# **Software Requirement Specification**



CONFIDENTIALITY/SECURITY WARNING

이 문서에 포함되어 있는 정보는 호서대학교 컴퓨터정보공학부의 자산이며, 호서대학교의 서면 허락 없이 사용되거나, 재 가공 되거나, 외부로 반출 할 수 없습니다.  
이 문서에 대한 열람 및 보관할 수 있는 사람은 분실, 도난, 반출에 대한 책임을 자동 승계하는 것에 대하여 동의하는 자에 한 합니다.

프로젝트명 / 버전

소속/ 작성자

# **1 프로젝트 개요**

## **1.1 프로젝트의 목표**

프로젝트의 수행 목표를 서술한다.

## **1.2 프로젝트의 범위**

프로젝트 수행(또는 개발) 범위와 프로젝트 결과물의 적용 범위를 서술한다.

## **1.3 문서 규칙**

본 문서의 작성과 읽을 때 필요한 규칙은 다음과 같다.

* + 굵은 텍스트는 강조해 야할 문자열에 사용하는 스타일이다.
  + 그림, 표는 반드시 캡션을 붙여서 사용한다. 캡션은 그림이나 표 위에 기재한다.
* 문서의 버전 및 표기 규칙은 다음과 같다.
  + 요구 사항이 확정된 상태를 "기본 형상(baseline)" 이라 한다. 최초 기본 형상 승인 시점의 문서 버전은 "1.0"으로 권장한다.
  + 기본 형상이 변경되어 재승인을 거칠 땐, 공식적 배포(예: RTM, 리뷰 등)가 이루어 질 때 마이너 버전을 "1"씩 증가시킨다.
* 기능 요구 사항의 검토 단계를 표현하기 위해 레이블(Label)을 사용한다.
  + **[TBD]**는아직 결정되지 않은 사항(TBD, To Be Determined)을 표시하는 레이블이다. 기본 형상의 선언 시점에는 [TBD] 기능 요구사항이 없어야 한다.
  + **[Next]**는 프로젝트의 구현 범위에서 제외할 요구 사항을 표시하는 레이블이다. [Next] 요구 사항은 추후 개발 범위에서 다시 검토한다.
* 기능 요구 사항의 우선순위를 표시 할 필요가 있을 때는 아래와 같이 표기한다. 기능 요구 사항은 고유한 식별 ID 부여를 권장 한다.
  + 숫자가 작을수록 우선순위는 높다.
  + 상위 우선순위를 상속한다. 상위와 다를 경우 별도로 표시한다.
  + **(우1)**: 높은 우선순위, 핵심 구동 기능
  + **(우2)**: 중간 우선순위, 구현에 큰 위험이 없는 기능. 사용자는 다음 버전까지 기다릴 수 있다.
  + **(우3)**: 낮은 우선순위, 위급하지 않은 기능. 사용자는 영원히 기다릴 수도 있다.
  + **(우4)**: 불필요한 기능, 개발시간만 소비하는 기능. 제품의 가치와 하등의 관계가 없는 기능
* 기능 요구 사항의 변경을 표시할 때 준수할 규칙은 다음과 같다.
  + 기울임 표시는 나중에 지원할 내용이거나, 지원하지 않을 내용을 명확하게 표시할 때 사용한다.
  + 취소선 : 기본 형상 확정 이후 단계에서 특정 요구 사항을 제외(spec-out)하면 취소선을 사용한다.
  + 기능 요구 사항의 추가, 수정, 삭제가 발생하면 문서를 업데이트 할 때 [버전][날짜][작업자][작업항목 및 변경 이유] 형식으로 로그를 남긴다.

## **1.4 용어 정의 및 약어**

본 문서에서 자주 사용하는 용어 및 약어는 다음 표를 참고한다. 용어/약어가 너무 많아 이곳에 기술하는 것이 부적절한 경우 별도의 장으로 나누어 본 문서의 마지막 장에 서술한다.

표 1-1 문서에서 사용하는 관련 용어/약어

| 용어/약어 | 의미 |
| --- | --- |
| SRS | 소프트웨어 요구 사항 명세서(Software Requirement Specification) |
| SDS | 소프트웨어 설계 명세서(Software Design Specification) |
| IRS | 인터페이스 요구 사항 명세서(Interface Requirement Specification) |
| TDS | 테스트 설계서(Test Design Specification) |
| TCL | 테스트 케이스 리스트(Test Case List) |
| 화면설계서 | 사용자 인터페이스 설계서, 또는 Story board |
| 디자인 스타일 가이드 | 사용자 인터페이스에 적용할 디자인 스타일 가이드 |
| PP | 제품 기획자(Product Planner) |
| PM | 프로젝트 관리자(Project Manager) |
| PE | 개발자(Program Engineer) |
| MK | 마케터(MarKeter) |
| QAE | 품질보증 담당자(Quality Assurance Engineer) |
| TW | 기술문서 담당자(Technical Writer) |
| UX | 사용자 경험 & 제품 디자인 (User eXperience) |

표 1-2 제품/기술/서비스 용어/약어

| 용어/약어 | 의미 |
| --- | --- |
|  |  |
|  |  |
|  |  |

## **1.5 관련 문서**

SRS 산출에 사용된 문서의 저장 경로는 다음과 같다.

(다음은 콘텐츠 예시임)

* SRS: hxxps://github/

## **1.6 대상 독자 및 역할별 체크리스트**

본 문서의 독자는 자신의 역할에 따라 다음 체크리스트를 기준으로 문서를 읽는다.

* 지도교수 / 멘토 ( 지도교수와 멘토는 프로젝트 진행과 관련하여 다음 사항을 확인한다. )

설계와 구현 상 제약조건을 모두 명시하였는가?

사용자 프로파일이 실제 사용자를 대표하는가?

학생들의 역량을 고려하여 주어진 기간 안에 구현 가능한 기술인가?

기술적으로 구현 가능한 기능인가?

프로젝트 기술에 대한 가정과 종속 관계를 모두 명시하였는가?

주요 기능이 제품 기능과 일치하는가?

제품 기능 기술이 구체적이고 명확한가?

제품 기능의 세부 기능에 부여된 우선 순위가 타당한가?

성능 목표가 계량 가능한 수치로 표시되어 있는가?

기본 형상의 선언 시점에 [TBD]가 있지는 않은가?

사용하는 오픈소스는 모두 기재되어 있는가? 라이선스 위배사항은 없는가?

문서와 소스코드 관리를github를 통해 관리하는가? 관리는 정확한가?

제품 기능에 사용자 입장에서 필요한 인터페이스와 입력 값/유효범위가 기술되어 있는가?

지적재산권 확보가 가능한 기술인가?

## **1.7 프로젝트 결과물**

### **1.7.1 결과물 명칭(가칭) 및 버전**

결과물 명칭(가칭) 및 타겟 버전을 서술한다.

### **1.7.2 지적재산권**

본 프로젝트와 관련하여 지적재산권화 되어야 할 주요 기술이 있다면 해당 기술에 대해서 간략히 기술한다.

본 프로젝트에서 사용하는 기술 중 타사에서 이미 특허화 된 내용이 있다면 특허의 명칭, 출원번호, 출원인의 해당 특허 내용을 간단히 기술한다.

본 프로젝트에서 사용하는 기술 중 본사에서 출원된 특허(또는 디자인)에 기반한 부분은 해당 특허(또는 디자인) 내용을 명칭, 출원번호, 특허(또는 디자인) 내용을 간단히 서술한다.

### **1.7.3.1 예시 : 특허 명칭(가칭) 1**

출원자

예시) 컴퓨터정보공학과 4학년 OOO

기술개요

기술개요에 대하여 설명한다.

# 2. 전체 설명

## 2.1 제품 조망

홈 IoT는 사용자가 가정에서 LED, 가스밸브, 가전기기, 난방, 커튼을 스마트폰 어플리케이션을 통해 사용자가 원격으로 상태를 확인/제어할 수 있도록 하여 편의성을 제공하는 제품이다.

## 2.2 시스템 구성

LED, 가스밸브, 가전기기, 난방의 상태를 각각의 센서를 통하여 아두이노 우노 보드에 정보를 받고, 아두이노 우노 보드에서 라즈베리파이에 센서를 통해 받아온 정보를 전송한다. 그리고 라즈베리파이는 사용자가 스마트폰 어플리케이션에서 상태를 확인하고 제어할 수 있도록 하는 역할을 수행합니다.

## 2.3 동작 방식

파이썬 플라스크를 이용하여 만든 웹서버를 통해 스마트폰이 서버에 접근하고, 어플리케이션에서 버튼을 조작하면 데이터를 MongoDB에 저장하여 LED, 난방, 가스밸브, 전자제품을 스마트폰으로 제어가 가능합니다. 아두이노 메가를 이용하여 아두이노 우노들을 컨트롤하고 아두이노 메가가 라즈베리파이(서버)에 연결됩니다. 아두이노 우노들은 각각의 센서와 연결하여 가스밸브제어, 난방제어, LED제어, 가전제품제어에 사용됩니다.

## 2.4 주요 기능

사용자는 안드로이드 기반 SHC(Smart Home Care) 어플리케이션을 통하여 집 외부에서 집에 있는 장치들을 모니터링 및 제어할 수 있다.

## 2.5 사용자 프로파일

2.5.1 프로파일1: 30 ~ 40대 주부

본 프로그램의 주요 사용자는 집에 있는 기기를 사용할 일이 많은 주부가 될 것이다.

1. 가스밸브, 전자기기, 난방 등 집안의 기기들을 사용하는 빈도가 높다.

2. 간단한 어플리케이션 조작을 쉽게 할 수 있다.

3. 집에 아무도 없을 때 제어할 일이 생긴다.

2.5.2 프로파일2: 1인가정 30대 직장인

1. 모바일 어플리케이션에 대한 이해도가 높다.

2. 이들은 집 안에서 보내는 시간보다 외출하는 시간이 길다.

3. 집안기기를 관리할 시간이 부족한 경우가 많다.

## 2.6 설계와 구현상 제약 조건

* 반드시 사용해야 할 기술

안드로이드, 서버, 라즈베리파이, 아두이노 간 통신

* 사용하지 말아야 할 기술
* 설계 도구
* 개발 도구

라즈베리파이, 아두이노, 파이참

* 개발 언어

파이썬, 코틀린, 자바, 아두이노

* 데이터베이스

MongoDB

* 서버

Flask

* 준수해야할 개발 규칙(프로그래밍 가이드, 오류코드 가이드)

함수의 이름의 단어 첫 알파벳은 대문자, 단어가 두개 이상인 경우 단어 사이에 \_(언더바)사용.

## 2.7 가정과 종속 관계

본 프로젝트를 수행하기 위해서 필요하거나, 반드시 수행 또는 결정되어야 할 전제조건 또는 선행되어야 할 프로젝트에 대해 기술하여, 그 결과에 따라 본 프로젝트의 어떤 부분에 어떻게 영향을 미칠지를 서술한다.

또한, 통제 불가능한 외부요소에 의해 영향을 받을 수 있는 경우, 그 요소에 대해 서술한다.

# 3. 환경

## 3.1 설치 환경

본 프로젝트의 결과물을 설치하기 위한 하드웨어, 소프트웨어, 네트워크 환경을 기술한다.

제품의 설치삭제와 빌드시스템, 마스터의 구성과 설치파일 목록 등에 대한 보다 자세한 사항을 기술해야 할 필요성이 있는 경우 설치요구명세서를 별도로 작성할 것을 권장한다.

설치요구명세는 샘플과 템플릿을 이용한다.

### 3.1.1 하드웨어 환경

결과물 설치에 필요한 하드웨어 최소 사양, 권장 사양을 기술한다. 산출물을 여러 시스템에 분리되어 설치해야 하는 경우, 각 설치 대상 시스템 별로 사양을 서술한다.

만일, 산출물이 IoT 기기에 설치되는 경우, 해당 기기의 하드웨어 사양을 서술한다.

참고로, 명시가 필요한 항목은 아래와 같다.

|  |
| --- |
| 장비 모델명(가칭) |
| CPU 사용(CPU종류, Clock, Core수, CPU개수) |
| Memory 용량 |
| Flash Disk 용량(Flash 사용시) |
| NIC 구성(Media 종류, 개수) |
| LCD 사양(LCD 구성 시) |
| 시리얼 포트 여부 |
| Bypass 사양(NIC 종류별 Bypass 지원 여부) |
| Power 사양(Dual/Single 여부) |

### 3.1.2 소프트웨어 환경

#### 3.1.2.1 운영체제

여기에 명시되는 운영체제는 개발 시에도 중요하지만, 테스팅 단계에서 모두 테스트해야 하므로 명확하게 작성한다. 다음과 같은 요소를 고려해서 작성한다.

* 플랫폼(Windows, Linux, Unix, Android, iOS 등)
* OS 버전 & 에디션(여러 개의 에디션이 존재한다면)
* 서비스 팩 또는 패치 요구 사항

#### 3.1.2.2 기타 소프트웨어

OS 외에 지원 또는 제한되는 소프트웨어를 명시한다. 예를 들면 DBMS나 웹 브라우저 같은 것들이 고려되어야 하는 경우가 많다. 물론 이름과 버전, 서비스팩, 패치 요구 사항을 정확하게 적어야 한다.

### 3.1.3 네트워크 환경

프로젝트 산출물이 네트워크 연결이 필요한지 명시한다. 어느 시스템(또는 네트워크)과 어떤 통신을 수행하는지는 통신 인터페이스에서 기술한다.

## 3.2 제품 설치 및 설정

본 프로젝트의 산출물의 설치나 설정 과정에서 필요한 요구사항을 기술한다. 예를 들어, DBMS나 .NET 프레임워크를 먼저 설치해야 하는 제품이 있을 수 있다. 설치에 필요한 다른 어플리케이션이나 라이브러리 파일을 프로젝트 산출물과 함께 배포하기로 결정했으면, 마스터 구성에 해당 애플리케이션과 라이브러리 파일 목록을 작성한다.

또한 제품을 실행하는데 필요한 기본 설정 요소 및 방법에 대한 요구사항을 기술한다. 설치나 설정 변경을 위해 별도의 툴이 필요한 경우에는 그 툴에 대한 부분도 명시한다.

## 3.3 배포

### 3.3.1 패키지 구성

#### 3.3.1.1 예시) 마스터 CD

#### 3.3.1.2 예시) 온라인 설치 패키지

온라인 설치 패키지는 마스터 CD 구성 형식과 다를 수 있다. 설치 프로그램만 제공하는 경우가 많기 때문에 매뉴얼과 기타 유틸리티의 제공 방법(예: '학과 홈페이지에서 로그인 후 제품 구매 내역에서 PDF 매뉴얼을 제공한다.' 또는 '마스터 CD와 동일한 구성물을 압축 파일로 다운로드')을 기술해야 한다.

### 3.3.3 배포 방법

본 프로젝트의 산출물 마스터를 어떤 방법으로 배포할 것인지를 서술한다. 예를 들어, CD로 배포하거나, 웹페이지, 또는 App Store 에서만 다운로드 받게 할 수 있다.

### 3.3.4 패치 및 업데이트 방법

배포 후 제공할 제품의 패치나 업데이트(엔진이나 시그니처를 사용하는 제품의 경우) 종류와 방법을 명시한다. 스마트 업데이트를 사용해야 하는지 여부와 수동 패치를 위한 설치본 제공 필요 여부 등도 모두 포함된다.

## 3.4 형상관리

### 3.4.1 산출물 위치

#### 3.4.1.1 소스코드

소스코드는 다음 경로에 저장한다.

* 예시: http://github/project...

#### 3.4.1.2 문서

문서 소스는 다음 경로에 저장한다.

|  |  |
| --- | --- |
| 문서 종류 | 위치 |
| 개발문서 | http://github/project/dev/ |
| UX문서 | http://github/project/ux/ |
| QA문서 | http://github/project/qa/ |
| TW문서 | http://github/project/tw/ |

### 3.4.2 빌드 환경

소스의 빌드를 위한 시스템 환경, 빌드 툴, 컴파일러 등에 대해서 명시한다.

## 3.5 결함관리

이슈관리를 위해 사용할 시스템 및 이슈 트래킹 프로세를 별도로 명시한다.

# 4. 인터페이스

## 4.1 하드웨어 인터페이스

필요한 경우 하드웨어 종속적인 요구 사항들을 기술한다. 사용자가 하드웨어(이하 장비)에 접근하기 위해 제공되는 물리적인 인터페이스로서 NIC(Network Interface Card), Serial 포트, LCD, LED 등 인터페이스를 모두 기술한다.

## 4.2 소프트웨어 인터페이스

## 4.3 통신 인터페이스

### 4.3.1 예시) 관리자용 통신 인터페이스

### 4.3.2 예시) 일반 사용자용 통신 인터페이스

# 5. 비-기능 요구 사항

## 5.1 성능

제품이 달성해야 하는 성능 요구 사항 및 성능 목표를 기술한다. 성능 테스트의 Pass/Fail 여부를 판단하는 근거가 되므로, 반드시 정량적으로 측정할 수 있도록 기술한다.

성능 측정 시 사용할 하드웨어 사양을 반드시 기재한다.

성능 측정 시 사용할 소프트웨어 종류와 설정 값을 함께 기술한다.

아래 성능 측정 항목은 프로젝트에 따라 선택 적용하거나 추가할 수 있다.

* 처리량(Throughput)
* 동시 접속 수(Concurrent Session)
* 응답 시간(Response Time)
* 리소스 사용량(Memory/CPU/Disk/Network Usage)
* 검사 시간(Scan Speed)
* 부팅 시간(Boot Time)

### 5.1.1 예시) 성능 시험 환경

|  |  |  |
| --- | --- | --- |
| 대상 | 내용 | 비고 |
| Hardware | Intel Core2 Duo E6300  DDR3 1G  HDD 1TB 7200RPM |  |
| OS | Windows 7 Professional SP1 |  |
| Software |  |  |
| Option |  |  |

### 5.1.2 예시) 성능 시험 성공/실패 기준

아래 성능 시험 실패 기준은 각 프로젝트의 성격에 맞는 성능 목표를 설정한다.

|  |  |  |
| --- | --- | --- |
| 대상 | 내용 | 비고 |
| Idle 상태 | CPU : 2% 미만의 CPU 사용량 Pass | (Committed Byte 기준) |
| 제품 동작 기능 | 대상 :  CPU : 30% 미만  Memory : 150MB 미만  Network : 네트워크 트래픽 분당 1M 미만 증가  Disk : 디스크 복사 기간이 미설치보다 5% 미만 저하  Boot Time : 제품 설치 후 부팅시간이 10% 미만 증가 | 옵션 별 다른 성공/실패 기준 적용 명시한다. |

## 5.2 보안

제품이 요구하는 보안성 기준치 요구 사항을 기술한다. 다음의 체크리스트를 활용한다.

### 5.2.1 보안성

최소한의 보안성 요구사항으로 아래 위치의 체크리스트 점검을 본 SRS에 첨부한다.

### 5.2.2 암호화

사용 가능한 암호 알고리즘을 확인한다.

### 5.2.3 취약성

최소한의 제품군 별 취약성으로 다음의 체크리스트가 활용한다.

## 5.3 국제화 & 현지화(I18N, L12N)

국제화(Internationalization, I18N)와 현지화(Localization, L12N)에 대해서 서술한다.

단일 국가나 지역에서만 사용되는 언어가 아닐 수 있으므로 여러 지역에서 사용되는 언어인 경우 지역을 함께 명시하는 것도 좋은 방법이다.

### 5.3.1 인코딩

특별한 이유가 없다면 언어적인 제약을 피하기 위해 유니코드(UTF-8)를 사용한다.

* Message 및 Text 인코딩 : UTF-8
* Log 인코딩 : UTF-8
* DataBase 인코딩 : UTF-8

### 5.3.2 언어 지원 계획

프로젝트 산출물을 판매 대상 시장에 따라 다국어 작업을 병행하거나, 기준이 될 언어를 지정한다. 여기엔 다음과 같은 내용이 기술되어야 한다.

* 기준 언어(현지화 작업의 기준이 되는 언어): 한국어, 영어, 중국어 등에서 하나만 선택

### 5.3.3 로케일

지원하는 로케일을 기술한다.

* 지원 로케일: OS의 언어 로케일을 따름

## 5.4 오픈 소스 라이선스

공개 소프트웨어(Open Source) 사용 여부와 라이선스 종류 및 중복 등에 대해 서술한다. 라이선스 회피 방법 및 소스 공개 범위와 방법에 대해서도 서술한다.

표 5-1 사용할 오픈 소스 라이선스

|  |  |  |
| --- | --- | --- |
| 제품/라이브러리 명칭 | 라이선스 유형 | 참고 |
| BusyBox | GPL v2.0 |  |
| Linux kernel | GPL v2.0 |  |

## 5.4.1 예시) BusyBox

펌웨어를 경량화하기 위해 BusyBox를 사용한다. BusyBox는 백그라운드에서 동작하여 명령어를 처리할 때 호출된다.

#### 5.4.1.1 공개 범위

GPL v2.0 를 준수하고 있어 소스 공개를 피할 수 없다. 공개하는 범위는 BusyBox 원 소스에 수정을 가할 수 있으므로 변경된 부분만 공개한다.

#### 5.4.1.2 공개 방법

소스 공개 요청을 최소화하기 위해 다음과 같은 방법을 사용한다.

* 메일에 의한 공개 요청 접수
* 사용한 오픈 소스 정보와 공개 요청을 접수할 메일 주소를 매뉴얼/도움말에 기재한다.

# 6. 제품 기능

주요기능에서 설명되었던 제품 주요 기능을 상세하게 분류하고, 설명한다. 상세화 수준은 작업량이 1~2주 정도로 산정 가능하도록 작성한다.

각 기능에는 입력과 출력, 입력 가능한 기본값과 유효범위, 기본상태가 기술되어야 한다.

* 6.1 IoT제품 기능

6.1.1 LED전등 제어

어플리케이션에 LED 점멸상태를 전송하여 사용자가 점멸상태를 확인하고 원격으로 점등/소등을 제어한다.

6.1.2 가스밸브 제어

어플리케이션에 가스밸브의 열림/닫침 상태를 전송하여 사용자가 상태를 확인하고 원격으로 열림/닫침을 제어한다.

6.1.3 가전기기 전원 제어

어플리케이션에 가전기기의 전원 상태를 전송하여 사용자가 원격으로 전원을 제어한다.

6.1.4 난방 제어

어플리케이션을 통해 난방 ON/OFF 상태와 난방 온도 및 실내 온도를 확인 할 수 있으며, 난방 ON/OFF 및 원하는 온도 제어를 할 수 있다.

* 6.2 서버 통신

6.2.1 안드로이드

6.2.1.1 IoT 스마트홈 연결

스마트홈 제어를 위해 서버와 연결하는 작업.

IP, Port번호를 입력 받아 서버와 연결

6.2.1.2 전등 Activity

각 방의 LED 전등 점멸 상태 확인

각 방의 LED 전등 제어

일괄 소등/점등 기능

6.2.1.3 난방 Activity

실내 온도, 난방 온도 확인

난방 ON/OFF 상태 확인 및 제어

난방 온도 제어

6.2.1.4 가스밸브 Activity

가스밸브 상태 확인

가스밸브 ON/OFF/AUTO 제어

(AUTO: 설정한 시간이 지난 후 자동 OFF)

6.2.1.5 전자제품 Activity

전자제품 상태 확인 및 ON/OFF/AUTO 제어

(AUTO: 전자 제품의 일정거리 내로 사용자가 접근할 경우 ON)

6.2.2 서버

6.2.2.1 웹 서버

어플리케이션과 서버는 파이썬 플라스크로 만들어진 웹 소켓을 통해 통신한다. 어플리케이션에서 전송하는 값을 데이터베이스에 저장한 후 필요로 하는 값을 어플리케이션에 전송한다.

6.2.2.2 블루투스

IoT제품과 서버는 블루투스로 통신하게 된다. 블루투스를 통하여 센서에서 얻어낸 값을 서버에 전송한다.