Journal of The Korea Society of Computer and Information
Vol. 22 No. 12, pp. 1-7, December 2017

www.ksci.re.kr
https://doi.org/10.9708/jksci.2017.22.12.001

# Design and Implementation of Smart Home based on openHAB

Jeong-Won Kim*, Young-Ju Kim**

## Abstract

More new devices and technologies are recently adapted on homes to aim at enhancing our lifestyle. But though they are not interconnected each other but controlled separately. To solve this problem, we have designed and implemented a common prototype based on openHAB and RaspberryPi they could speak to each other to create a really automated and smart environment at home. The proposed prototype can merge the existing variable devices, add and remove new features at runtime because of its modular design. The proposed prototype based on a low-cost platform showed its potential as a smart home and provide a new UX to users.

▸Keyword: Smart Home, Internet of Things, openHAB, Raspberry Pi

## I. Introduction

최근 각 가정에는 새로운 기술의 장치들이 속속 등장하고 있다. 특히 스마트 홈 시스템의 경우 다양한 솔루션들이 개발되어 실생활에서 적용되고 있으며 최근에는 IoT(Internet of Things) 형태의 장치들도 등장하고 있다. 이 시스템들은 제조사의 의도에 맞게 훌륭하게 동작 하지만 문제는 사용자가 아닌 제조사가 정의한 유스케이스(use case)에 의해 설정되고 운용된다는 것이다. 이들 장치들이 사람의 생활 방식의 변화를 요구하지만 장치들 간의 인터페이스나 지능적 환경을 제공할 공통적인 수단이 부족하다. 따라서 사용자 중심의 스마트 홈 시스템이 필요하다. 본 논문에서는 이러한 점을 고려하여 사용자에 초점을 두어 사용자의 유스 케이스 최적화된 IoT 기반의 스마트 홈 시스템을 제안하고자 한다.

본 논문에서는 다양한 장치에 공통적인 인터페이스를 제공하기 위하여 openHAB2[1] 플랫폼을 적용한다. openHAB2 는 다양한 제조사 장비간의 공통 연결 수단을 위한 통합 플랫폼을 제공한다. openHAB2 는 이질적인 가정 자동화 시스템을 통합하여 단일 사용자 인터페이스를 제공하는 소프트웨어이다. 구체적인 특징은 다음과 같다. openHAB2 는 벤더 중립적이며 다양한 하드웨어 및 프로토콜을 지원하고 JVM 이 동작하는 모든 장치에서 서비스 가능하다. 또한 사용자의 취향에 최적화된 서비스를 지원하기 위한 규칙 엔진을 제공하며 웹 기반 UI 뿐만 아니라 iOS 및 안드로이드용 네이티브 UI 를 지원하며 오프 소스 기반으로 제작되어 개방성을 지원한다. 그리고 새로운 장치에 쉽게 연결되고 통합될 수 있는 API 를 제공하며 확장성 또한 우수하다.

또한 본 연구에서는 스마트 홈을 위해 별도의 서버가 아닌 저가의 공개 플랫폼인 라즈베리파이에 openHAB2 를 탑재하는데 라즈베리파이는 35달러 정도의 개인용 컴퓨터에 근접하는 초소형 장치로서 신용 카드 크기이며 리눅스 계열의 운영체제가 탑재된다. USB 포트에 키보드, 마우스 등 다양한 입출력 장치가 탑재되며 HDMI 포트를 통해 LCD, TV 등 다양한 디스플레이를 연결할 수 있으며 최근 모델 3 의 경우 와이파이 어댑터가 기본 장착되어 각 가정에 보편적으로 설치되어 있는 무선 랜에 연결하여 외부에서도 스마트 홈에 접속할 수 있는 IoT 환경을 제공한다.

이미 스마트 홈에 대한 많은 연구가 있어 왔으나 장치간의 상호 연결성에 대한 제공이 부족하므로 본 연구에서는 홈 서버 하드웨어로 라즈베리파이를 사용하고 장치간 연결성을 지원하는 미들웨어로서 openHAB2 를 이용하여 스마트 홈 프로토타입을 제안하고자 한다. 지금까지 제시되고 있는 많은 스마트 홈 기술들은 각각 고유의 환경에서는 만족할 만한 성능을 제시하지만 각각의 환경을 벗어난 사용자의 요구를 수용하기가 힘들며 업그레이드 또한 불편한 것이 사실이다. 따라서 본 연구에서 제안하는 프로토타입은 모듈화된 구조로서 실행 중에 장치를 제거하거나 교체가 가능하고 사용자 스스로 홈 환경을 디자인하는 규칙을 제공하도록 하여 사용자 친화

• First Author: Jeong-Won Kim, Corresponding Author: Young-Ju Kim
*Jeong-Won Kim (jwkim@silla.ac.kr), Dept. of Computer Software and Engineering, Silla University
**Young-Ju Kim (yjkim@silla.ac.kr), Dept. of Computer Software and Engineering, Silla University
• Received: 2017. 08. 22, Revised: 2017. 09. 11, Accepted: 2017. 11. 29.

적 스마트 홈 환경을 구성할 수 있다. 또한 태양광추적기를 통한 태양광 발전량을 클라우드 서비스하여 진정한 스마트 홈 서비스를 제공한다.

논문의 구성은 다음과 같다. 먼저 2장에서는 스마트 홈에 관련된 최근 연구와 본 연구를 비교한다. 3장에서는 openHAB2 에 기반한 스마트 홈의 구조 및 설계에 대한 상세한 내용을 소개하고 4장에서는 프로토타입의 실험 결과를 제시한다. 마지막으로 5장에서는 결론과 향후 연구 내용을 설명한다.

## II. Preliminaries

스마트 홈에 대한 연구가 활발히 이루어짐에 따라 다양한 스마트 홈 모델이 제안되어 왔는데 주요 특징으로는 가정에서 사용되는 다양한 장치를 자동화하기 위해 각 연구가 제안하는 모델을 서버에 구축하는 것이다[2]. 몇몇 모델들은 가정의 전력 관리를 위해 홈 게이트웨이를 각 가정에 구축하거나 스마트 홈을 클라우드에 연결하기도 한다[3, 4, 5]. 스마트 홈에서 장치를 연결하는 다양한 방법이 있는데 유선은 최초 건축시 설계되어 변경에는 비용 부담이 있고 오래된 건축물에는 구현하기가 더 곤란하다. 따라서 무선 방식이 선호되는데 와이파이(WiFi), 블루투스가 일반적이며 지그비(ZigBee)는 저가, 저 전력, 그리고 구현의 용이성으로 인해 인기 있는 모델이다[6, 7]. 스마트 홈 연구에서 가정의 조명을 효율적으로 관리하는 연구가 다수 이루어졌는데 조도의 강도를 조정하기 위해서 사용자의 수동적 입력에 의존하는 방법[8], 또는 센서 네트워크를 이용하는 방법[9, 10] 등이 있는데 이러한 연구들은 지능적 조도 관리를 위해서 센서 네트워크를 대부분 이용하고 있는데 이는 구축 시 비용이 높아지는 단점이 있다[11]. 지그비, 블루투스 등의 기술이 스마트홈에 활용되고 있으나 최근 와이파이는 IPTV, IoT 등의 환경 변화로 각 가정에 대부분 설치되어 있는 네트워크로 스마트 홈에서 활용성이 매우 높다[12]. 따라서 본 연구에서는 와이파이를 스마트 홈에 활용하여 IoT 환경에 최적화된 모델을 제시하고자 한다. 또한 기존의 연구들이 제안하는 모델의 유스 케이스에 정형화되어 있어 새로운 유스 케이스 의 적용이 곤란한 경우가 많다. 제안하는 모델의 차별성은 모듈화 구조로 인하여 디바이스의 삭제 및 추가가 자유로우며 사용자 정의형 유스 케이스가 자유롭게 적용될 수 있어 진정한 의미의 스마트 홈 서비스 경험을 제공할 수 있다는 것이다.

## III. Smart Home implementation based on openHAB2 and Raspberry Pi

openHAB2 는 기존 솔루션을 대체하는 것이 아니라 이질적인 시스템을 통합하고 강화하는 시스템의 시스템이다. 따라서 openHAB2 에 소속되는 하위 시스템은 독립적으로 설정되는 것을 기본으로 하여 프로세스나 장치를 연결해야 하는 매우 복잡한 작업을 요구한다. 이 복잡한 과정을 openHAB2 는 추상화 기법을 통해 해결한다. 핵심 개념은 데이터 중심 인터페이스를 제공하는 일명 '아이템'(item)으로서 물리적 장치나 웹 서비스와 같은 가상 자원, 그리고 감지기가 출력하는 계산 값 등이 될 수 있다. 따라서 스마트 홈에서 특정 장치, 예를 들면 IP 주소나 센서의 ID 등을 정의할 필요 없이 아이템이라는 추상화 기능만 정의하면 되는 것이다. 이를 통해 특정 장치나 서비스를 교체할 경우 UI 나 규칙을 변경할 필요가 없으므로 시스템의 유지 보수, 관리, 확장성이 매우 높은 장점이 있다. 또한 openHAB2 시스템 자체가 모듈화 디자인 기반이므로 새로운 시스템을 바인딩이라는 개념으로 쉽게 연결할 수 있고 실행시간에 특정 기능을 제거할 수 있어 다양한 종류의 장치나 시스템을 적용할 수 있다.

### 3.1 Prototype implementation

프로토타입 구현에 있어서 openHAB은 실행속도와 안정성에서 앞서는 면이 있었으나 그 차이가 두드러지지 않고, openHAB2 가 이클립스 스마트 홈 프레임워크를 기반으로 구현되어 규모가 커진 측면이 있고 확장성이 높으며 사용자 인터페이스도 우수하며 안정성도 검증이 되어 openHAB2 버전을 적용하였다.

본 연구에서 개발하는 각종 IoT 단말장치와 OpenHAB2 게이트웨이 소프트웨어가 MQTT(Message Queue Telemetry Transport) 프로토콜 기반으로 통신하는데 이를 위해 기존의 MQTT 바인딩 모듈을 일부 수정하여 OpenHAB2 소프트웨어에 적용하였다. 개발된 장치들을 실제 환경에 적용하여 테스트를 진행하여야 하나 현실적인 한계가 있고, 스마트 게이트웨이의 부하 등을 평가하기 위해 추가적인 기능들이 요구되어 스마트 홈 테스트 베드를 구현하였다. 구현된 스마트 홈 테스트 베드는 그림 1과 같이 구성되며 ① 집 모형, ② 천정 IoT 장치, ③ 거실 IoT 장치, ④ 외부 감시 장치, ⑤ 출입문 제어 장치, ⑥ 전력량 측정 장치, ⑦ 배전반 테스트 베드, ⑧ 전원 장치, ⑨ 스마트 게이트웨이 ⑩ 유무선 공유기 등 10개의 모듈로 구성된다. 다음은 주요 부분에 대한 설명이다.

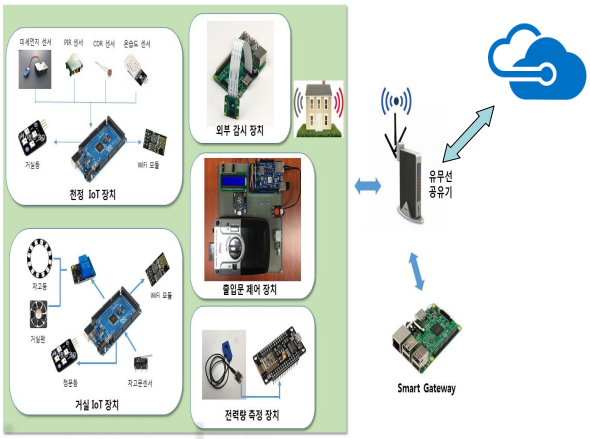


Fig. 1. Smart Home structure

### 3.2 IoT devices of ceiling

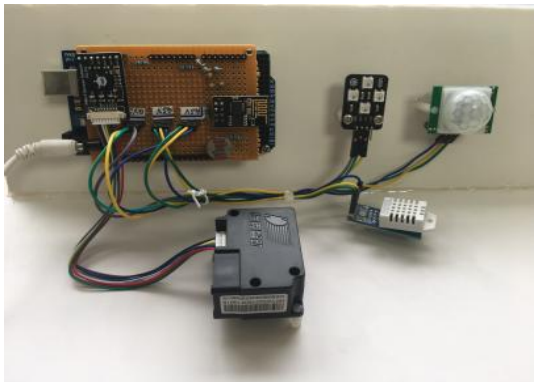

Fig. 2. Ceiling IoT devices

실내 천정에는 미세먼지 센서, PIR 센서, CDR 센서, 온습도 센서, 조명이 아두이노 모듈에 연결되며 와이파이를 통해 게이트웨이에 의해 제어된다. 이 조합이 하나의 방을 구성할 수 있으므로 추가로 설치되면 가정의 모든 실내가 조명, 온습도, 공기 질을 게이트웨이를 통해 모니터링하고 제어할 수 있다(그림 2).

미세먼지의 경우 광학 방식의 센서를 사용하여 PM2.5와 PM10을 동시에 측정한다. 센서와 아두이노는 UART 방식으로 통신하는데 센서로부터 20바이트 데이터를 받으면 4~7 번째 비트(d4~d7)가 PM2.5이고 8~11(d8~d11) 번째 비트가 PM10 이다. 계산식은 아래와 같다.

$$PM2.5(ug/m^3) = d4 \times 256^3 + d5 \times 256^2 + d6 \times 256^1 + d7$$
$$PM10\ (ug/m^3) = d8 \times 256^3 + d9 \times 256^2 + d10 \times 256^1 + d11$$

### 3.3 IoT devices of living room

거실 IoT 장치에는 차고 등, 팬, 정문 등, 차고문 센서 등이 게이트웨이에 연결되어 거실, 차고, 정문의 공기, 조명 등을 제어한다(그림 3).

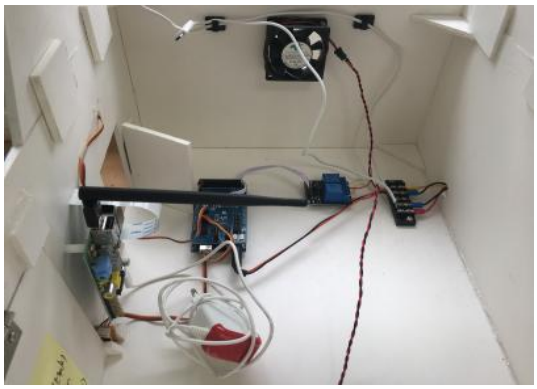

Fig. 3. Living room IoT devices

### 3.4 Exterior monitoring devices

외부 감시 장치는 영상 녹화를 통해 구현하였는데 Pi 카메라로 Mpeg 영상을 사용자 단말로 스트리밍한다. 동영상 스트리밍을 지원하는 경우에 라즈베리파이3 보드의 CPU 및 네트워크 부하가 크게 증가하여 MQTT 메시지를 제때에 처리하지 못하였다. 이에 동영상의 엔코딩 방식을 MJPEG, 프레임 크기는 320x240, 전송 프레임 속도는 초당 5 프레임으로 각각 축소하여 적용함으로써 안정적으로 동작함을 확인하였다(그림 4).

Fig. 4. Outdoor IoT devices

### 3.5 Front door devices

본 연구에서 제안하는 스마트 홈의 기능 중 하나는 외부 감시 장치에 의해 외부인이 집을 방문하면 원격에서 출입문을 제어하고 방문자의 영상을 녹화하고 가족 구성원은 지문에 의해 출입문을 개방하는 기능을 제공하는 것이다. 이를 위해 본 연구에서는 도어락 장치, 지문 센서 모듈, 그리고 아두이노 기반 출입문 제어 장치를 구현하였다(그림 5). 지문인식 모듈은 Fingerprint Scanner Device GT(511C3) 로서 SmackFinger 3.0 알고리즘을 기반으로 하고 ARM Cortex M3 Core CPU를 장착하여 200장의 지문 이미지를 저장할 수 있다. 또한 360 도 방향 전환이 가능하며 통신 프로토콜은 UART 이고 JST-SH 커넥터로 아두이노와 RS-232 시리얼 인터페이스로 호스트 단말과 지문 이미지를 전송한다. 이 모듈은 패킷 핸드쉐이킹으로 통신 프로토콜이 구현되는데 명령 패킷, 응답 패킷, 그리고 데이터 패킷의 세 가지로 구성된다.

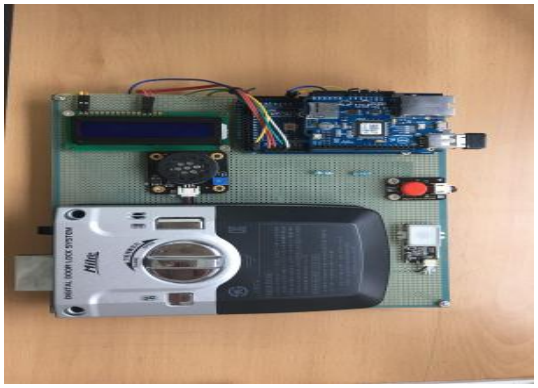

Fig. 5. Front door based on finger print recognition

```
double readACCurrent() {
  1. ADCpeak_average from ADC port
  2. ADCrms = ADCpeak_average x 0.7071
  3. Vamplified_rms = (ADCrms x Vref) / 1024
  4. Vrms = Vamplified_rms / 2
  5. Irms = Vrms x AC_Current_Detection_Range
  6. Irms_calibrated by linear calculation
  7. return Irms_calibrated
}
```

Fig. 6. Power measurement function

### 3.6 Power monitoring device

스마트 홈의 여러 기능 중 전력 제어 서비스가 중요한 부분이므로 본 연구에서도 집안의 전력을 모니터링하고 제어하는 기능을 제공한다. 먼저 전력량의 측정은 전류센서를 부착한 아두이노 미니를 통해 이루어지는데 ADC 포트에 읽은 값을 AC 전기신호에 따라 사인파(Sine Wave)의 절대값 형태로 나타나는데, 우선 노이즈 등이 측정값에 미치는 영향을 줄이기 위해 ADC 포트로부터 32개의 값을 읽어 평균값을 계산한다. 그리고 RMS(Root-Mean-Square) 값을 구하기 위해 0.7071을 곱한 후에 전압 값으로 변환한다. 이때, Vref는 3.3이고, ADC Resolution이 10 비트이므로 1024로 나누어준다. 본 논문에서 사용한 전류 센서는 센서의 전압 출력범위가 0~1.0V로 낮아 출력 신호를 OP-Amp를 사용하여 2배로 증폭하여 사용하므로 원래의 값을 구하기 위해 2로 나누어준다. 최종적으로 산출된 전압 값에 센서의 AC 전류 검출 최대값(20A)을 곱하여 전류 값을 산출한다. 또한 사용한 Vref 값이 권장 사양인 5V와 다르고 ESP8266의 ADC 검출값이 안정적이 못한 경향이 있어 다양한 전류 값에 대해 출력 값을 측정한 후에 회귀분석을 통해 선형 변환 모형을 구하고 이를 사용하여 측정값을 조정하였다. 아래 그림 6은 본 연구에서 사용한 전류 측정 함수의 pseudo code 이고 그림 7은 구현한 전력 측정 모듈이다.

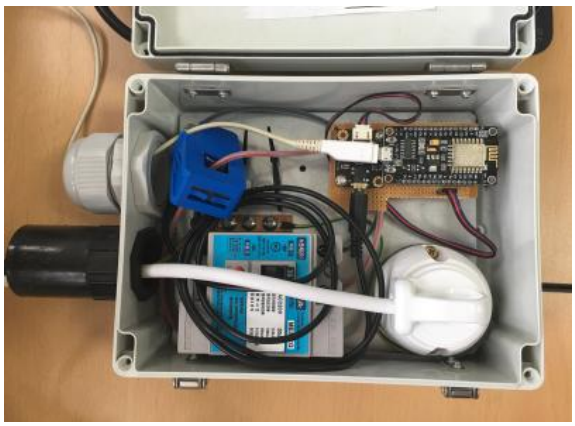

Fig. 7. Power monitoring device

## IV. Experiment results

본 논문의 주요 목적은 저가의 하드웨어와 공개 소프트웨어로 스마트 홈을 구축할 수 있다는 것과 사용자 경험 위주의 진정한 스마트 제어 환경을 제공하는 것이다. 이 목적을 위해 그림 8은 본 논문에서 구현한 스마트 홈의 프로토타입으로서 openHAB2 로 라즈베리파이에 구현한 스마트 홈 게이트웨이, 액세스 포인트, 그리고 각종 센서부로 구성된다. 4장에서는 구현 결과로서 사용자 인터페이스, 스마트 홈 규칙 엔진, 그리고 CPU, 메모리 등의 부하 측정 결과를 소개한다.


Fig. 8. Implemented smart home

### 4.1 User interface

① 사용자 인터페이스 시작 화면 – 모든 사용자 인터페이스가 시작되는 화면으로 현재 날짜가 표시되고 Ground Floor, Outdoor 제어와 현재까지의 사용 전력량을 표시한다(그림 9).

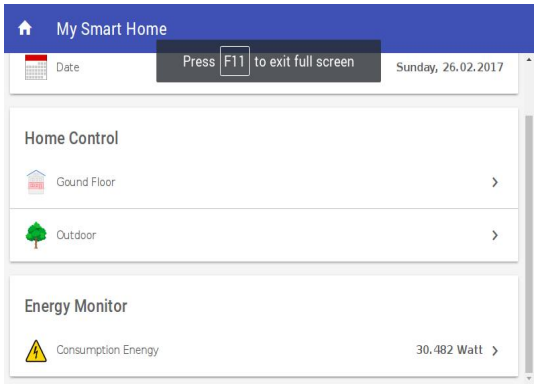


Fig. 9. Main page

② 거실 인터페이스 화면 – 거실의 환경 요인(온습도, 미세먼지)들의 센싱 값들을 확인할 수 있을 뿐만 아니라 거실의 장치, 즉 천장 등, 거실 팬, 현관문들에 대한 제어 메뉴를 표시하여 사용자가 장치를 제어할 수 있도록 하였다. 또한, 미세먼지 상세 표시 화면에 대한 메뉴를 표시한다(그림 10, 그림 11).

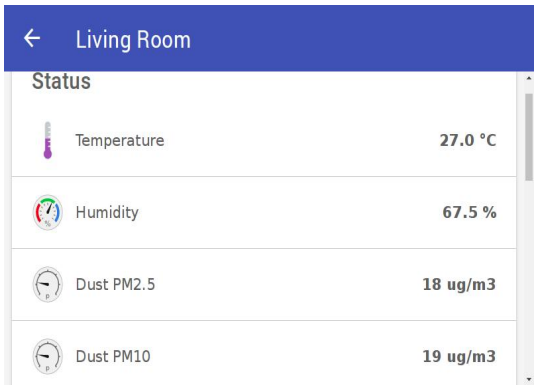


Fig. 10. Living room (environment sensing)

③ 미세먼지 상세 표시 화면 – 스마트 홈 장치가 동작한 이후로 계속 측정된 미세먼지 측정값을 하단에 그래프로 표시하고 최대치와 최소치를 상단에 표시한다. 그래프는 시간, 일, 주 단위로 표시를 선택할 수 있다(그림 12).

④ 전력량 모니터링 화면 – 현재 사용 중인 전류량과 분 단위 사용 전력량을 수치와 그래프로 출력하는 화면이다. 그래프는 시간, 일, 주 단위 출력을 선택할 수 있다(그림 13).

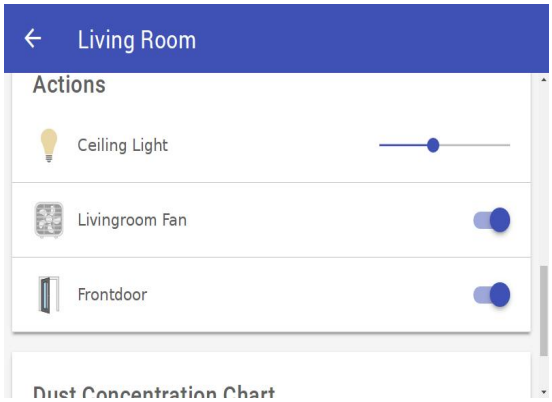


Fig. 11. Living room(ceiling light, Fan, etc)

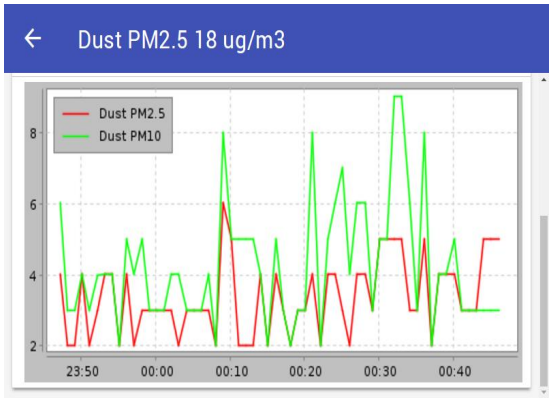


Fig. 12. Fine dust monitoring

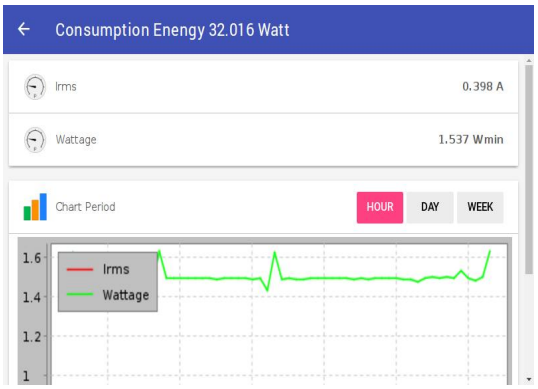


Fig. 13. Power graph

## 4.2 Rule engine

사용자에게 진정한 스마트 홈의 사용자경험을 제공하기 위해서 본 연구에서는 다양한 규칙을 제공한다. 이는 openHAB2에서 'rule engine' 이라고 부르는데 그림 14에서 보듯이 "Lights on at Sunset" 규칙은 해가 질 때 집안의 조명을 조절하고 각종 장치를 제어하는 규칙을 제공한다. 구체적으로 보면 해가 지는 이벤트(sun:local:set#event)가 발생하면 현관문 조명, 현관의 장식 커튼, 그리고 거실의 조명이 켜진다. 또한 사용자는 여기에 다양한 반응을 할 수 있는데 VoiceRSS 를 바인딩하여 사용자에게 TTS(Text to Speech) 로 거실벽의 스위치가 있음을 알리거나 저녁 준비를 하라든지 조용히 있는 경우가 많으면 불 끄는 음성 메시지를 플레이한다.

```
rule "Lights on at Sunset"
when
  channel 'sun:local:set#event' triggered START
then
  if (LightsOnAtSunset.state==ON) {
    FrontDoorLight.sendCommand(ON)
    FrontDoorDecorations.sendCommand(ON)
    LivingRoomLight.sendCommand(ON)
    switch
     ThreadLocalRandom::current().nextInt(0,3+1):
    { case 0: say("거실벽에 스위치가 있습니다.");
      case 1: say("어두워지고 있어요");
      case 2: say("저녁 준비할 시간입니다");
      case 3: say("불 끌까요");
     }
  }
end
```

Fig. 14. UX example

## 4.3 Load analysis

스마트 홈을 구현하는 데는 홈 서버나, 전용 임베디드 장치 등 다양한 방법이 있는데 본 연구의 목적이 공개 플랫폼으로 스마트 홈의 구현 가능성을 확인하는 것이므로 구현한 시스템의 모든 기능이 동작할 때 라즈베리파이의 CPU 부하와 메모리 사용량을 측정하였다.

Table 1. Shell Script for Performance Evaluation

| parameter | shell script |
|---|---|
| RAM | free -h \| grep -v +> /tmp/memoryUsage |
| Load average | loadaverage=$(top -n 1 -b \| grep "load average:" \| awk '{print $10 $11 $12}') |
| System uptime | tecuptime=$(uptime \| awk '{print $3,$4}' \| cut -f1 -d,) |

측정 도구는 메모리는 'free', CPU 부하는 'top', 'uptime' 명령어를 표 1에서 보는 바와 같이 스크립트로 만들어서 1시간 동안 데이터를 축척하였다. 그리고 매 10분마다 현관에 외부인이 왔음을 가정하고 비디오 스트리밍을 2분간 진행하였다.

그림 15에서 보듯이 일반 조명, 팬, 온습도 등 센서를 조작하는 경우는 평균 10% 이하의 CPU 사용률을 보이고 있다. 그러나 비디오 스트리밍을 할 경우 CPU 사용률이 최고 60% 이상 올라가고 있지만 나머지 기능에 영향을 줄 정도는 아닌 것으로 판단된다. 통상 외부인의 식별에 1분 미만의 스트리밍으로 인식이 가능하므로 감내할 수준으로 보인다.

그림 16은 실험 기간 동안의 메모리 변화량을 측정한 것이다. 실험에 사용한 라즈베리파이 3 모델은 메인메모리 용량이 1GB로 스마트 홈이 동작하지 않더라도 평균 65% 정도의 메모리를 점유한다. 실험에서 매 10분 마다 동영상 스트리밍을 하고 있는데 그림에서 보듯이 메모리 사용량이 80% 이상 급격히 증가하고 있다. 이 기간 동안에도 15% 정도 여유 메모리가 있어 서비스하는 데는 큰 무리가 없는 것으로 판단된다.

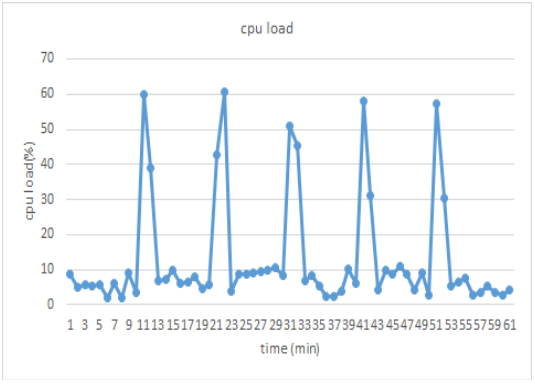


Fig. 15. CPU load (%)

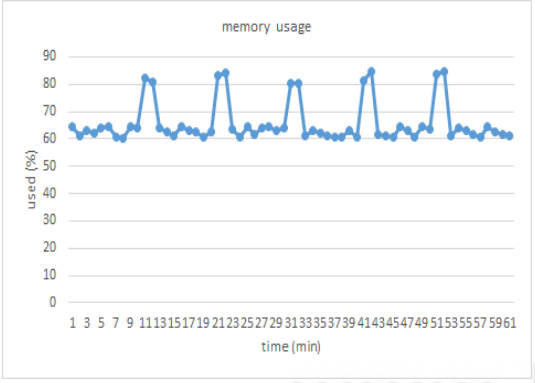


Fig. 16. memory usage (%)

### 4.4 Cloud service

기존의 스마트 홈이 가전 기기의 제어와 모니터링에만 우수한 성능을 제공하고 있는 점을 감안하여 본 연구에서는 태양광 추적을 통한 태양광 발전기를 설치하여 전력 생산량을 클라우드에 올리고 사용자가 발전량을 시간과 장소에 관계없이 모니터링 할 수 있는 시스템을 구현하였다. 구현된 태양광 발전기는 그림 17에서 보는 바와 같이 4개의 조도 센서로 태양광과 패널이 항상 수직을 이루어 발전량을 최대화할 수 있도록 구성하였다.

태양광 발전량 클라우드 서비스는 마이크로소프트 애져(Azure)의 IoT Hub를 사용하여 제공한다. 태양광 발전기는 10초마다 생산한 전력량(watt)을 IoT Hub 에 메시지로 전송하는데 클라우드에서는 IoT Solar 웹앱을 그림 18과 같이 구성하여 실시간 자료를 제공하므로 사용자는 원격에서 전력 생산량을 모니터링 할 수 있다.

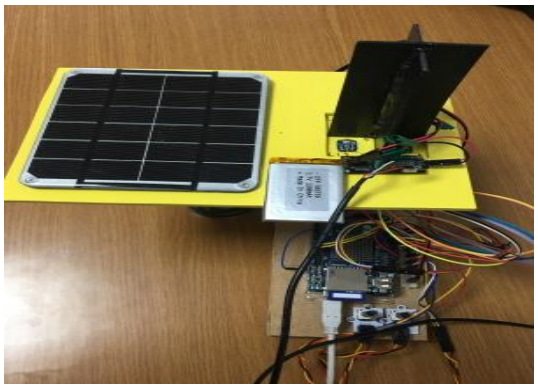

Fig. 17. Solar power with sun tracker

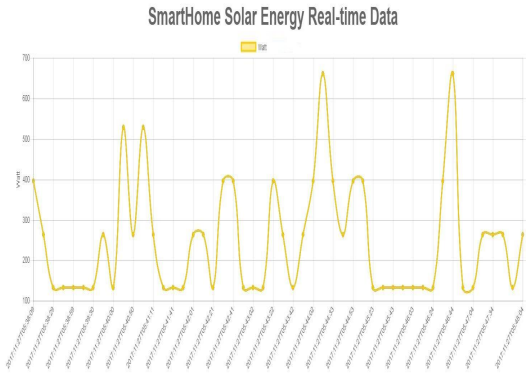


Fig. 18. Solar power monitoring WepApp (http://openhavsmarthome.azurewebsites.net)

## V. Conclusions

본 논문에서는 공개 플랫폼인 openHAB2 와 라즈베리파이를 사용하여 스마트 홈 프로토타입을 구현하였다. 제안하는 스마트 홈은 이질적인 장치들을 통합하는 인터페이스를 제공하며

실시간에 장치를 추가 및 제거할 수 있다. 또한 규칙 엔진을 통하여 사용자 중심의 유스 케이스를 제공하여 사용자는 새로운 UX 를 경험할 수 있다. 구현한 시스템은 CPU, 메모리 부하 실험 결과 저가의 하드웨어로 스마트 홈 서비스를 충분히 제공할 수 있었다. 그리고 태양광 추적을 통한 태양광 발전기를 장착하여 생산한 전력량을 클라우드 서비스로 제공하여 진정한 스마트 홈으로서의 기능을 제안하였다.

한편 본 논문에서 제안하는 모델의 통신은 와이파이를 사용하고 있는데 최근 로라(Lora)가 많은 관심을 받고 있다. 따라서 향후 연구 과제로는 제안하는 스마트 홈 모델에 로라를 적용하여 저전력으로 지속 가능한 통신 방식을 홈 게이트웨이에 적용하고 집안내의 다양한 에너지 사용과 에너지 생산량을 클라우드에 저장하고 머신러닝을 통해 사용량과 생산량을 예측하는 서비스를 개발하는 것이다.

## REFERENCES


[1] openHAB, *https://www.openhab.org/introduction.html*

[2] I. Han, H.S. Park, Y.K. Jeong, K.R. Park, "An integrated home server forcommunication, broadcast reception, and home automation," International Conference on Consumer Electronics, pp. 253-254. 2006.

[3] D.M. Han, J.H. Lim, "Design and implementation of smart home energy management systems based on zigbee," IEEE Trans. Consum. Electron. Vol. 56, No 3. pp. 1417-1425, 2010.

[4] Z. Wei, J. Li, Y. Yang, D. Jia, "A residential gateway architecture based on cloud computing," IEEE International Conference on Software Engineering and Service Sciences, 2010, pp. 245-248, 2010.

[5] D. Valtchev, I. Frankov, "Service gateway architecture for a smart home," IEEE Communication Magine. Vol 40, No. 4, pp. 126-132, 2012.

[6] I. Chew, V. Kalavally, C.P. Tan, J. Parkkinen, "A spectrally tunable smart ledlighting system with closed-loop control," IEEE Sense Journal. Vol. 16, No. 11, pp. 4452-4459, 2016.

[7] D.M. Han, J.H. Lim, "Smart home energy management system using IEEE802.15. 4 and zigbee," IEEE Trans. Consumer Electronics, Vol. 56, No, 3, pp. 1403-1410, 2016.

[8] J. Byun, I. Hong, B. Lee, and S. Park, "Intelligent household led lighting system considering energy efficiency and user satisfaction," IEEE Trans. Consumer Electronics, Vol. 59, No. 1, pp. 70-76, 2013.

[9] Y.K. Tan, T.P. Huynh, and Z. Wang, "Smart personal sensor network control for energy saving in DC grid powered led lighting system," IEEE Trans. Smart Grid, Vol. 4, No. 2, pp. 669-676, 2013.

[10] M.S. Pan, L.W. Yeh, Y.A. Chen, Y.H. Lin, and Y.C. Tseng, "A WSN-basedintelligent light control system considering user activities and profiles," IEEE Sensse Journal, Vol. 8, No. 10, pp. 1710-1721, 2008.

[11] A. Pandharipande, and D. Caicedo, "Adaptive illumination rendering in led lighting systems," IEEE Trans. System Management Cybernetic Systems, Vol. 43, No. 5, pp. 1052-1062, 2013.

[12] K. Gill, S.H. Yang, F. Yao, and X. Lu, "A zigbee-based home automation system," IEEE Trans. Consumer Electronics, Vol. 55, No. 2, pp. 422-430, 2009.


## Authors


Jeong Won Kim received the B.S., M.S. and Ph.D. degrees in Computer Science from Pusan University, Korea, in 1995, 1997 and 2000, respectivelyDr. Kim joined the faculty of the Department of Computer Information Engineering at Silla University, Pusan, Korea, in 2002. He is currently a Professor in the Department of Computer Software and Engineering at Silla University. He is interested in embedded computing, Internet of Things and cloud computing.

Young Ju Kim received the B.S., M.S. and Ph.D. degrees in Computer Science from Pusan University, Korea, in 1990, 1995 and 1999, respectively Dr. Kim joined the faculty of the Division of Computer Information Engineering at Silla University, Pusan, Korea in 2000. He is currently a professor in the Division of Computer Software Engineering at Silla University. He is interested in embedding computing, robot control and cloud computing.