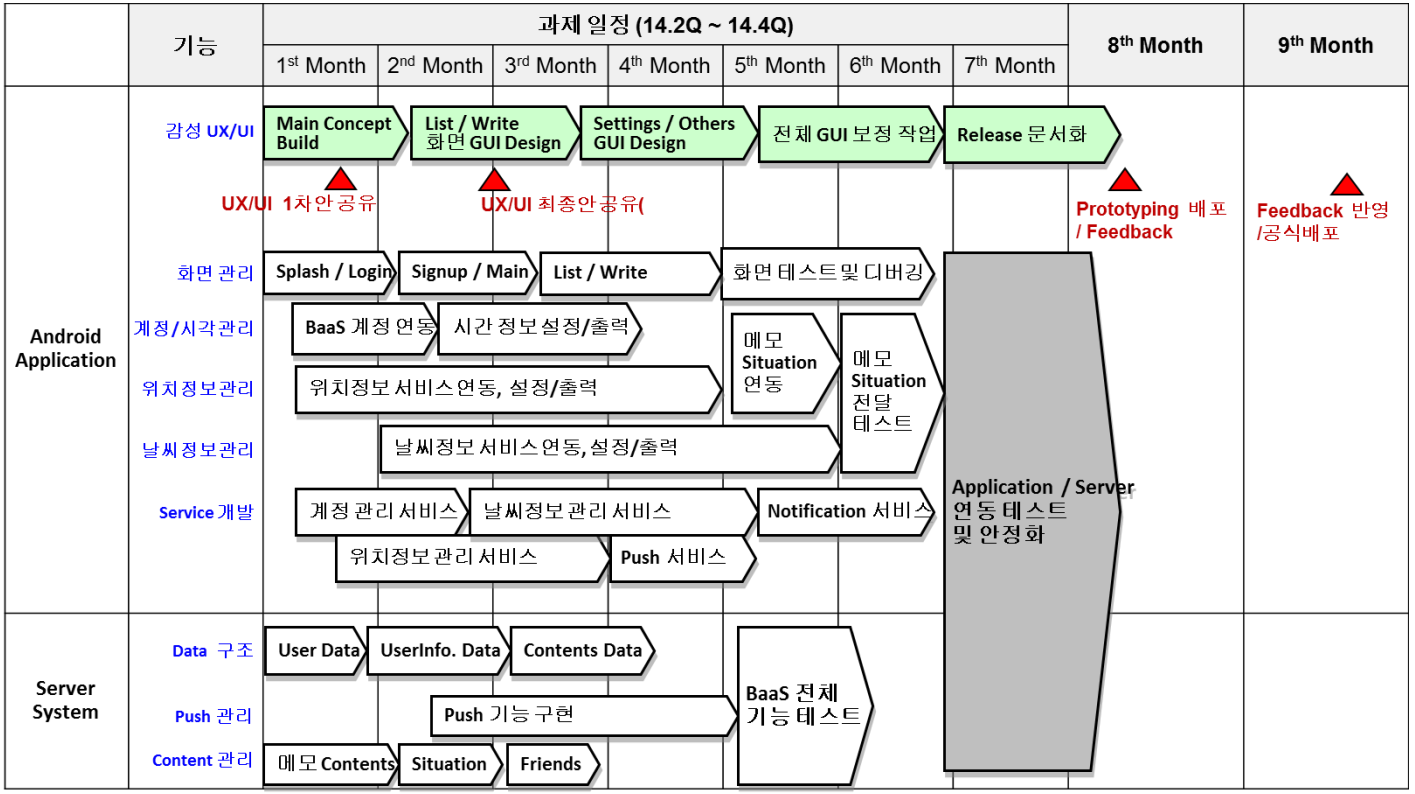
| 품질 속성 | 설명 |
|---|---|
| Flexibility | ■ 다양한 상황 정보 및 Content를 지원할 수 있어야 함<br>■ 즉, 유연한 형태의 서비스 연동 구조 및 어플리케이션 개발이 가능하여야 함<br>■ 이를 위해서는 기존 구조의 확장을 커버할 수 있어야 함 |
| Security | ■ 접근 권한을 제한하고 사용자 Content에 대한 보호 관리가 필요함<br>■ 데이터의 무결성을 보장해야 함 |
| Reusability | ■ View와 Model을 분리할 수 있도록 함<br>■ 이 경우 분리된 모듈 간 인터페이스를 고려하여야 함 |
| Operability | ■ 다양한 형태의 입력에 따라 필요한 기능성을 수행할 수 있어야 함<br>■ '다양한 형태의 입력'이라는 조건의 제약을 두어야 할 수도 있음 |
| Portability | ■ 하나 이상의 대상 단말을 지원하여야 함<br>■ 이 경우 대상 단말을 같은 OS에 다른 하드웨어 스펙, 다른 OS에 같은 하드웨어 스펙, 다른 OS에 다른 하드웨어 스펙 등 여러 가지 OS 및 하드웨어 플랫폼을 고려하여야 함 |

- **Design Constraints**
  - 다양한 상황정보 및 Content를 지원할 수 있는 구조로 설계하여야 함
    - 후보 상황정보 : 시간, 위치, 날씨 등
    - 후보 Content : 메모, 사진 등
  - 기존 기능을 Enhance할 수 있는 Extension을 위한 인터페이스를 제공하여야 함
    - 공통부/가변부 정의 후 대응
  - 사용자 정보가 관리될 수 있는 데이터베이스 스키마를 설계하여야 함
    - 사용자 계정정보와 사용자 추가입력 정보, 사진파일 등 사용자 관련 정보가 연동되어야 함

- **비즈니스 Constraints**
  - Schedule

    본 프로젝트는 진행 중에 참여 인원이었던 LG CNS 인원이 복귀하고, LG전자 인원 중 아키텍트 본인(정민규 수석연구원)을 제외한 인원들의 지원이 끊기게 되어 리소스에 개발 일정이 의존성을 가지게 됨

  -



- **비즈니스 Constraints**
  - Project lifetime

    상황 기반 Content 공유/알림 서비스는 다양한 Content를 고려하고 있다. 따라서, 프로젝트가 끝난 이후에도 SW 아키텍처 설계는 보완되고 Refactoring 되어 플랫폼으로 발전하고자 함
  - Workforce

    SW아키텍트 : 정민규 수석

    아이디어 기획/제안, UX 초안 : 정민규 수석, ~~안양재 과장(CNS)~~

    구현 : 정민규 수석, ~~안양재 과장,~~ 이용환 주임, ~~이상윤 과장(CNS)~~

    UX / GUI 지원 : ~~최수련 대리(CNS)~~

- **기술 Constraints**
  - 개발환경

| ID | Name | Description |
|---|---|---|
| TC-01 | Platform | Use Android Smartphone (Optimus LTE2 또는 G2모델)<br>Use 3G/4G/Wi-Fi<br>Use Cloud Service (e.g. BaaS) |
| TC-02 | 클라이언트SW | Use Android java language (Embedded SW)<br>Use GPS / Phone DB, Control<br>Use Cloud Service SDK<br>Android Application Type |
| TC-03 | 서비스SW | Use Android java language<br>Android Service Type |
| TC-04 | 클라우드SW | Use java / java script language<br>Kinvey (BaaS Cloud Service) & SDK |

  - 운영환경

| ID | Description | Difficulty |
|---|---|---|
| TC-05 | Content 전송은 클라우드 서비스를 기반으로 이루어진다. | High |
| TC-06 | 사용자에 대한 인증이 이루어져야 한다. | Medium |
| TC-07 | 사용자는 사용자 설정을 미리 해야 한다. | Medium |
| TC-08 | 사용자 스마트폰에서 상황정보를 이용하기 위해 백그라운드로 돌고 있어야 한다. | Low |
| TC-09 | 상황정보들을 얻기 위해서 다양한 서비스들을 활용할 수 있다 | High |
| TC-10 | Content 종류가 다양하므로 그에 따른 설정들도 달라져야 한다. | High |
| TC-11 | 사용자가 친구들을 주소록을 통해 찾기가 가능해야 한다. | Medium |
| TC-12 | 기능 동작 중에도 스마트폰 기본 기능들은 정상적으로 동작해야 한다. | Low |
| TC-13 | 클라우드 서비스에서 로깅이 이루어질 수 있다. | High |

- **Project Constraints**

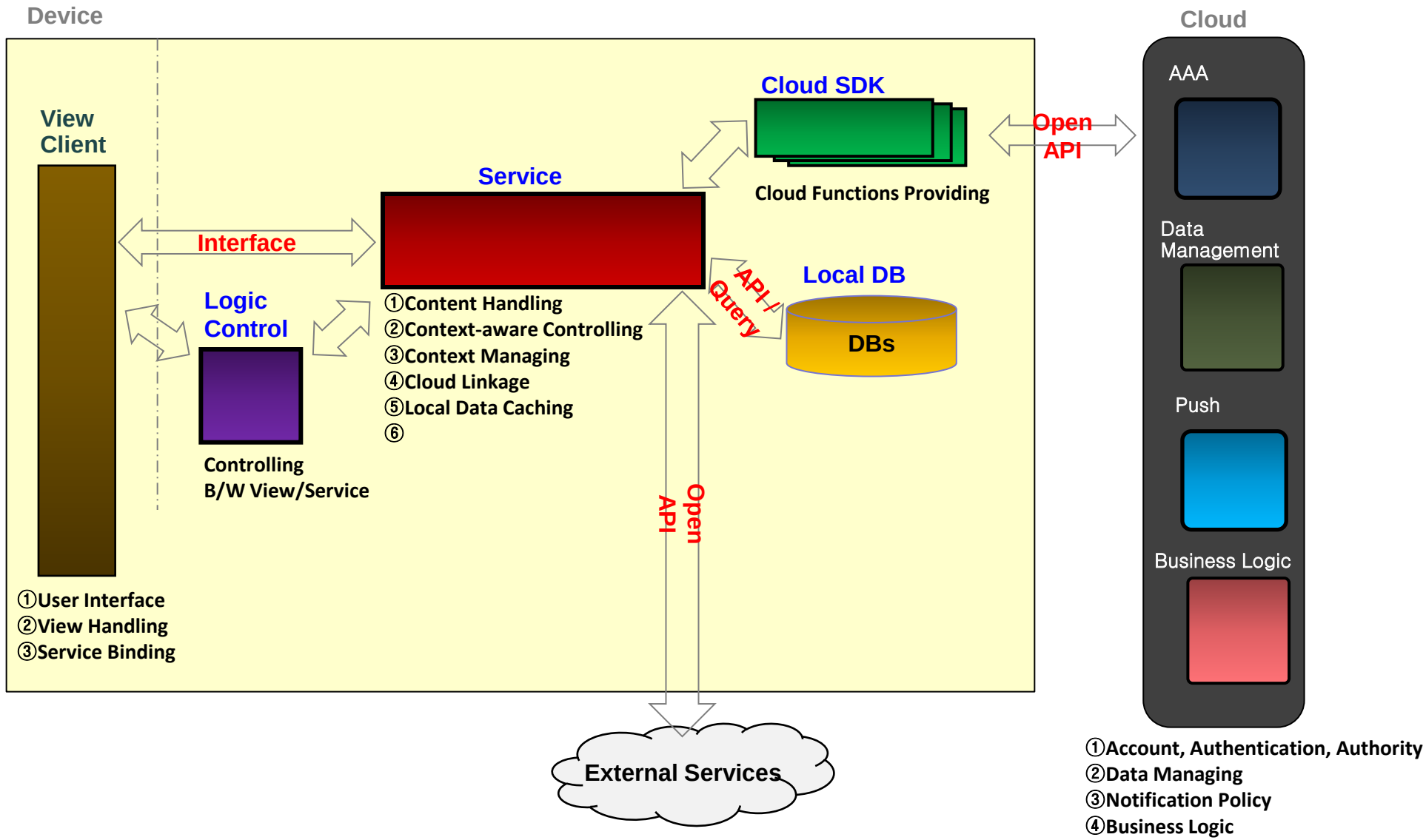
| 유형 | 설명 |
|---|---|
| 고려사항 | Proto버전의 Application + 서비스 지원 Server + 기술 Practices<br>Exchangeability, Extensibility, Independency |
| 개발형태 | 상황 기반 Content 서비스를 위한 Prototyping 기획 및 설계, 구현<br>(Android Application + Server) |
| 요구사항 명확도 | 기능적 요구사항 불명확<br>비기능적 요구사항 불명확 |
| 도입 기술 | Android Framework, BaaS(Backend as a Service), Push 등 |
| 기술적 난이도 | 다소 높음 (새로운 Concept 및 기능 Prototyping) |
| 개발방법론 / 도구 | OOAD / OOP, UML, Business Modeling |
| 핵심 이슈 | 여러 상황정보를 담은 Content 서비스 아키텍처 |
| 리스크 | 인원/Resource 빈약<br>LG전자 조직 내 지원 無<br>BaaS 활용<br>구조 신뢰성 확보 |
| 아키텍트 역할 | Model과 View가 분리된 구조 설계<br>상황정보 기반의 Content 제공 구조 정의<br>핵심 클래스/데이터 모델 정의<br>스케쥴/인원/이슈 Trade-off 정리<br>설계/구현 이슈 해결 등 |

Section 03.

# Elements and Roles

- 3.1. Relationship among elements
- 3.2. Descriptions on Key Elements and their roles
- 3.3. 상황 기반 Content 공유/알림 서비스 & its Context
- 3.4. Data Modeling

# 3.1. Relationship among elements

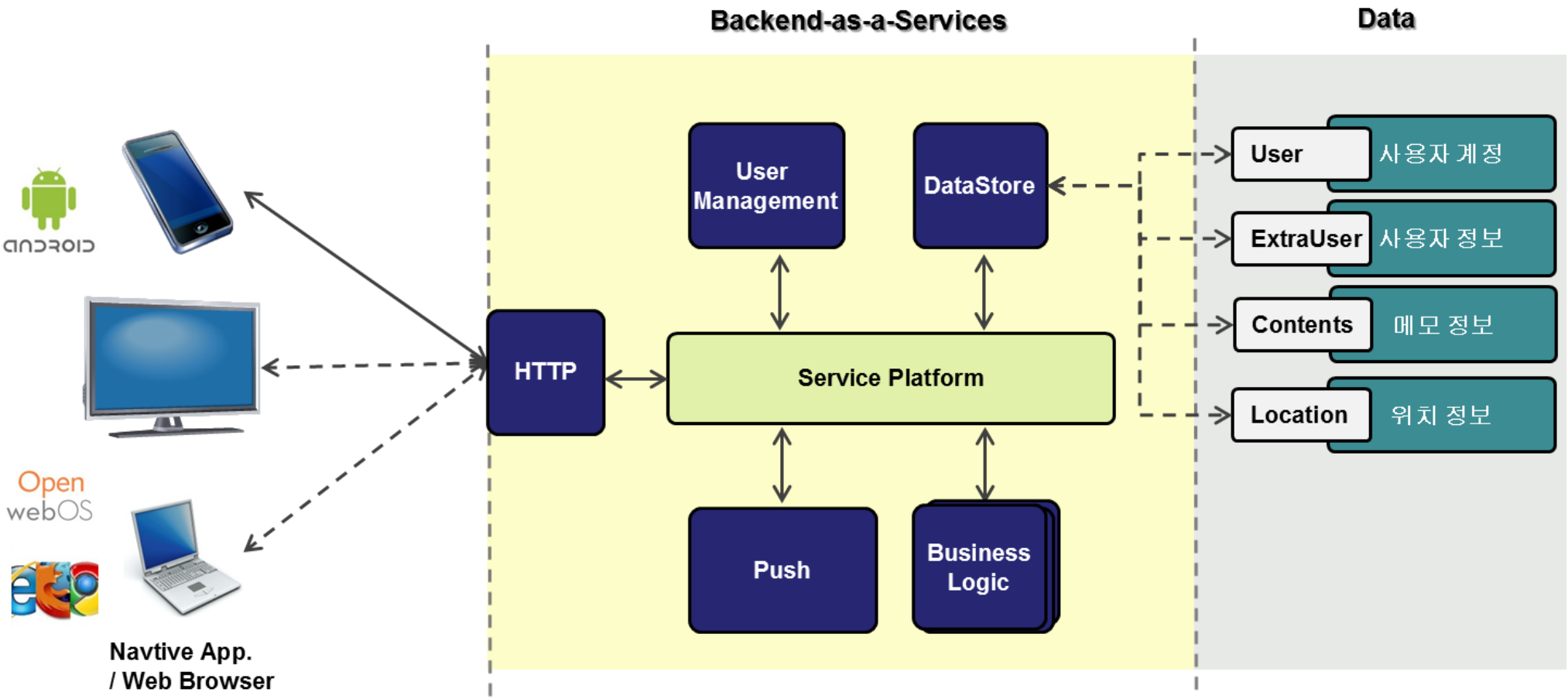


- **View Client Component**
  - 사용자 인터페이스를 담당하는 컴포넌트
  - 사용자의 입력을 받아 Service Component로 전달하고, Service Component로부터 처리결과와 이벤트들을 수신하여 여러 View로 사용자에게 표현한다.
    - 사용자의 Client의 예: Web Browser, 모바일 디바이스에 설치되어 있는 App, Smart TV 등에 설치되어 있는 App 등
  - 입력 데이터의 유형
    - 상황정보 (시간, 위치, 날씨 등)
    - Content (메모, 사진 등)
    - 친구 정보
  - 주요 기능 목록
    - User Interaction Handling (e.g.  User, Content, Context)
    - View Handling (e.g. Main, Subs)
    - Service Binding (e.g. Data Sending/Event Notification)

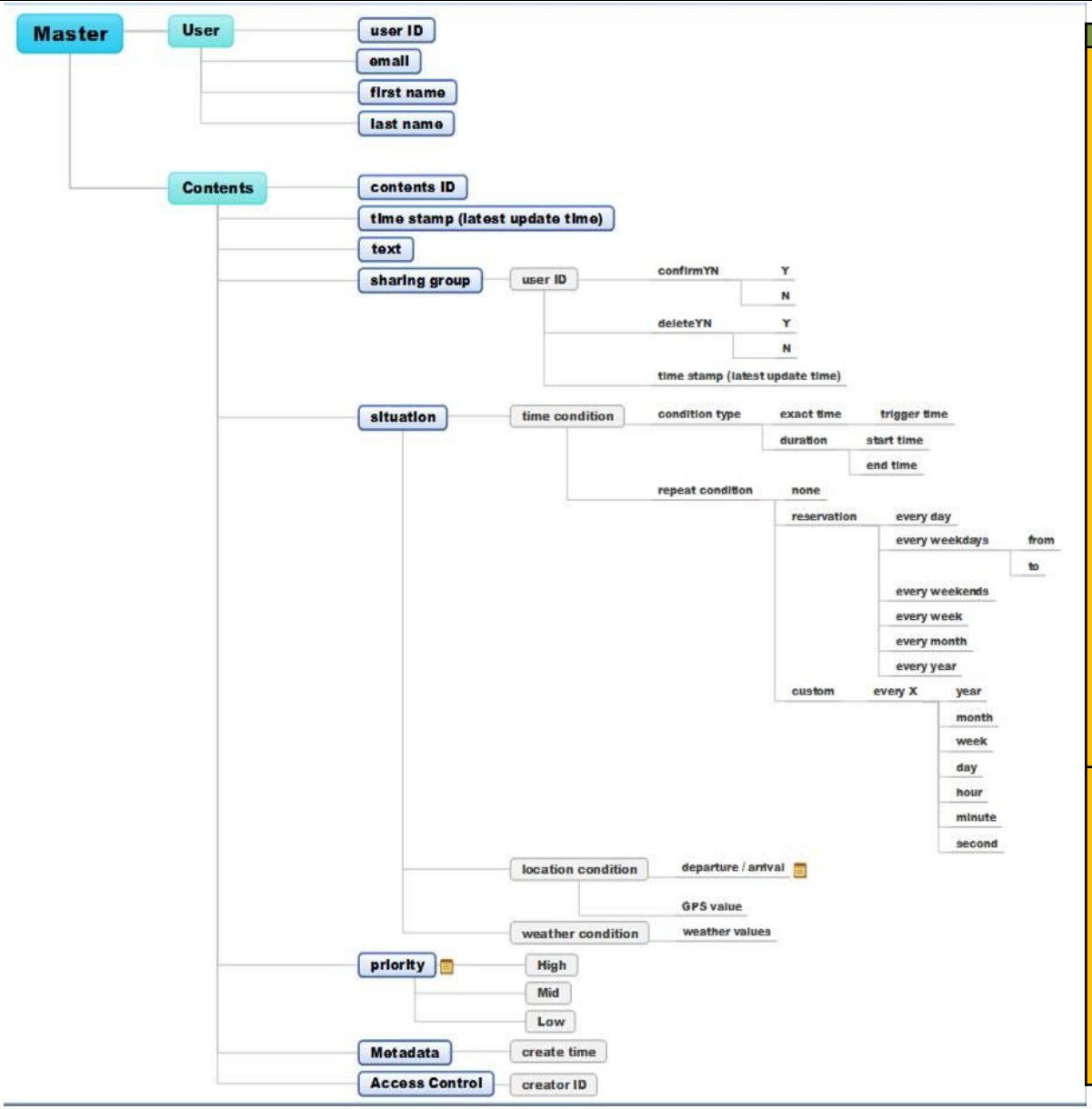
- **Service Component**
  - 데이터 관리와 클라우드와의 연동을 담당하는 컴포넌트로써, 주요 Feature들을 지원하기 위한 기능성을 제공함
    - Context Handling, Aware Controlling, Managing
      - 실제 상황정보 데이터의 설정/관리/인식 등을 담당함
    - 입력받은 이벤트에 따라 서비스/클라우드 연동을 실행하는 기능
      - 사용자 관리, Content 전송/수신 등을 위해 클라우드와 연동하여 결과를 수신함
    - 단말에서 관리하는 저장소를 관리하기 위한 기능
      - 저장소: Local DB, Preference 등

- **Cloud**
  - 사용자 관리, 접근권한, 데이터 저장, Notification, 비즈니스 로직 등의 기능을 가지는 BaaS 솔루션
  - 다음과 같은 기능성을 제공함
    - 사용자 계정, 인증, 권한 등을 관리함
    - 데이터 저장 및 관리
    - Google Push 기능 연동
    - 다양한 시나리오 적용을 위한 Business Logic

# 3.3. 상황 기반 Content 공유/알림 서비스 and its Context



# 3.4. Data Modeling




| 구분 | 테이블 | 컬럼 | 비고 |
|---|---|---|---|
| Server | Contents | id | |
| | | text | |
| | | situation | time situation/location situation/weather situation |
| | | contentType | text/picture |
| | | memoType | private/share |
| | | sharingGroup | sharing buddy list |
| | | status | create/delivered/read |
| | | completed | Y/N |
| | | geoloc | |
| | | address | |
| | | latitude | |
| | | longitude | |
| | | timeStamp | |
| | | metadata(_kmd) | |
| | | access control(_acl) | |
| | Users | id | |
| | | userName | |
| | | authToken | |
| | | first_name | |
| | | last_name | |
| | | age | |
| | | profileImage | profile image |
| | | group | |
| | | phoneNum | |
| | | _push | |
| | | metadata(_kmd) | |
| | | access control(_acl) | |
| | UserInfo | id | userName |
| | | buddyList | |
| | | contentsList | |
| | | metadata(_kmd) | |
| | | access control(_acl) | |
| Client | contents | id | |
| | | text | |
| | | situation | |
| | | contentType | |
| | | memoType | |
| | | sharingGroup | |
| | | status | |
| | | complete | |
| | | timeStamp | |
| | User | id | |
| | | userName | |
| | | | |
| | buddyList | buddy_id | buddy user name |
| | | profileImage | buddy profile image thumbnail |

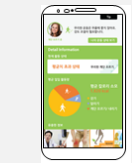
Section 04.

# Architecture Description (Lv. 01)

- 4.1. 주요 설계 전략
- 4.2. Applied Views of Architecture Design
- 4.3. 아키텍처 드라이버 명세서
- 4.4. 아키텍처 설계:Structural View – Component Diagram
- 4.5. 아키텍처 설계:Functional View – Activity/Sequence/Flow Diagram

# 4.1. 주요 설계 결정 (Design Decisions)

## Flexibility of user interface


Smartphone

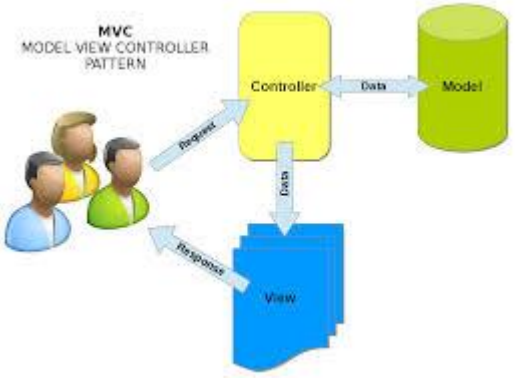

Wearable


Laptop

다양한 UI를 가지는 디바이스 지원 가능

《derive》

**MVC Architecture**




《effect on》

**View Client**

## Reusability of logic and data structure


Time


Location


Weather

…

《derive》

**Service**

- Background Daemon
- BaaS SDK Using
- Local Database
- Context-aware Monitoring
- …

여러 상황정보들로 확장 응용 가능

《effect on》

## Security of access user or data


User Account


Access Control


Authentication

…

《derive》

**Backend-as-a-Service**

- User. Management
- Data Management
- Biz Logic
- Notification…

# 4.2. Applied Views of Architecture Design


## Physical View

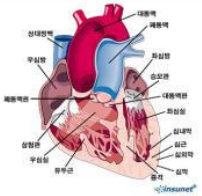
- Smartphone
- IoT/Wearable Dev.
- Cloud (BaaS)
- Sensors
- Networks

**상황기반 Content 공유/알림 서비스**

| Client SW | Service SW | Cloud |
|---|---|---|

## Structural View

- Modules (Components)
  (View Client, Service, Cloud…)
- Interfaces
- Data Schema



## Functional View

- Activity diagrams
- Sequence diagrams
- Data flow

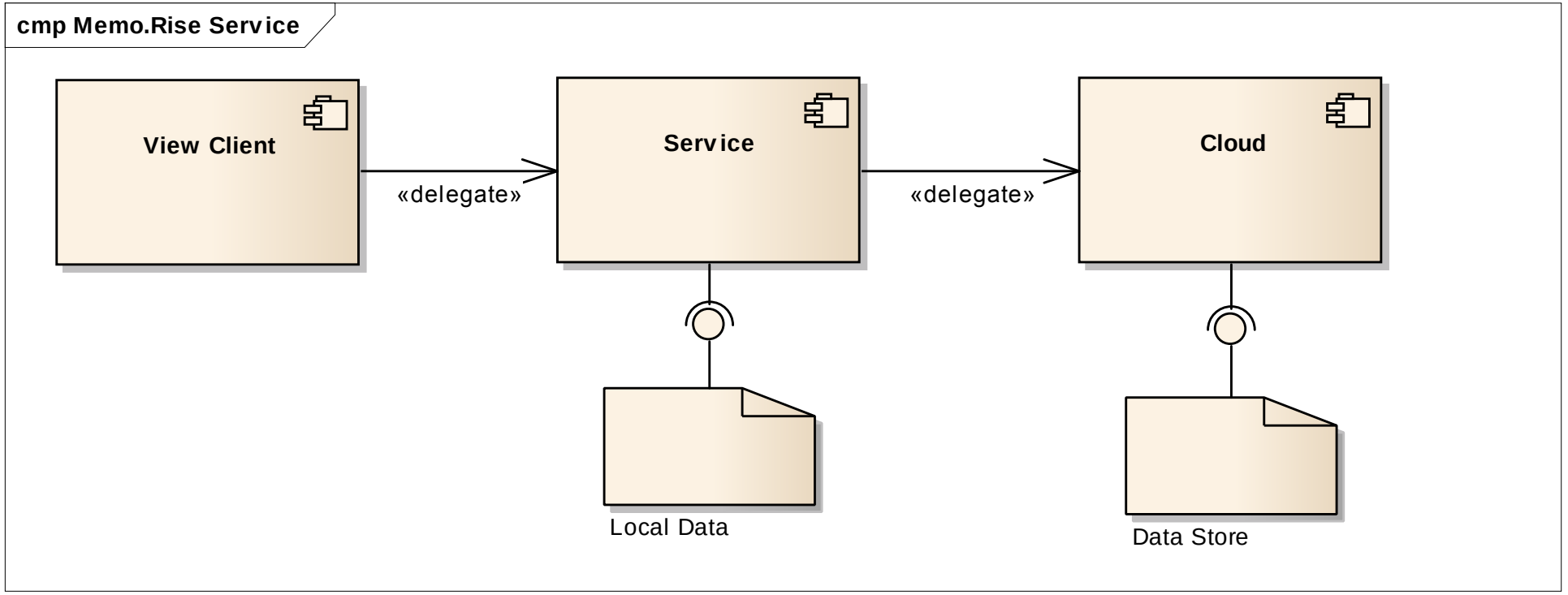
# 4.3. 아키텍처 드라이버 명세서

| 품질속성 | 아키텍처 대안 | | Tradeoff 분석 |
|---|---|---|---|
| Flexibility vs. Performance | 상황기반 Content 공유/알림 서비스는 다양한 상황정보와 함께 여러 Content 유형에 대한 추가를 기존 구조의 최소한 변경으로 기존 서비스에 영향 없이 적용 가능해야 한다. | | 상황기반 Content 공유/알림 서비스는 2개의 Android Application(App/Service)와 클라우드(SDK) 컴포넌트들이 결합되어 있는 형태이다. 따라서 다중 기능을 가지는 컴포넌트들은 서로 인터페이스를 공유한다. |
| | 아키텍처 스타일 | Layered Style | 컴포넌트 내 Modifiability와 Adaptation Layer 추가 시 Portability 지원하는 장점이 있지만, 현 서비스의 컴포넌트들이 서로 유기적인 결합이 발생하므로 적용하기 힘든 패턴임 |
| | | Decomposition Style | 구조나 업무 분리는 용이하지만, 한 컴포넌트가 하나 이상의 연관 관계를 가지기 어려워서 적용이 힘듦 |
| | | Uses Style | 컴포넌트 간 서로 depends-on 관계에 있으며, subset들의 계획을 지원함. 또한, 변경의 효과를 확인할 수 있음. 기본적인 Uses Style에 Event-based Style과 Factory Design Pattern 활용 가능. |
| Reusability vs. Maintenance | 상황기반 Content 공유/알림 서비스는 다양한 클라이언트 플랫폼을 고려하여 설계되어야 하고 공통부의 재사용성을 지원해야 한다. 또한, 구조적으로 플랫폼 dependency를 최소화 해야 한다. 이는 실행되는 소프트웨어의 패키지 구조 기반이 된다. | | 상황기반 Content 공유/알림 서비스는 다양한 클라이언트 플랫폼을 지원하기 위해 View 역할의 분리를 필요로 한다. 즉, 특정 플랫폼들에 의존적이지 않은 View와 데이터 제공 구조를 가져야 한다. |
| | 아키텍처 스타일 | Data Model Style | 데이터 Entity들과 Relationship의 구조를 가지고 저장되는 Content들을 표현하며, 실제 Data Store로의 Access를 관리. 이는 View 구조와의 분리를 통해 재사용성 제공 가능. |
| | | Shared Data Style | 공유된 데이터에 접근가능하게 하고, 데이터 생산자와 소비자 간의 decoupling도 가능함. 각 컴포넌트들의 데이터가 일관적이지 않아 활용 제한. |

# 4.3. 아키텍처 드라이버 명세서

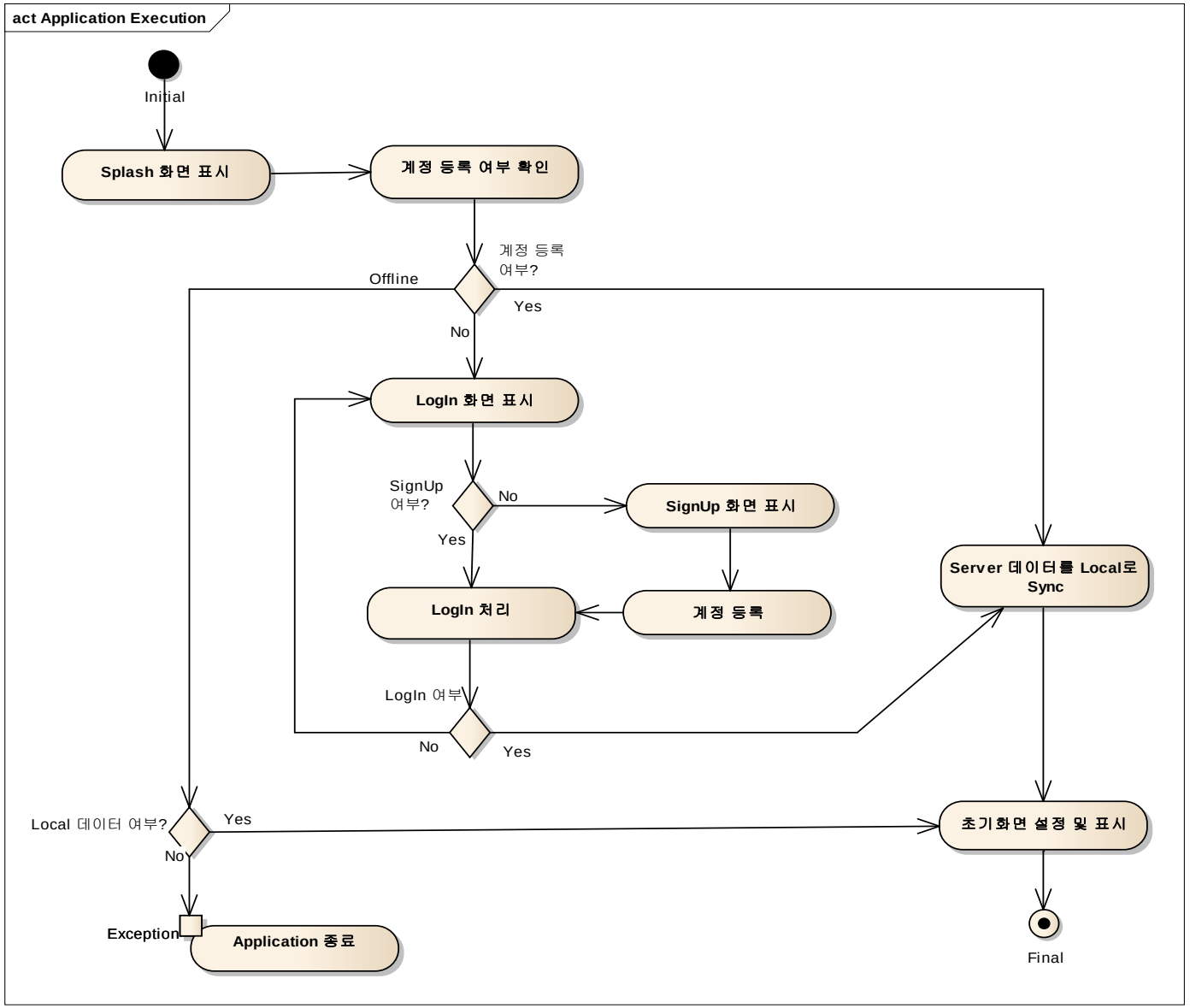
| 품질속성 | 아키텍처 대안 | | Tradeoff 분석 |
|---|---|---|---|
| Security vs. Time | 상황기반 Content 공유/알림 서비스는 사용자에 대한 접근 제한을 고려해야 한다. 사용자 본인 뿐만 아니라, 친구들과의 Content 공유 시에도 이와 함께 비용적인 측면을 고려한 구조가 필요하다. | | 상황기반 Content 공유/알림 서비스는 사용자 계정과 함께 접근 권한 설정이 가능해야 하며 그 권한별 Content를 활용할 수 있어야 한다. |
| | 아키텍처 스타일 | Deployment Style | TPM과 같은 HW와 연동 시 활용 가능. SW Element와 Environmental Elelement의 연관 관계로 할당과 Execution/Migration 이용. |
| | | Work Assignment Style | 클라우드의 사용자 계정, 접근 관리, 데이터 저장소 각각의 컴포넌트에 역할(Work)를 할당하고 각 역할별로 Responsibility 할당. Data stores style과 함께 활용 가능. |

- CD for Situation-based Contents Share/Notification Service



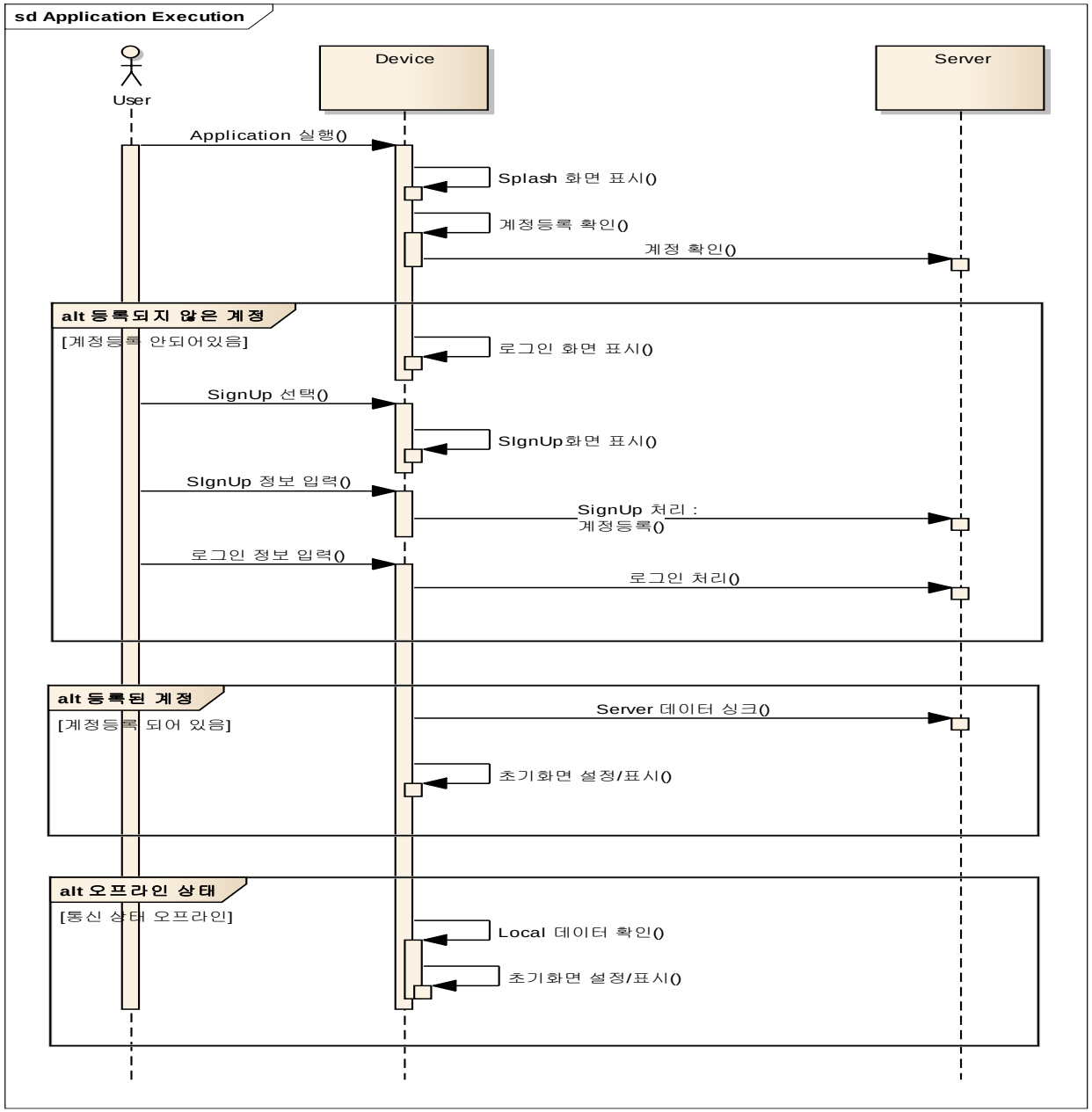
# 4.5. Functional View – Activity Diagram

- **AD for Service**

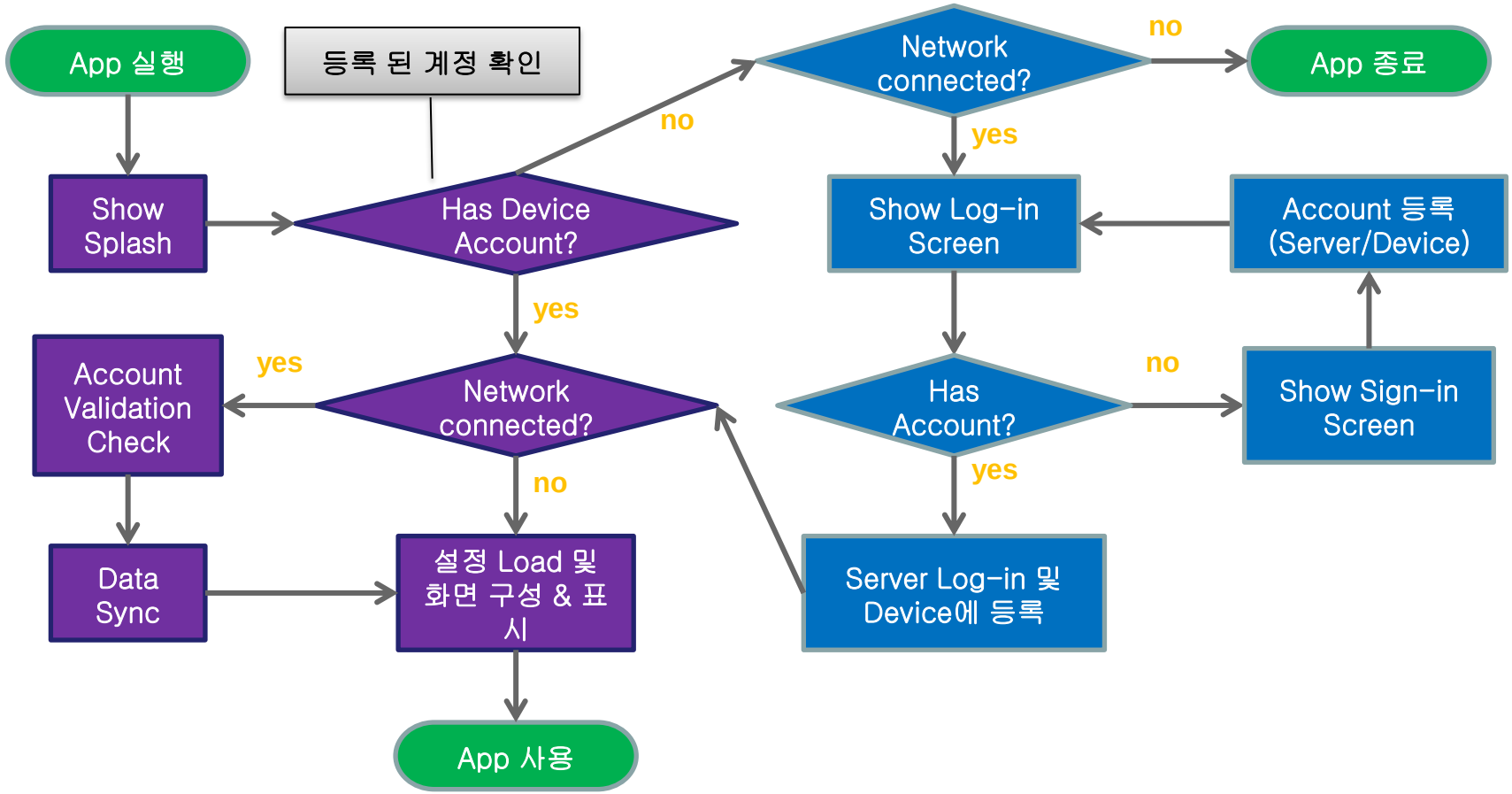


# 4.5. Functional View – Sequence Diagram

- **SD for Service**



- **FD for Service**



Section 05.

# Architecture Description (Lv. 02)

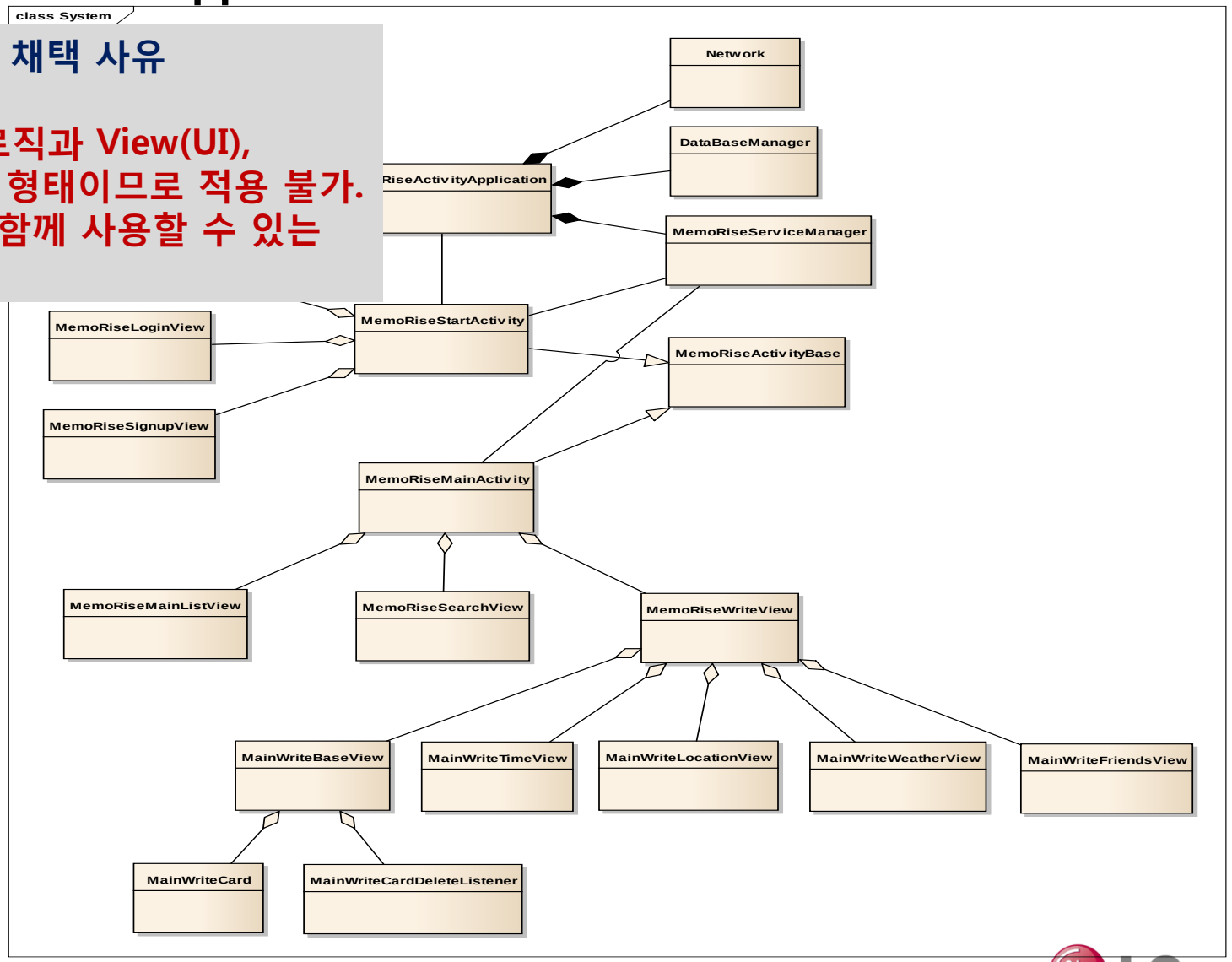
- 5.1. 적용된 아키텍처 패턴
- 5.2. Tactics for Quality Attributes
- 5.3. Structural View – Class Diagram
- 5.4. Dynamic View – App User Life Cycle

## View 클라이언트 컴포넌트 : Application Controller 패턴

*** 기타 대안 및 해당 아키텍처 채택 사유**

MVC 패턴

**: Application 특정 Control 로직과 View(UI), Model 코드가 완전히 분리된 형태이므로 적용 불가. 따라서, Flow 로직과 View를 함께 사용할 수 있는 ACP 패턴 활용함.**

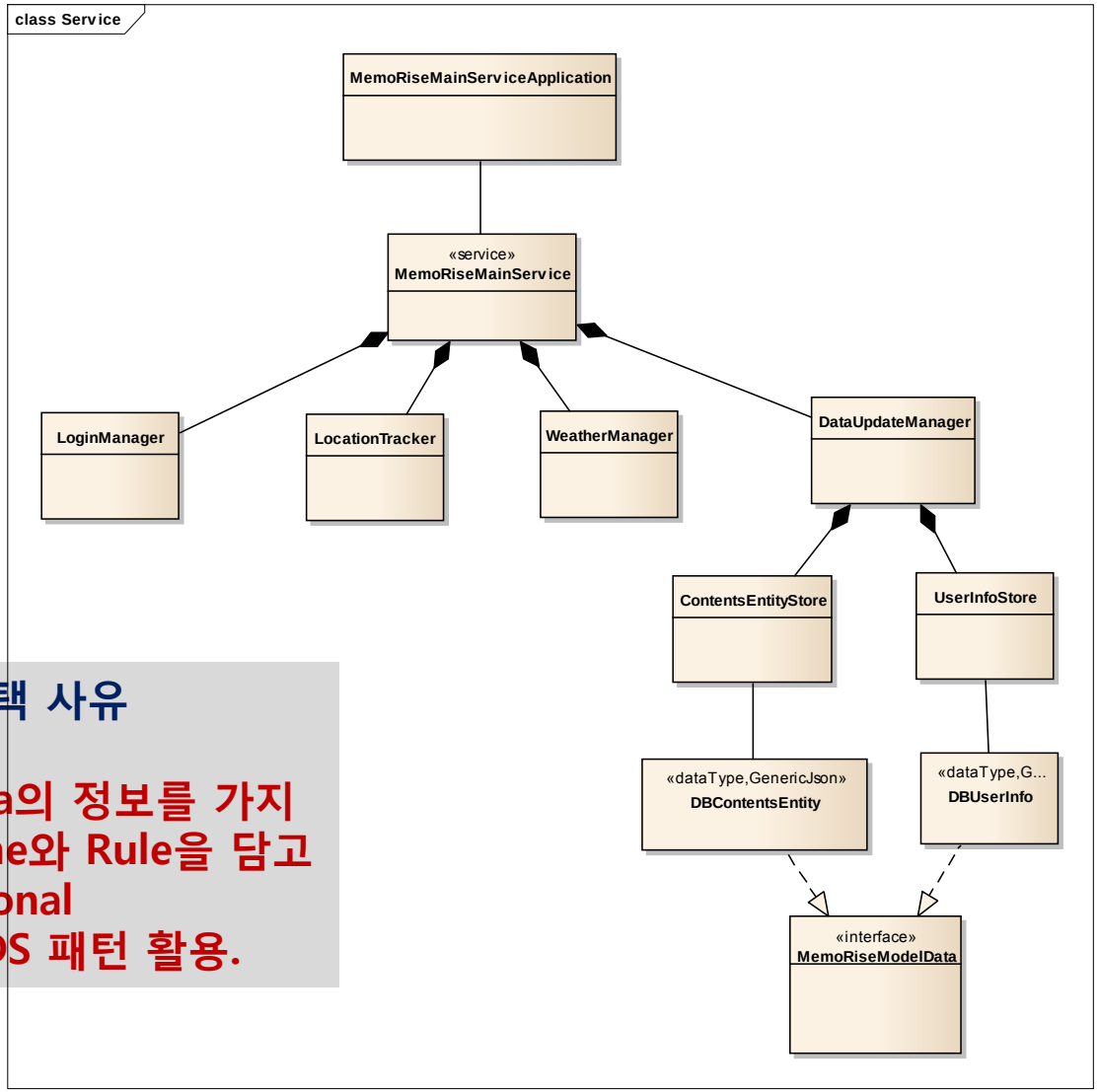







34

## Data 관리 컴포넌트들 : Data Stores 아키텍처 패턴




*** 기타 대안 및 해당 아키텍처 채택 사유**

Shared-Data Pattern

**: Data Entity들이 실제 저장 Data의 정보를 가지는 Object로 사용되어, Key, Name와 Rule을 담고 있어야 함. 기능적 의존성(Functional Dependency)은 피해야 함으로 DS 패턴 활용.**





35

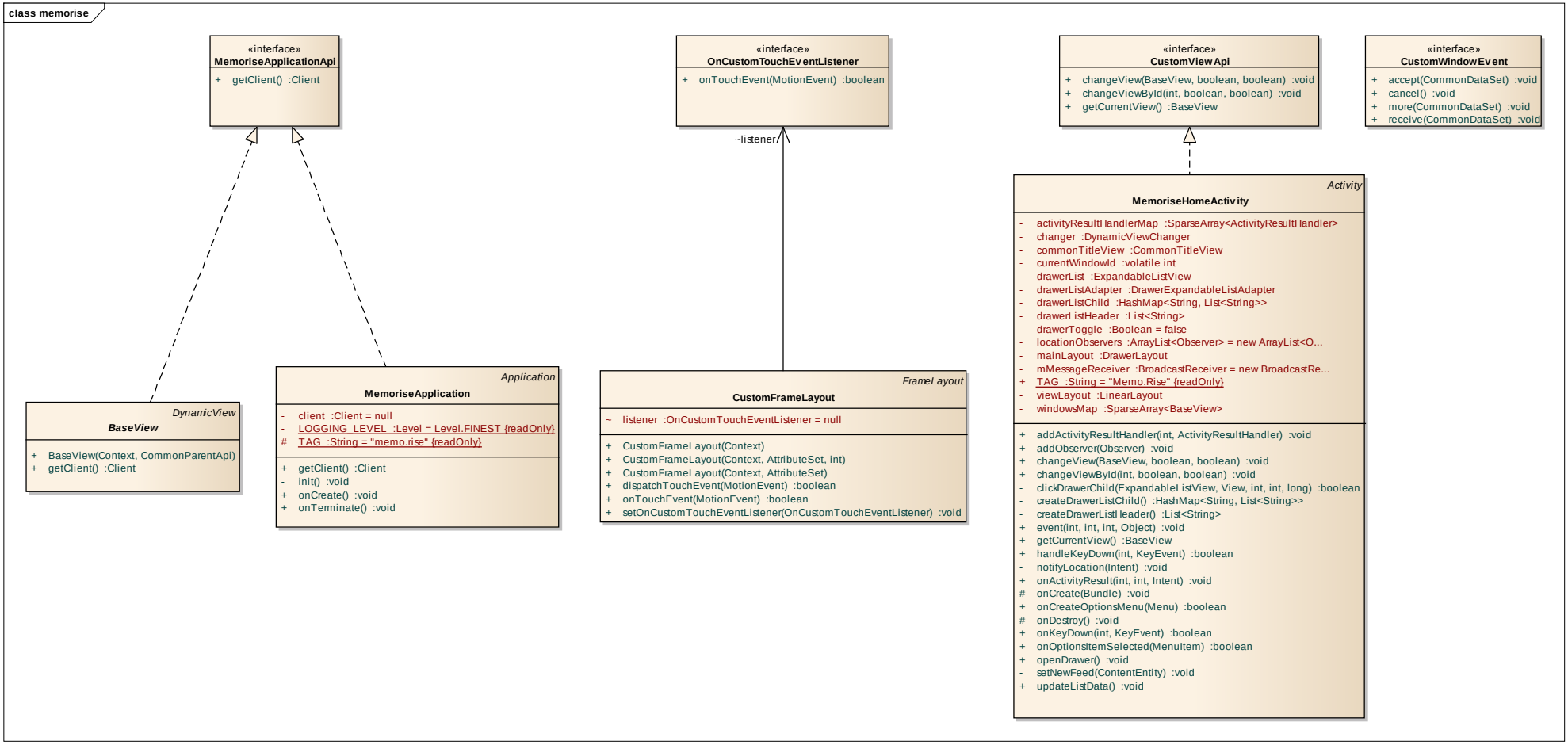
## View 클라이언트 컴포넌트 : Application Controller 패턴

| Application Controller 패턴 활용 | |
|---|---|
| 주요 기술적 접근방법 | Application의 View Flow를 모아서 관리하여 모든 View들의 로직을 바꾸지 않고 한 곳에서 로직과 Flow를 변경 가능한 ACP 패턴을 사용함. |
| Promoted 품질속성 | Flexibility, Reusability, Modularity, Extensibility |
| Promoted 품질속성 사유 | View(UI) 별로 동일한 로직 객체를 활용할 수 있으며, Application 레벨에서 Test가 가능하며, 요청 기능들에 대해 유연성을 가질 수 있음. 요청에 대한 다양한 실행을 개별적으로 제공하므로 변경이 용이함. |
| Inhibited 품질속성 | Efficiency, Complexity |
| Inhibited 품질속성 사유 | 전달 파라미터의 복잡성이 증가할 수 있고 Efficiency의 Loss가 발생할 수 있음. |

## Data 관리 컴포넌트들 : Data Stores 아키텍처 패턴

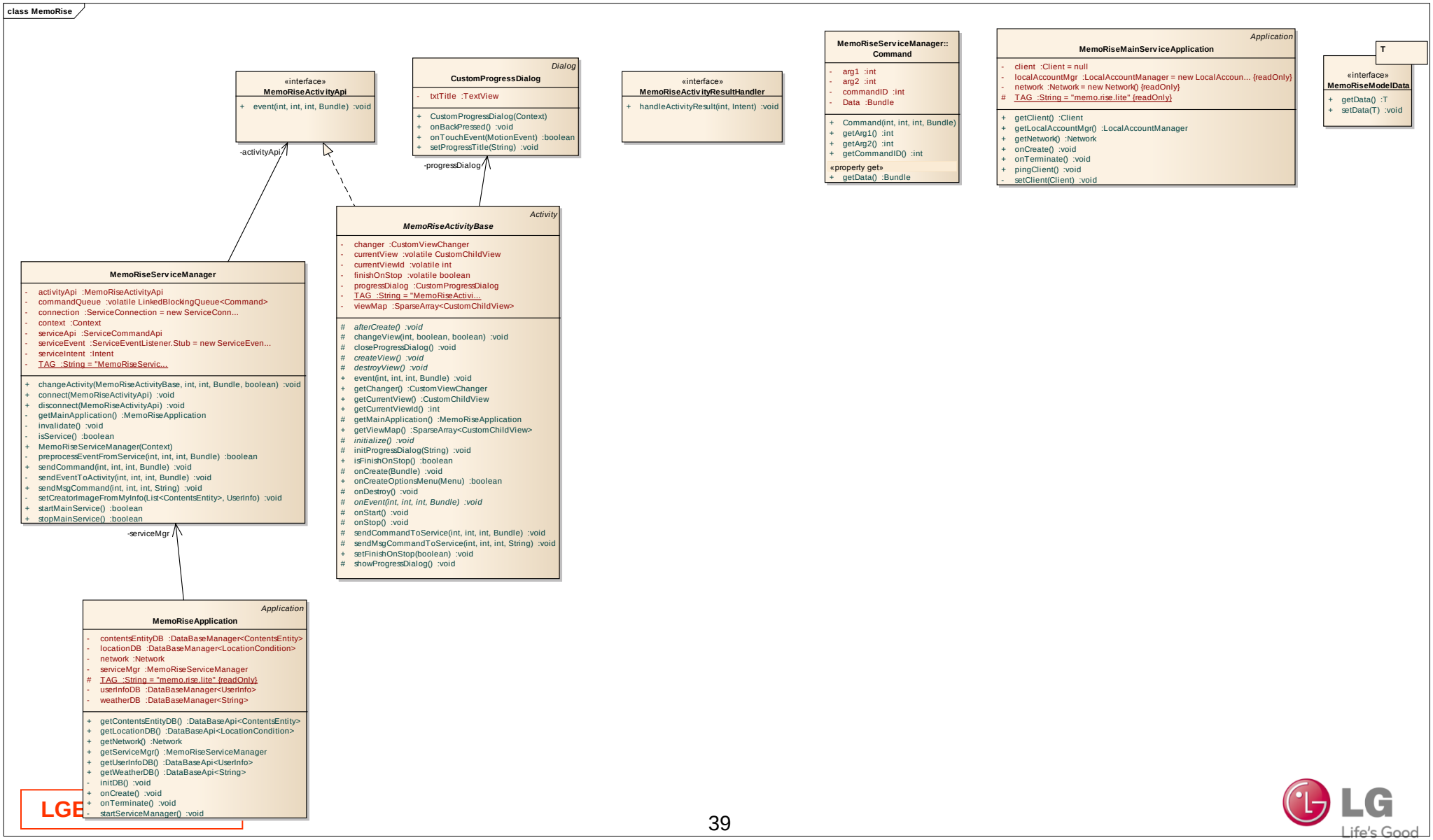
| Data Stores 패턴 활용 | |
|---|---|
| 주요 기술적 접근방법 | Data 관리/운영과 함께 Data를 모으기 위해 DS 패턴을 사용함. |
| Promoted 품질속성 | Reusability, Maintainability, Complexity |
| Promoted 품질속성 사유 | Direct Access의 필요성을 줄이고, 표준 format으로 데이터 제공 가능. Maintenance 비용 줄일 수 있음. |
| Inhibited 품질속성 | Efficiency, Performance |
| Inhibited 품질속성 사유 | Historical한 데이터는 주의하지 않으면 loss가 발생할 수 있음. 데이터 변환 시에 문제 발생 가능성 존재. |

- CD for Client - Release 결과 참조

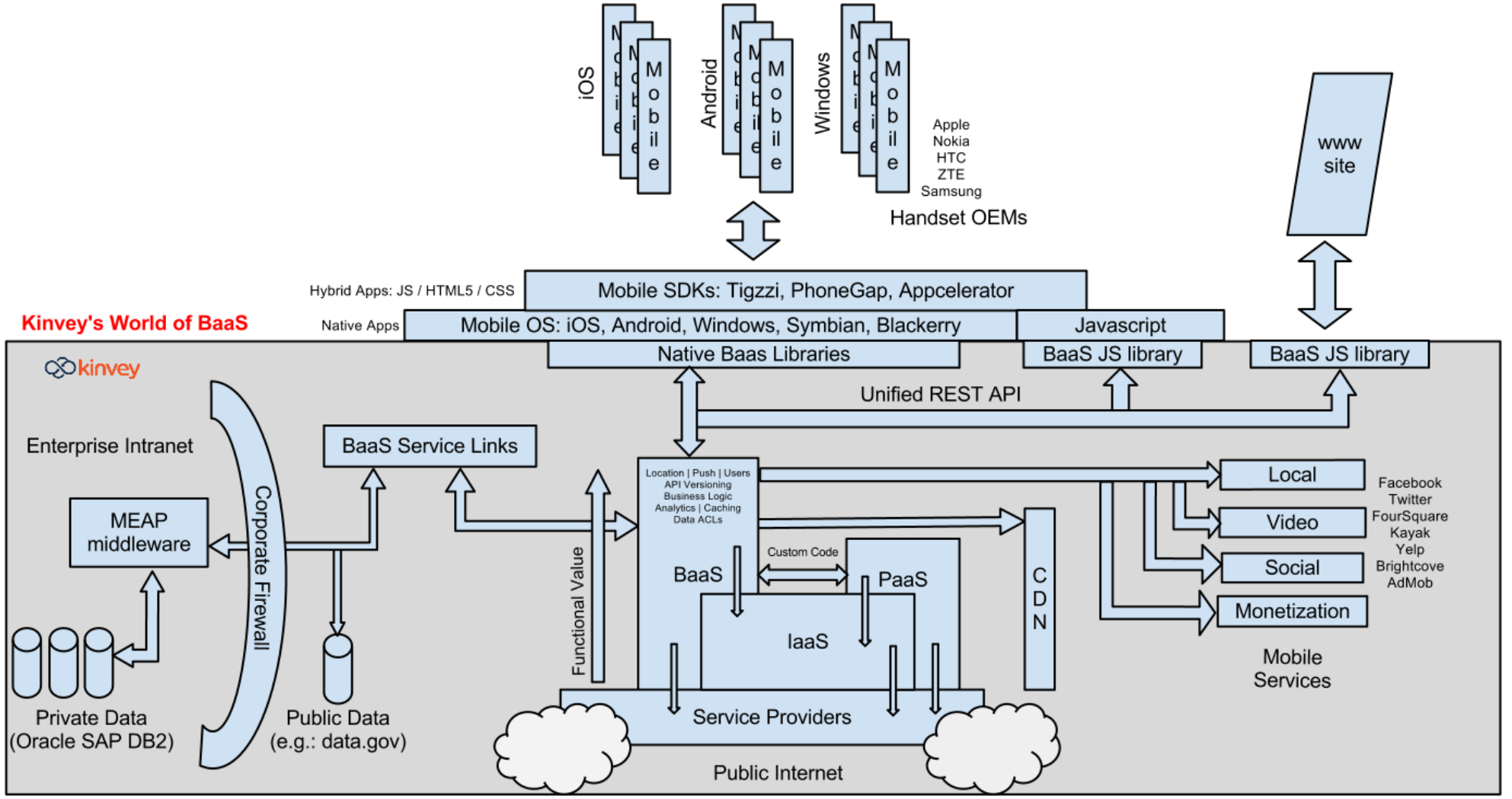


# 5.3. Structural View – Class Diagram

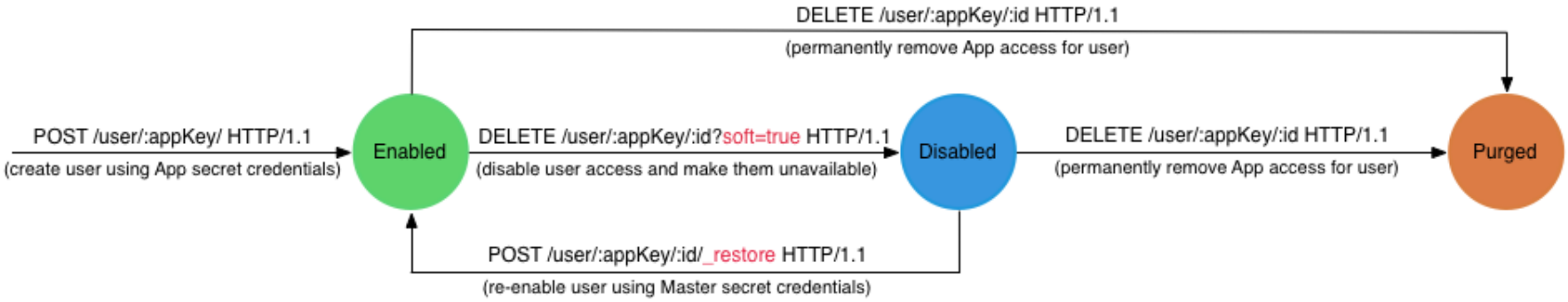
- CD for Service - Release 결과 참조



- **Architectural Flow for BaaS Cloud**



- **User Life Cycle for Cloud (BaaS:Kinvey)**




Contents of user entity at each stage


```
{
  "_id": "517ae1dc4b19a33c10000003",
  "username": "anniebourne",
  "password": "aRandomString",
  "_kmd":
  {
    "lmt": "2013-04-26T20:21:48.582Z",
    "ect": "2013-04-26T20:21:48.582Z"
  },
  "_acl": {
    "creator": "517ae1dc4b19a33c10000003"
  }
}
```

```
{
  "_id": "517ae1dc4b19a33c10000003",
  "username": "anniebourne",
  "password": "aRandomString",
  "_kmd":
  {
    "lmt": "2013-04-26T20:21:48.582Z",
    "ect": "2013-04-26T20:21:48.582Z",
    "status": {
      "val": "disabled",
      "lastChange": "2013-04-26T23:08:21.816Z"
    }
  },
  "_acl": {
    "creator": "517ae1dc4b19a33c10000003"
  }
}
```


User data purged from user collection.

**App User Life Cycle Management - Kinvey REST API Version 1**

Section 06.

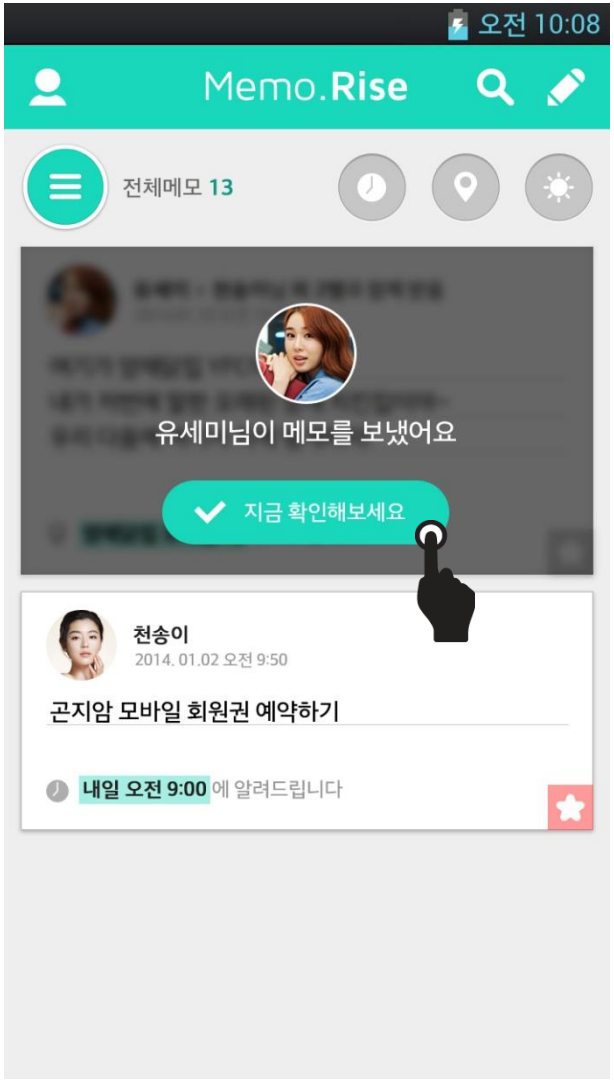
# Service Scenario for Validating Architecture

- 6.1. Use case Description
- 6.2. Memo.Rise U.I. Design
- 6.3. Memo.Rise Prototyping

# 6.1. Usecase Description

| Use Case Title: 특정 상황에 친구에게 알려주는 메모 | Use Case ID: UC_001 |
|---|---|
| General Use Case Description: 특정 상황(시간, 위치, 날씨 등)이 되면 작성한 메모를 친구에게 알려주는 서비스 | |
| Entities Involved: Android Application(클라이언트, 서비스), 클라우드(BaaS) | |
| Preconditions:<br>사용자가 등록한 친구 리스트가 존재한다.<br>위치명에 대해 위치정보를 검색해 주는 서비스가 존재한다.<br>지명, 일시를 통해 날씨를 알아내는 서비스가 존재한다. | |
| Primary Use Case Flow of Events: | |
| 1. 사용자는 앱 상의 메모 작성 아이콘을 선택한다. | |
| 2. 메모 작성 창이 뜨면 메모를 작성한다. | |
| 3. 친구 추가를 눌러 공유할 친구를 선택한다. 이 때, 친구 이름으로 검색 가능. | |
| 4. 시간 아이콘을 선택한다. | |
| 5. 시간 설정 창이 뜨며, 시간정보를 설정(날짜, 시각, 반복여부)한다. 이 때 반복여부는 “안함, 매일, 매주, 2주마다, 매달, 매년” 존재. | |
| 6. 위치 아이콘을 선택한다. | |
| 7. 위치 설정 창이 뜨며, 위치정보를 설정(지도위치, 도착할때/떠날때)한다. 이 때 지명으로 지도의 위치 검색이 가능하다. | |
| 8. 작성 완료 아이콘을 누르면 해당 메모가 본인과 공유한 친구에게 전달된다. | |
| 9. 공유된 친구에게 Push Notification이 뜬다. | |
| 10. Notification을 클릭하면 바로 메모라이즈의 리스트를 보여준다. 이 때 리스트는 정렬 가능하다. | |
| Primary Use Case Post Conditions:<br>사용자는 메모라이즈 앱 상에서 메모 결과를 확인할 수 있다. | |

# 6.2. Memo.Rise UI Design - Main

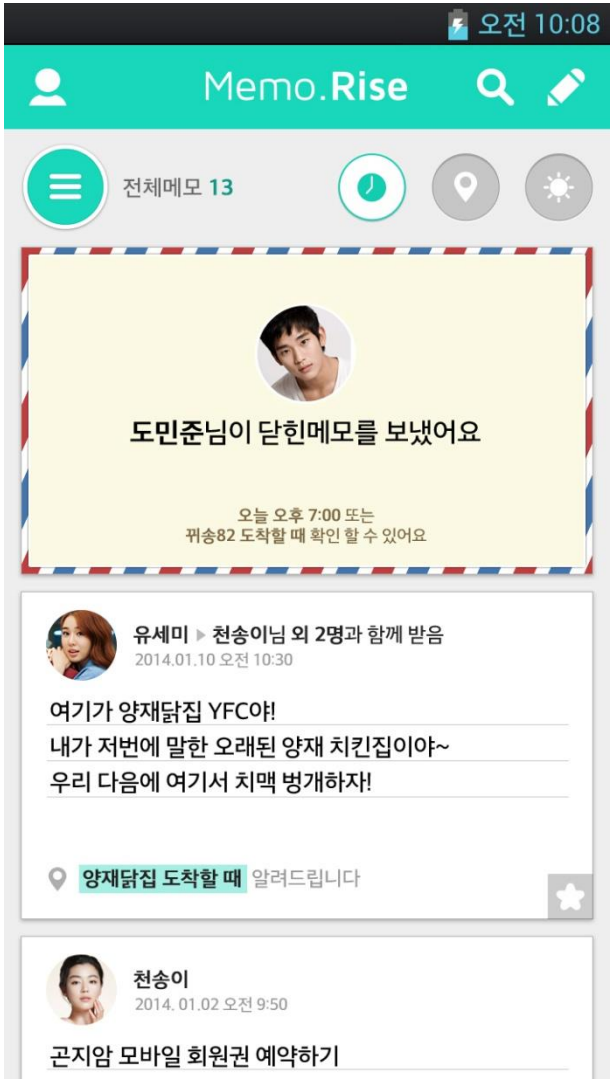


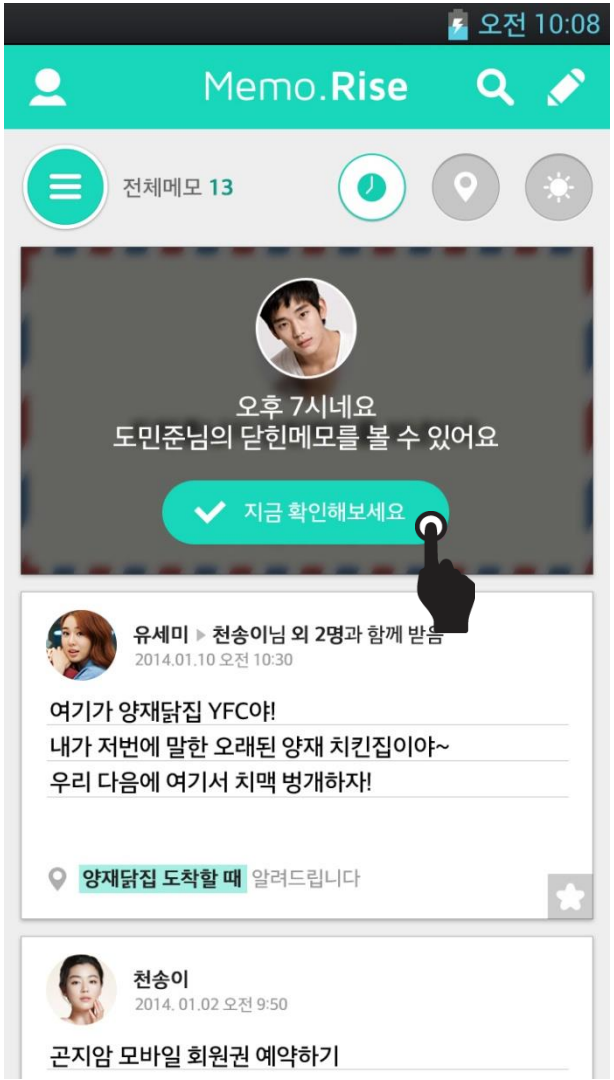

• 친구가 보낸 메모에 알림 메시지 함께 제공


• 확인 버튼 터치 후 친구의 메모가 보임


• 상황 선택 시 뜨는 팝업

# 6.2. Memo.Rise UI Design - Main



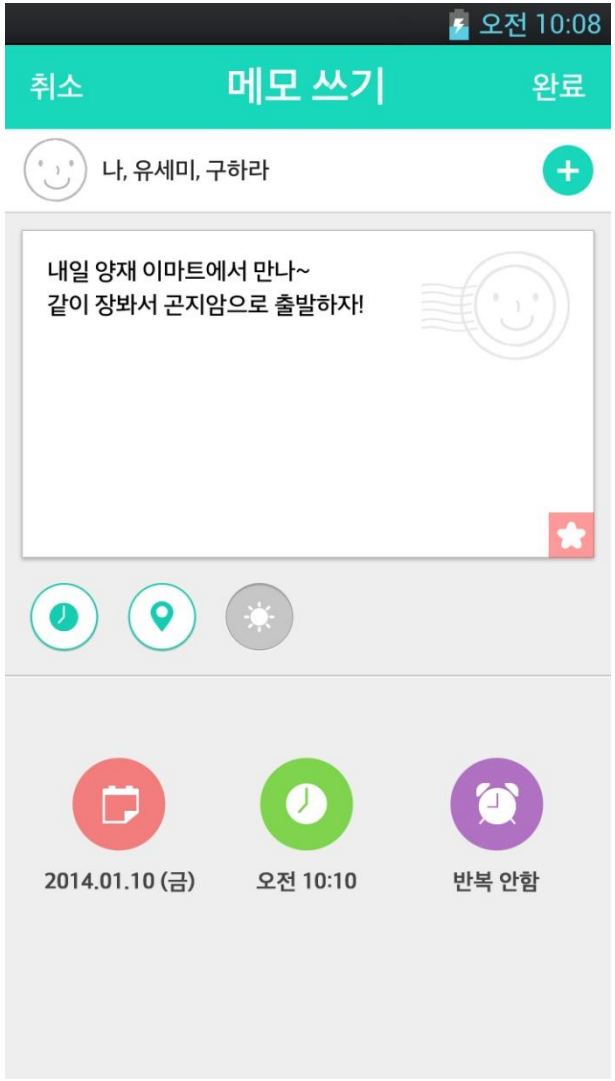





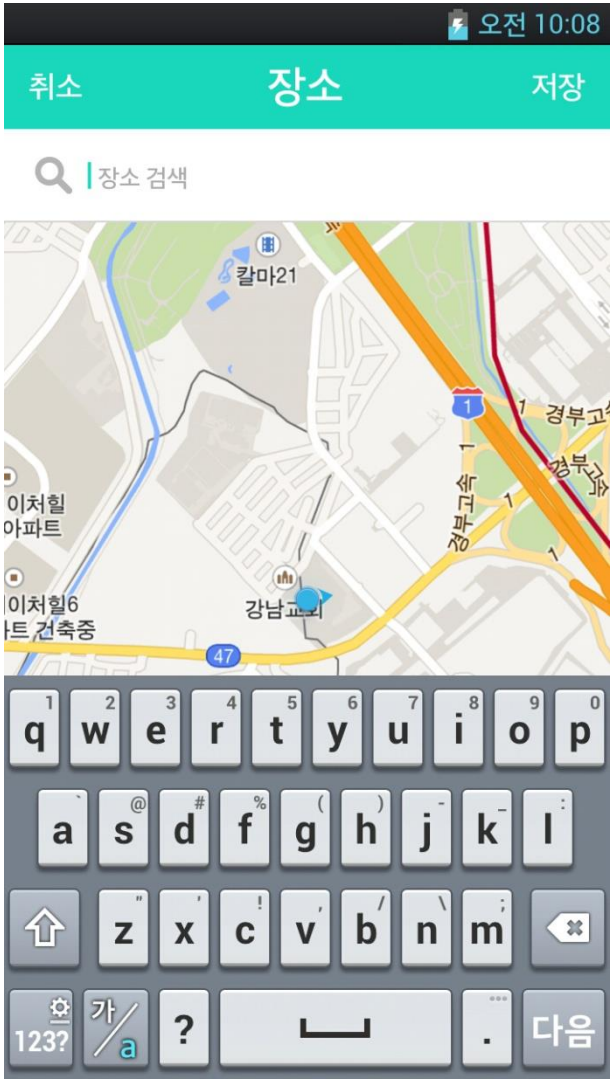
- 닫힌 메모는 **설정한 상황 전 까지 메모 내용을 확인 할 수 없음**

- 상황 도래 시 시간/장소/날씨 중 어떤 상황으로 메모 내용을 확인 할 수 있는지 알림 메시지와 함께 알려줌

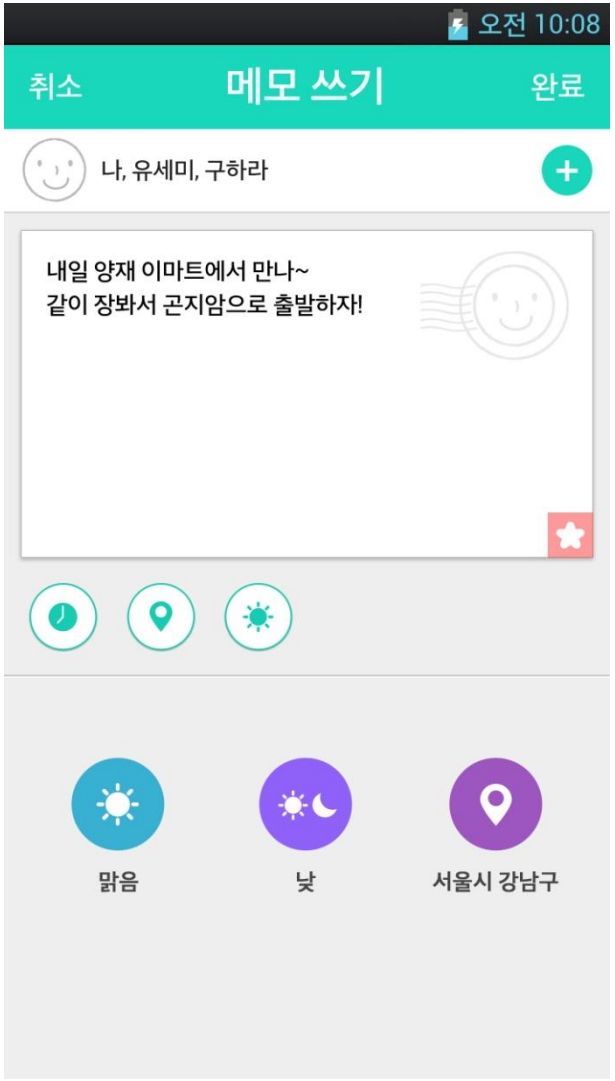
# 6.2. Memo.Rise UI Design – 상황 설정




• 시간입력




• 장소입력




• 날씨입력

- **Dev. Environment**
  - Android Smartphone
    - Java
    - Kinvey SDK
    - Google Play
  - Kinvey(BaaS)
    - Java Script
- **Test Environment**
  - JUnit

Section 07.

# Lessons Learned & Future Works / 사업적 기여 및 성과 항목

- Lessons Learned & Future Works
- 사업적 기여 및 성과 항목

- **Architecture 프로세스 틀 견고화**
  - 요구사항 관리/변경, 구조 설계 지원 프로세스
- **SW Review 프로세스 진행 (요구사항, 설계, 코드 등)**

**SW 아키텍트로서 기여항목**

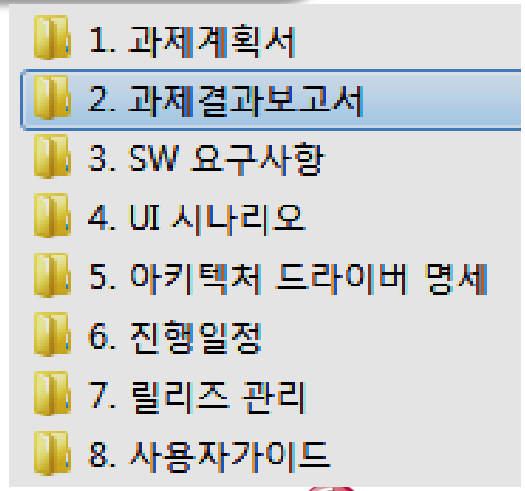
- 아이디어 제안/기획
- 요구사항 및 도메인 Context 분석
- 클라우드 솔루션 분석/비교/선정 (BaaS:Kinvey)
- 시나리오(Usecase) 전개 및 품질속성 도출
- 아키텍처 및 클래스 설계, 공유 및 전파
- SW Review로 역량 강화 및 구현 안정성/품질 강화
- 구현 및 설계 이슈 디버깅/개선
- BaaS 솔루션 Practice 전파
- 연관 특허 출원

**Reflection**

- Resource가 부족할 때 상황 대응
- 기존 팀/파트/PL 구조의 한계 (유연한 조직체계 필요)
- PM, Architect 조직 필요
- CASE Tool 사용 및 설계 노하우 팀원 공유 부족
- 파트 Boundary 없이 업무 진행 어려움
- BaaS 솔루션에 대한 심도 깊은 이해와 활용

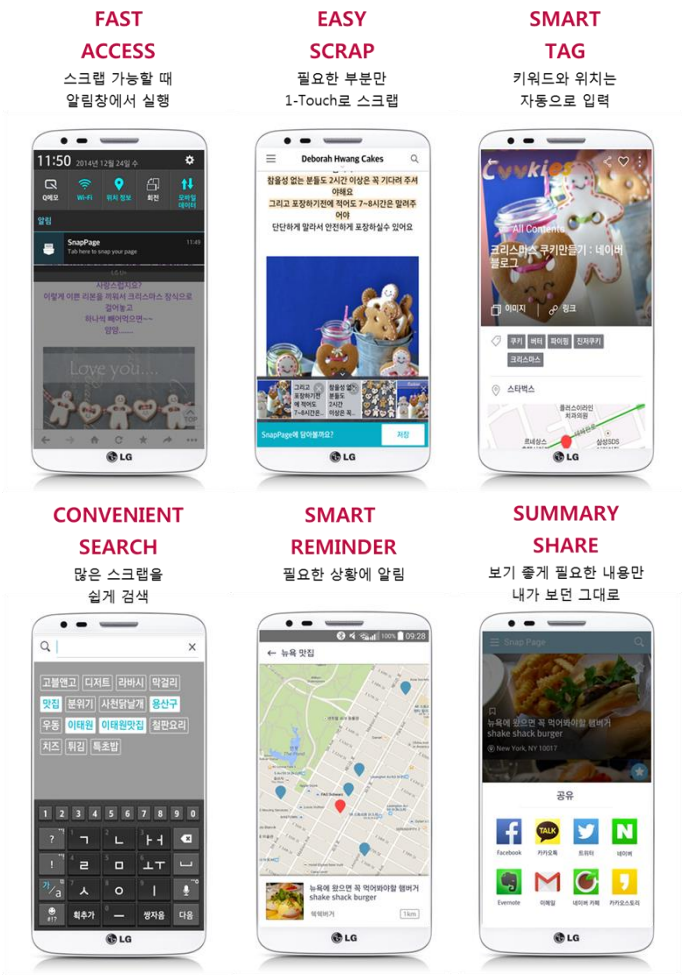
**Future Works**

- 요구사항 변경, 다양한 서비스/플랫폼에 대응 가능한 상시 **Architecture** 프로세스
- **Resource** 확보
- **IoT** 디바이스 연계 대응
- **Content** 및 상황정보 확장
- 요구사항, 구조 설계, 코드에 대한 **SW Review** 프로세스



# 사업적 기여 및 성과 항목

## BaaS 활용 Practice들을 사내 프로젝트에 전파/협업




[SnapPage]

## 연관 특허 출원

발명자 | 나의 IP현황

| 총 출원건 | | 국내 진행 | | 해외 진행 | |
|---|---|---|---|---|---|
| 국내 총 의뢰건 | 29 | 심사청구대상 | 3 | 심사청구대상 | 14 |
| 국내 총 출원건 | 12 | 심사중 | 0 | 심사중 | 0 |
| 해외 기술별 총 출원건 | 24 | 사내등록 | 3 | 등록 | 53 |
| 해외 국가별 총 출원건 | 110 | 공개기보 | 0 | 거절/포기 | 30 |
| | | 등록 | 1 | 등록유지포기 | 0 |
| | | 거절/포기 | 18 | | |
| | | 등록유지포기 | 2 | | |

나의 IP현황 > 총 출원건 > 국내 총 출원건

총건수 : 12건

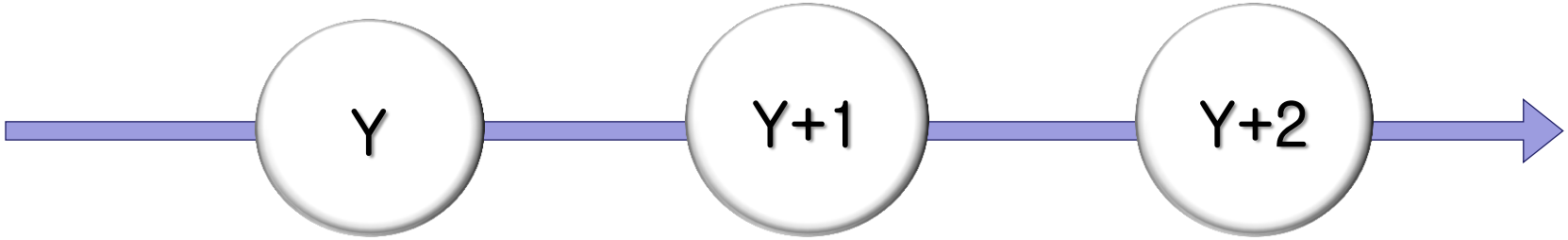
| No | File | 진행상태 | LG-REF | 발명의 명칭 | 발명자 | 사업부 |
|---|---|---|---|---|---|---|
| 1 | | 연차업체이관 | DML08-010 | 멀티 도메인 매니저의 운영 방법 및 도메인 시스 | 조성현(趙 成 鉉),정민 | CTO_Convergence 연구 |
| 2 | | 추가심사청구 | DML09-019 | [UI]메뉴 검색 장치 및 그 방법 | 정민규(鄭 民 圭) | CTO_Software Platform |
| 3 | | 추가심사청구 | DML09-023 | 명령어 처리 장치 및 그 방법 | 정민규(鄭 民 圭) | CTO_Creative Innovatic |
| 4 | | 포기 | CIC12-007 | SMS를 통한 IA 원격 대화 서비스의 인증 시나리 | 최권열(催權烈),박재민 | CTO_Creative Innovatic |
| 5 | | 출원 | CIC13-037 | 이동단말기 및 그것의 제어 방법 | 정민규(鄭 民 圭),안양 | CTO_Creative Innovatic |
| 6 | | 건담당자지정 | | 휴대단말기 환경에서 대용량 파일 Text의 효율적 | 권영미(211763),정민규 | Multimedia 연구소 |
| 7 | | 건담당자지정 | | 새로운 Form Factor와 Display 연동방법을 가지는 | 정민규(鄭 民 圭),천두 | CTO_Creative Innovatic |
| 8 | | 건담당자지정 | | 메시지(음성, 문자 등 멀티미디어 포함) 수신이 | 정민규(鄭 民 圭) | Multimedia 연구소 |
| 9 | | 건담당자지정 | | 모바일기기에서 Tracing log 관리 및 활용방안 | 정민규(鄭 民 圭) | Multimedia 연구소 |
| 10 | | 건담당자지정 | | 특정 Sync Protocol 지원이 불가능한 PC와 휴대단 | 정민규(鄭 民 圭) | Multimedia 연구소 |
| 11 | | 건담당자지정 | | IPTV interoperability 구현 기법 | 권영미(211763),정민규 | Multimedia 연구소 |
| 12 | | 건담당자지정 | | 기능별 확장가능한 모바일기기와 그 활용방법 | 정민규(鄭 民 圭) | Multimedia 연구소 |

# Q & A

# Appendix

- Roadmap
- Value Proposition
- 품질 속성 시나리오

# Roadmap




| | | | |
|---|---|---|---|
| 주요 특징 | - **Approach**<br>: Application (Content 생성/소비)<br>: BaaS (Content 중계)<br>- Client/Server Architecture를 활용한 상황정보 기반 메모<br>- Serendipity, Mission, Caring 시나리오 | + **Approach**<br>: Device (Content 생성/중계/소비)<br>+ IoT 기반 기기 연동<br>: P2P Architecture를 활용한 상황정보 기반 Content(Text, Media)<br>+ Collaboration 시나리오 | + **Approach**<br>: Content/Application 플랫폼<br>+ Platform 기반으로 한 Ecosystem 구축 |
| 비즈니스 특징 | - AppStore 기반 서비스 Incubation<br>-특화시나리오(ex. Random, 특정상황 등) 활용 Memo Sales | + 다양한 형태(c.f. iBeacon, DLNA 등) Device Sales<br>+ Content 서비스 확장 | + 상황정보 기반의 Content Marketing<br>+ Ecosystem 혹은 플랫폼 사용료 |

# Value Proposition

## For 정보나 데이터 사용이 많은 사람

+ 다양한 스팸 정보들 사이에서 **꼭 원하는 정보**를 알림 받음

+ 해야지 하고는 잊어버리는 SMS 문자나 일반 메모와 달리 **필요한 상황**에 알림

## For 누군가를 Caring하고자 하는 사람

+ **도움이 필요하거나 챙겨줘야 할 대상**에게 사용자가 지정한 상황에 알림/서비스 제공

## *For LG*

***+ Content(메모, Media...) 서비스 제공 (SBC 또는 3rd Party 제휴)***

***+ Device Sales (MC사업본부 등)***

***+ 사용자 인프라 및 데이터 확보***

## For 다른 사람에게 정보/데이터를 전달하고자 하는 사람

+ 사용자가 알고 있는 **정보나 획득한 데이터를 다른 사람에게 전달**

## For 다른 사람과 협업하고 싶어하는 사람

+ 공동의 관심사, 업무, 작업 등을 **여러 사람과 같이 각자의 상황에 맞추어 진행**

# Value Proposition

## 고객 관점

+ 사용자 상황에 맞는 Content(정보/데이터 등) 알림 또는 공유
+ 관계 Network 내 사용자 간 서로 Collaboration 할 수 있는 환경 제공
+ 손쉬운 Content 생성

+ 사용자 경험 확장
사용자 목적(단순알림, 전달, 협업, Caring 등)에 따라 다양한 활용
+ 다양한 상황(시간, 장소, 날씨, Activity, 교통상황 등)에 따른 대응
+ Serendipity 경험 요소

## 비즈니스 관점

+ Positioning & Segmentation
1st target-가족,친구 2nd target-CP/SP 이용자
+ 특화 서비스 유료화
사용자 인프라 및 데이터 활용 유료서비스 전개
+ 서비스 확장
일정, 알람, 리마인더 등을 확장하여 Content 결합(생성/중계/소비) 알림 서비스
+ 틀을 깨는 새로운 메모 (Q.Memo, SNS, SMS 등 포함)

+ 다양한 Device 시장 공략 가능 (Device Sales)
Content 생성/중계/소비 기능 조합 Device
+ Wearable 및 IOT 기술 융합 서비스 (Device Sales)
+ Application(CIC)에서 Device(iBeacon, DLNA 등 : MC, HE 등)로 플랫폼 확장 (Supply chain 시장 확대)
+ 플랫폼 고도화 (기존 업체와 제휴)
Content Provider와 Consumer와 상생
+ 자사 Device 사용자 확대

- 품질 속성 시나리오

| 품질속성 | 품질속성 시나리오 | | |
|---|---|---|---|
| Flexibility | Scene_01. 시스템 실행 시 상황정보의 종류(현재 시간, 위치, 날씨)나 사용자 데이터 및 Content(현재 메모)를 확장하고자 할 때 기존의 아키텍처나 데이터 모델의 유연성을 확보하여 개발 할 수 있다. | | |
| | Scenario 1 | Stimulus | 상황정보, 사용자 데이터, Content가 추가/확장 |
| | | Source | 운영되고 있는 시스템 |
| | | Environment | 시스템 운용되고 있는 중 |
| | | Artifact | 유연성 지원을 위한 Flow 관리와 데이터 구조의 분리 |
| | | Response | 추가/확장된 시스템 실행 |
| | | measure | 추가/확장 개발의 용이성 및 동작 여부 |
| Reusability | Scene_01. 시스템 실행 시 필요한 공통 기능인 1) 서비스 컴포넌트의 데이터 관리 모듈, 2) 상황정보 연동 모듈 등을 기존 시스템을 활용하여 다양한 플랫폼 어플리케이션을 개발하고자 할 때 View 클라이언트 부분만 추가 개발하고 해당 관련 기능성을 재사용할 수 있다. | | |
| | Scenario 1 | Stimulus | 서비스 컴포넌트의 재사용 기능성을 Library 형태를 활용하여 다른 내부 시스템 컴포넌트, 외부 시스템 등이 사용 |
| | | Source | 다른 내부 컴포넌트, 외부 시스템 컴포넌트 |
| | | Environment | 1. 시스템 운용되고 있는 중 내부 데이터/상황정보 추가 요청한 경우<br>2. 외부 시스템에서 기능성을 요청한 경우 |
| | | Artifact | Library로 개발해놓은 재사용 가능한 기능성/비기능성 |
| | | Response | 요청에 따른 기능성/비기능성 실행 |
| | | measure | 다른 내부 컴포넌트, 혹은 외부 시스템 등이 원하는 기능이 정상적으로 동작하였는지 여부 |

- 품질 속성 시나리오

| 품질속성 | 품질속성 시나리오 | | |
|---|---|---|---|
| Security | Scene_01. 시스템이 운용되고 있는 중에 사용자가 데이터 접근 시 권한이 있는 Content와 사용자 정보에만 접근할 수 있어야 한다. | | |
| | Scenario 3 | Stimulus | 시스템이 관리하는 사용자 데이터 접근 발생 |
| | | Source | 시스템 데이터 관리 컴포넌트 및 API |
| | | Environment | 시스템이 운용되고 있는 중<br>사용자의 여러 기능 실행 중 |
| | | Artifact | 시스템의 데이터 접근 모듈 |
| | | Response | 사용자에 대한 인증 및 접근 제한 |
| | | Measure | 권한이 없는 사용자의 접근 제한 확인<br>권한 있는 사용자의 접근 가능 확인 |

# 소프트웨어 아키텍처 범위