블루투스, 아두이노, 웹

태그

사물인터넷 IoT 통신

스마트홈의 ICT관점의 기술적 구성 요소

(1) 센서(Sensor)

스마트홈은 주택이 스스로 거주자를 둘러싸고 있는 환경의 변화를 감지해서 적절한 서비스를 스스로 제공하는 기능을 수행해야 하므로, 주변 환경 변화를 감지하기 위한 다양한 센서들이 필요함

- 온도·습도·열·가스·조도·초음파 센서부터 원격 감지, 레이더, 위치, 모션, 영상 센서 등
- 유형 사물과 주위 환경으로부터 정보를 얻을 수 있는 물리적인 센서도 포함됨.
- 최근에는 영상, 음향 등의 이종 정보를 기계학습(machine learning)을 통하여 받아들
- 이는, 복합적 상황 인지 능력을 가진 다중 센서 기술들이 개발되고 있음.
- 스마트폰, 웨어러블 기기 등에 가속도계, 자이로스코프와 같은 모션 센서들이 탑재되어 활용되어 왔고, 환경 정보(대기 오염도, 수질, 가스 등), 위생(곰팡이, 박테리아, 바이러스 등), 생체 정보(지문, 홍채, 혈압 등)를 높은 정확도로 측정할 수 있는 센서들이 개발되면서 주택, 도시, 헬스케어 등 다양한 영역에서 활용되고 있음.

센서의 종류

- 화재 센서 : 화재 감지기, 화재 발생 여부 감지, 연기 감지기
- 가스 센서 : 가스 감지기, 가스 누출 탐지
- 방범 센서 : 동체감지기, 사람의 움직임을 감지 / 적외선을 이용한 감지, 문의 열림, 담힘으로 감지, 유리 파손 감지기
- 온도 센서 : 현재 실내 온도를 체크하는 센서
- 검침 센서 : 전기, 수도, 가스, 온수 사용량을 체크하는 센서로 원격 검침에 사용되고 있음
- 지문 인식(생체) 센서 : 출입문 제어, 전자상거래 인증용
- 근접(RF) 센서 : 주출입구 출입 제어 및 차량 출입에 사용
- 조도 센서 : 불 밝기 감지
- 원격 진료용 센서 : 협압, 체온, 심전도 등

```
컬러감지 센서 : 의료분야에서 많이 사용되는 센서
- 밸런스 설정
  white 색상의 파장을 스캔
  5초 후 black 색상의 파장을 스캔

- 색상 감지
  색상의 파장을 스캔

- 화면 출력
  섹싱 김지에서 감지한 파장을 serial monitor에 출력한다.
```

컨트롤러

컨트롤러는 센서에 의하여 감지된 환경 변화나 사용자에 의하여 입력된 명령 등의 각종 정보들을 분석하여 필요한 조치를 확인하고, 특정 기기가 적절하게 작동하도록 명령하고 관리하는 장치임.

인공지능과 같이 스스로 정보를 분석하고 판단해서 조치하는 '인공지능 컨트롤러'와 월패드나 스마트폰, 태블릿 PC, TV, PC등과 같이 사용자가 직접적으로 명령을 입력하는 '컨트롤 디바이스'가 있다.

- 인공지능 컨트롤러
  구글 어시스턴트, 아마존의 알렉사, KT의 기가지니, SK텔레콤의 누구, 카카오의 카카오미니, 네이버의 웨이브(클로바) 등과 같은 인공지능 스피커 또는 스마트홈 허브와 같은 형태로 공급되고 있음

사용자의 명령이 없더라도 일정한 조건이 되면 필요한 조치를 취할 수 있도록 프로그램이 입력되어, 해당 기기에 자동으로 명령을 전달하는 것도 인공지능 컨트롤러에 포함됨.

- 컨트롤 디바이스
  스마트홈 허브의 경우, 월패드나 스마트폰 등과 같은 컨트롤 디바이스를 통한 직접적인 명령입력 방식보다 거주자와 자연어 대화형의 인터페이스를 통해 정보처리를 함으로써 보다 자연스러운 스마트 홈 서비스를 가능하게 한다.

유무선 네트워크

센서에 의하여 취득한 정보나 컨트롤러에 의하여 실행되는 명령이 해당 기기들에 실시간으로 전달되기 위해서는 그에 적합한 유무선 네트워크가 구축되어야 한다.,

- WIFI
  통신 속도가 빠르고 도달 거리가 긴 반면에 전력 사용량이 많기 때문에 크그가 크고 전력 공급량이 많은 가전제품에는 적합하지만, 전구, 가스밸드, 도어락 등과 같은 크기가 작고 저전력을 사용하는 기기에는 적합하지 않음
- 비콘, 지그비, Z-Wave
  근거리, 소용량, 저 전력의 특성을 가진 무선 네트워크 프로토콜들은 실내의 크기가 작고 전력 사용량이 적은 기기들 간의 통신에 적합함.

사용자 인터페이스

사용자와 스마트홈의 접점으로서 사용자의 의지가 스마트홈에 전달되는 중간 매개 방식에 해당함, 사용자 인터페이스는 사용자가 스마트홈에 정보나 명령을 입력하는 부분과 스마트홈이 처리한 결과를 출력하는 부분으로 나눌 수 있음

- 입력방식
  구두 명령과 같은 소리를 인지하는 인공지능 스피커의 마이크, 동작 형태나 사용자 얼굴 등과 같은 시작을 인지하는 카메라, 그리고 스마트폰이나 월패드 과 같은 모바일 또는 장착된 입력기기등으로 나눌 수 있음
- 출력방식
  소리로써 사용자에게 처리 결과를 알려주는 인공지능 스피커, 시각적 처리를 해주는 모니터, 모바일 기기 등으로 나눌 수 있음

플랫폼

- 폐쇄형 플랫폼
  하나의 기업에서 표준을 정해 개발하므로 기능적으로 활용 범위가 넓고 신속한 체계를 구축하는 반면, 사용자 입장에서는 선택의 폭이 제한되는 단점
- 개발형 플랫폼
  모든 기업들의 기기들과 호환되므로 선택의 폭이 넓기는 하지만, 주어진 표준에 한정된 기능 구현으로 제약이 있을 수 있음

홈 서버

- 다양한 멀티미디어 홈서비스를 제공하기 위하여 멀티미디어 데이터를 저장, 관리, 분배하는 기능을 수행
- 홈네트워크에 접속된 각종 가전기기의 제어, 관리 및 연동을 담당
- 홈서버는 논리적인 의미를 가짐
- 독립적은 기기로 존재할 수도 있고 다른 기기에 탑재되는 경우도 있으며, 외부 네트워크에서의 서비스 원격 관리, 가정 내의 오디오, 비디오, 게임 및 디지털 방송 서버, 에너지 관리, 홈 자동화, 보안 서버 등의 기능을 담당

홈 서버를 위한 네트워크를 위한 구성

홈서버는 가정 내에서 AV기기나 PC가 뿔뿔이 흩어지게 취급하고 있던 멀티미디어 콘텐츠를 일괄관리 거주자가 좋아하는 때에 좋아하는 장소에서 시청 할 수 있는 환경 실현

- 마음에 든 것을 정리해 DVD에 기록하거나 인터넷을 통해서 원격지로부터 시청 가능한 기능과 기능 및 에너지 관리, 홈 자동화, 보안 서버를 가진 것도 있음

사물인터넷(INTERNET OF THINGS)

사물인터넷은 기존의 유선통신을 기반으로 한 인터넷이나 모바일 인터넷보다 진화된 단계로 인터넷에 연결된 기기가 사람의 개입 없이 상호간에 알아서 정보를 주고 ㅂ받아 처리하나다.

사물 인터넷의 특징

- 고요하게 식별 가능한 사물이 만들어낸 정보를 인터넷을 통해 공유하는 환경,
- 연결 대상이 인간에서 사물, 공간, 자연으로 광범위하게 확장
- 3대 핵심 기술, 초고속 이동통신, 고감도 센서, 빅데이터 처리의 발전과 저 가격화로 가시화되기 시작

사물통신이 기기중심의 하드웨어적 접근이었다면, 사물인터넷은 솔루션 중심의 서비스 지향적 접근이라 할 수 있다.

유비쿼터스

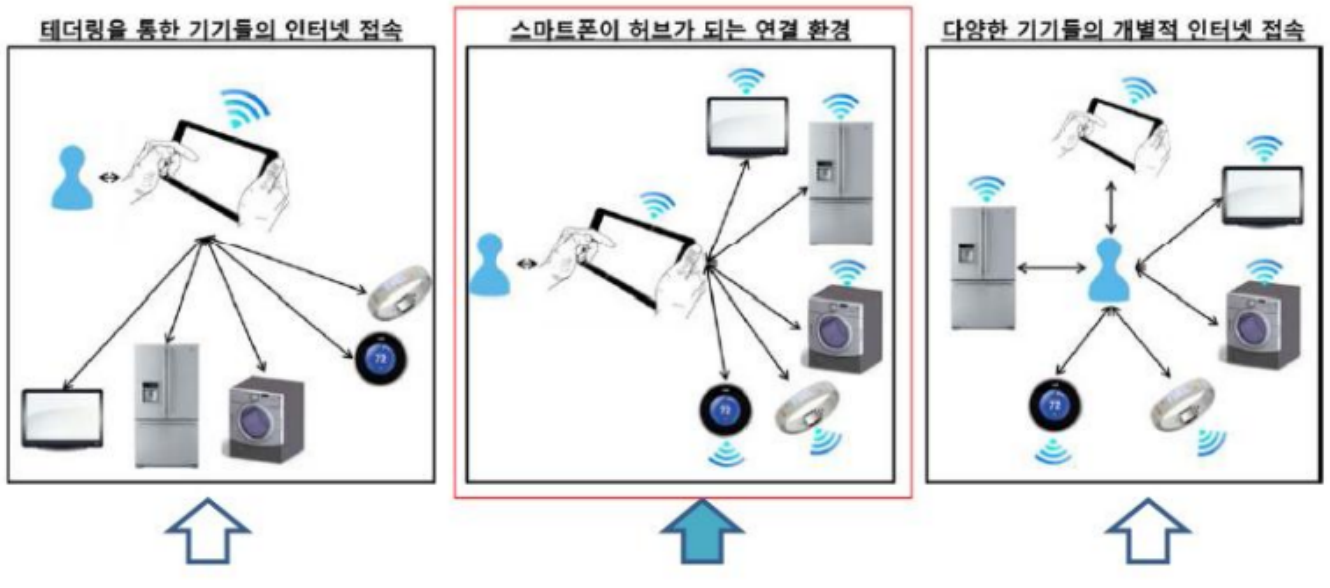
사용자가 네트워크나 컴퓨터를 의식하지 않고 장소에 상관없이 자유롭게 네트워크에 접속할 수 있는 정보통신 환경

M2M(Machine to Machine)

사물인터넷의 초기 단계로, 기계와 기계 간에 이뤄지는 통신이다.

2. 사물 인터넷의 사용 형태

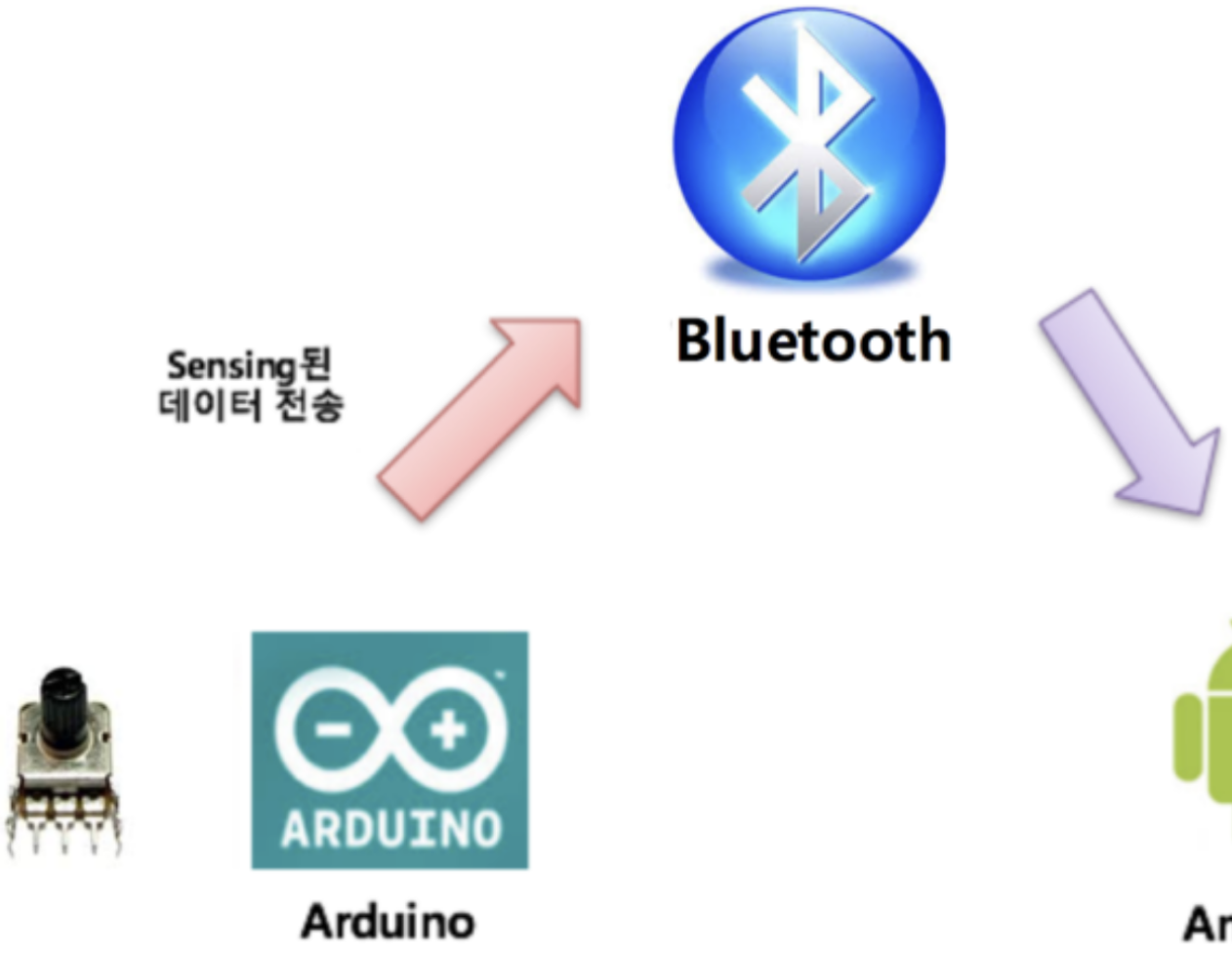
현재의 스마트폰 중심의 소비자 사용형태의 연장선상에서 IoT의 사용 환경이 나타날 가능성이 높음



사물과 앱 블루투스 통신을 통한 데이터 통신

*** 구현 및 설계**

아래와 같은 구조를 구현할 것이다.



위의 구조는 아두이노에서 가변저항의 값을 센싱하고 이를 블루투스를 통해 안드로이드 앱에서 확인하는 과정이다. 또한 안드로이드 앱에서 특정 명령을 주어서 아두이노의 LED를 제어하는 구조이다. 와이파이나 이더넷을 사용하여 위의 구조를 구현 시 서버를 개발해야 되고 연결과정 또한 복잡하나 블루투스를 이용하면 페어링을 이용하여 쉽게 아두이노와 앱이 연동되어서 효과적이다.

웹 애플리케이션 또는 웹페이지만 클라이언트 애플리케이션의 일부로 제공하려는 경우 `WebView`를 사용하면 됩니다. `WebView`클래스는 Android의 `View`클래스의 확장으로, 웹페이지를 활동 레이아웃의 일부로 표시할 수 있게 해 줍니다. 탐색 컨트롤이나 주소 표시줄 등 완전히 개발된 웹브라우저의 기능은 전혀 포함되어 있지 *않습니다*

`WebView`의 모든 작업은 기본적으로 웹페이지를 표시하는 것입니다.

```
webview.apply {
    webViewClient = WebViewClient()
    settings.javaScriptEnabled = true
}
webview.addJavascriptInterface(WebAppInterface(this), "Kchoi")

webview.loadUrl("https://m.aomgael.com/test")
```

위 코드가 의미하는 것을 설명하자면,

먼저 webview.apply부분은 웹뷰가 자바스크립트를 사용할 수 있게 하겠다는 말이고,

addJavascriptInterface에서는 WebAppInterface라는 클래스에 this(context)를 넘긴다는 뜻이다.

WebAppInterface는 아래 코드에 있다.

📲 앱과 웹의 통신 플로우

여기서 눈여겨 봐야하는 것은 native 속 함수와 javascript 속 함수이름이 같다는 것 볼 수 있다.

웹에서 앱을 호출하고 값을 보낼 때

Native code

```
@SuppressLint( ...value: "NewApi")
@JavascriptInterface
fun bluetoothEnable(userInfo: String) {
    val data: JSONObject =  JSONObject(userInfo)
    connectBluetooth(data)
}
```

Javascrip code

이제 웹 애플리케이션은 `WebAppInterface`클래스에 액세스할 수 있습니다.

예를 들어 다음은 사용자가 블루투스 버튼을 클릭하면 블루투스 기능을 부르는 자바스크립트코드입니다.

`WebView`가 이 인터페이스를 웹페이지에 사용할 수 있도록 자동으로 설정합니다. 따라서 버튼을 클릭하면 `clickBluetooth()`함수가 `Android` 인터페이스를 사용하여 `bluetoothEnable()` 메서드를 호출합니다.

```
// javascript 코드
블루투스 버튼을 클릭하는 이벤트 함수
clickBluetooth() {
      let message = {
        action: "connectBluetooth",
        accessToken: this.$store.state.accessToken,
      };
      /*global Webview*/
      /*eslint no-undef: "error"*/
      Webview.blueConnect(message);

 },
```

```
// javascript에서 native 코드 속 함수를 호출하는 함수
이벤트 함수에서 앱을 호출할때 설정하는 모듈 속 함수
blueConnect(message) {
    if (userAgent.indexOf("android") !== -1) {
      window.Android.bluetoothEnable(JSON.stringify(message));
    } else if (userAgent.indexOf("iphone") !== -1 || userAgent.indexOf("ipad") !== -1) {
      console.log("iphone: " + window.webkit.messageHandlers.webViewScriptHandler);
      window.webkit.messageHandlers.webViewScriptHandler.postMessage(message);
    }
  },
```

user agent는 HTTP 요청을 보내는 디바이스와 브라우저 등 사용자 소프트웨어의 식별 정보를 담고 있는 request header의 한 종류이다.

보통 HTTP 요청 에러가 발생했을 때 요청을 보낸 사용자 환경을 알아보기 위해 사용한다.

Mozilla 정보/버전 + 운영체제 정보 + 렌더링 엔진 정보 + 브라우저
형태로 노출

AOS의 경우

```
Android
Mozilla/5.0 (Linux; Android 9; SM-G955N Build/PPR1.180610.011; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/75.0.3770.89 Mobile Safari/537.36
```

IOS의 경우

```
Mozilla/5.0 (iPhone; CPU iPhone OS 12_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) CriOS/71.0.3758.78 Mobile/15E148 Safari/605.1
```

각 정보를 보면 알 수 있듯이 안드로이드폰에서는 Android, 아이폰에서는 iPhone이라는 문자열을 가지고 있는것을 볼 수 있다.

이걸 통해서 아이폰과 안드로이드폰을 구분할 수 있다

앱에서 웹을 호출하고 값을 보낼때

Javascript code

```
// 카카오 로그인 완료 후 데이터 받기용 브릿지
kakaoSignResponse(kakaoSignData) {
  const accountType = kakaoSignData.type;
  const signData = kakaoSignData.kakaoAccount;

  const slicedPhoneNum = signData.phoneNumber.slice(4, 20);
  const getPhoneNum = `0${slicedPhoneNum.slice(0, 2)}${slicedPhoneNum.slice(
    3,
    7
  )}${slicedPhoneNum.slice(8, 20)}`;
```

Native code

```
    var callScript: String = "javascript:Native.kakaoSignResponse(${jsonObject.toString()})"
    mWebView.loadUrl(callScript)
}

override fun onFailure(call: Call<JsonElement>, t: Throwable) {
    Log.d( tag: "Callable", t.message.toString())
    var callScript: String = "javascript:Native.kakaoSignResponse({})"
    mWebView.loadUrl(callScript)
}
```