선택적 동적 서비스 컴포지션이 가능한 안드로이드 사물인터넷 애플리케이션

우성필, 허세현, 임장관, 김성훈, 김대영

한국과학기술원 전산학과

Android Internet of Things Application using Selective Dynamic Service Composition

Sungpil Woo, Sehyun Heo, Janggwan Im, Seonghoon Kim, Daeyoung Kim
Computer Science Department, KAIST

요 약

현재 스마트 폰 주변에는 다양한 지능사물들이 존재하고, 그 수가 기하급수적으로 증가하고 있다. 이러한 지능사물과 스마트 폰의 인터랙션은 스마트 폰 내부 센서만을 이용하여 개발 된 안드로이드 애플리케이션과 비교하여 좀 더 진보된 형태의 애플리케이션을 제공하지만, 구현 시점에서 정적으로 특정한 사물과의 바인딩이 결정되어 제공되기 때문에, 하나의 애플리케이션은 구현 시 바인딩이 정의된 사물들과만 인터랙션이 가능하다. 스마트 폰의 이용자들은 이동하고, 그에 따라 주변 상황과 주변에 있는 사물의 종류가 계속해서 변하지만, 현재의 안드로이드 애플리케이션은 사용자에 이동에 따라 계속해서 변하는 주변상황에서 사용자가 원하는 사물과 사용자의 목적에 적합하도록 동적, 선택적으로 서비스 컴포지션 하는 기능을 제공해 주지 못한다. 이 논문에서는 현재의 안드로이드 애플리케이션 구조에 OSGi 프레임워크와 개발자가 이용 가능한 IoT-API를 제공 함으로써 런 타임에 사용자의 선택에 따라 주변 사물과 서비스 컴포지션이 가능한 새로운 형태의 애플리케이션의 구조를 제안한다.

1. 서 론

현재 스마트 폰 주변에는 유용한 정보와 서비스를 제공하는 다양한 지능사물들이 있고, 그 수가 기하급수적으로 늘어나고 있다. Gartner (2010)의 전망에 의하면 이러한 지능사물의 수가 2040년에는 1조 개에 이를 것으로 추산하고 있다.

이렇게 많은 수의 지능사물이 등장함에 따라 스마트 폰 내부의 센서들만을 이용하는 기존 안드로이드 애플리케이션과 차별되는, 그림 1과 같이 사용자들에게 지능사물과의 인터랙션이 가능하게 하는 애플리케이션들이 제공되고 있다. 사용자는 스마트 폰을 통해서 조명이 색이나 밝기를 자유롭게 조절할 수도 있고 (그림 1-a), 자신의 신체의 신체(뇌)의 상태나 주변 환경의 상태정보를 지능사물로부터 자유롭게 제공받고 이용할 수도 있다 (그림 1-b, 2-c). 여기서 든 예시 외에도 제조사와 개발자는 스마트 폰을 통해 사용자들에게 다양한 주변 지능사물들을 이용한 애플리케이션을 제공할 수 있다.

그림 1 외부 사물을 이용한 새로운 형태의 애플리케이션

본 연구는 미래창조과학부 및 한국산업기술평가관리원의 산업융합원천기술개발사업(정보통신)의 일환으로 수행하였음.
[10041313, UX지향 모바일 SW플랫폼 개발]

하지만 현재의 안드로이드 애플리케이션은 스마트 폰 주변에 존재하는 지능사물에 대한 고려가 반영되지 있지 않은 상태에서 그 구조가 고안 되었다. 그로 인하여, 현재 구조에서 애플리케이션 개발자는 스마트 폰 내부의 기능만으로 애플리케이션을 개발하거나 혹은 지능사물을 이용하여 애플리케이션을 개발하더라도 대부분은 하나의 애플리케이션에 하나의 사물만 연결된 형태로 서비스를 제공하며, 연결 된 사물의 경우에도 특정 제조사와 모델에 종속 된다. 따라서 같은 기능을 가진 다른 제조사의 사물이 존재하더라도 해당 애플리케이션에서는 사용할 수 없다. 또한 개발자는 애플리케이션을 개발할 때, 각 사물의 세부사항까지 모두 파악해야하기 때문에 구현을 하는 데에 많은 시간과 노력이 필요하며, 특정한 사물에 종속 된 애플리케이션만을 개발할 수 있다.

이 연구에서는 개발자가 스마트 폰과 주변 사물 사이의 인터랙션이 필요한 안드로이드 애플리케이션을 개발할 때, 런 타임에 사물 소프트웨어 번들이 사용자 선택에 의해 동적으로 조합되어 애플리케이션을 구성하는 새로운 형태의 사물인터넷 안드로이드 애플리케이션의 구조를 제안한다.

본 논문의 구성은 다음과 같다 2장에서는 새로 제안하는 사물인터넷 애플리케이션의 구조를 설명하고, 3장에서는 사물인터넷 애플[1]리케이션이 기존의 안드로이드 애플리케이션과 가진 차이점을 서술할 것이다. 4장에서는 사물인터넷 애플리케이션을 이용한 시나리오와 그 구현에 대한 서술을 하고, 마지막으로 5장에서 연구의 결론

과 향후 연구 방향에 대하여 서술을 할 것이다.

2. 사물인터넷 애플리케이션 구조

기존 안드로이드 애플리케이션에서 지원하지 못하는, 스마트 폰 주변에 존재하는 지능사물들과의 선택적 동적 서비스 컴포지션 기능을 지원하기 위해, 애플리케이션 내에 OSGi 프레임워크와 IoT-API를 추가한 다음과 같은 사물인터넷 애플리케이션구조를 제안한다(그림 2).

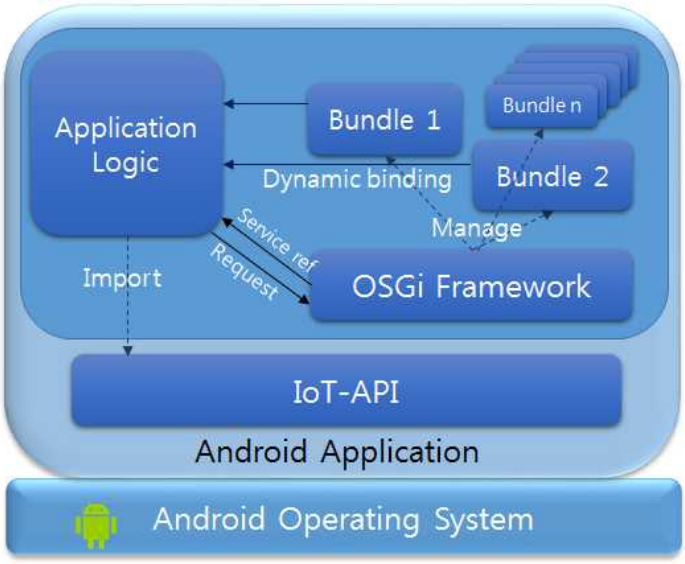

그림 2 사물인터넷 애플리케이션의 구조

2.1 OSGi 프레임워크

OSGi 프레임워크는 동적인 컴포넌트 모델을 지원하는 프레임워크이며, 프레임워크 상에서는 번들 단위로 응용 프로그램을 구동한다. 번들은 OSGi 프레임워크에서 사용하는 소프트웨어 모듈의 단위로, 프레임워크의 재시동 과정 없이 원격지를 통해 설치가 가능할 뿐만 아니라, 시작, 정지, 업데이트, 제거기능 또한 재시동 없이 런 타임에 가능하다. 또한 번들은 자신이 오버라이딩 한 인터페이스 이름을 기반으로 프레임워크에 서비스를 등록/검색이 가능하다. 가령 번들이 인터페이스A를 오버라이딩 하여 구현하였다면, 해당 번들과 바인딩을 하기 위해서는 번들과 동일한 인터페이스A를 import하여 이용하는 로직을 가지고 있으면 된다.

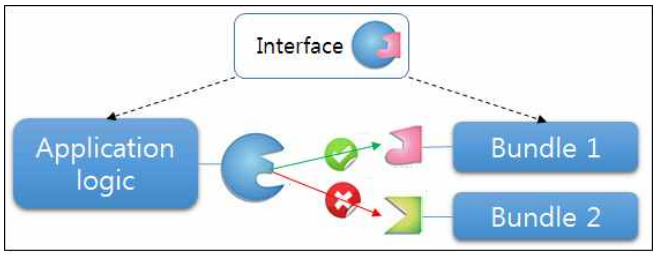

그림 3 인터페이스가 매칭될 때 동적 바인딩이 가능한 OSGi 번들과 안드로이드 애플리케이션.

번들 간의 바인딩 뿐만 아니라, 안드로이드 애플리케이션 내에서도 번들이 오버라이드 한 인터페이스를 import하여 애플리케이션 로직을 구성하면, OSGi 프레임워크에서 런 타임에 번들을 검색하여 매칭 되는 번들은 애플리케이션에 동적으로 바인딩 될 수 있다(그림 3).

2.2 IoT-API

2.1절에서 살펴본 것처럼, 애플리케이션 개발자가 선택적 동적 서비스 컴포지션을 구현하기 위해서는, 조합 하고자 하는 사물 번들이 가진 인터페이스를 import하여 애플리케이션을 구현해야 한다. 이를 위해서 사물인터넷 애플리케이션에는 특정 사물 군의 기능을 정의한 표준 인터페이스의 라이브러리인 IoT-API가 필요하다.

```
import android.widget.ScrollView;
import android.widget.TextView;
import android.widget.Toast;
import Standard_IoTApp_Interface.Bulb;
```

그림 4 IoT-API의 사용 예시

IoT-API는 기존 library와 동일하게 안드로이드 애플리케이션에서 import하여 이용할 수 있다(그림 4). 해당 API의 내부에는 인터페이스와 애플리케이션에서 이용 가능한 사물의 기능목록들이 다음과 같이 제공된다(그림 5).

```
package Standard_IoTApp_Interface;

public interface Bulb {
    public void turn_on_bulb();
    public void turn_off_bulb();
    public void change_color(int i, int j, int k);
    public void alert_bulb();
    ...
}
```

그림 5 IoT-API 전구의 Interface 예시

사물의 제조사에서는 IoT-API에서 제공하는 표준 인터페이스(그림 5)에 맞추어 번들을 만들어 제공해야 한다. 위에서 제공하는 표준에 맞추어 제공되는 사물은 런 타임에 해당 라이브러리를 이용하여 구현 된 사물인터넷 애플리케이션과 동적으로 조립이 가능하게 된다. IoT-API에서 제공하는 인터페이스 표준은 제조사의 요청에 따라 주기적으로 수정/업데이트 될 수 있다.

2.3 사물인터넷 애플리케이션 동작 절차.

사물인터넷 애플리케이션은 기존의 안드로이드 운영체제 위에서 동작한다. 애플리케이션 개발자는 사물인터넷 애플리케이션에서 제공하는 IoT-API를 이용하여, 조합하고자 하는 사물의 API를 import하여 애플리케이션을 구현 한다. 사용자가 완성 된 애플리케이션을 실행하면, 애플리케이션은 import하여 이용한 인터페이스를 통해 OSGi 프레임워크에 번들의 목록을 요청하고, 사용자의 선택에 따라, 선택 된 사물들의 번들이 애플리케이션의 런 타임에 동적으로 애플리케이션과 조합 되어 사물인터넷 애플리케이션을 완성하게 된다. 즉 사물인터넷 애플리케이션은 기존 안드로이드 애플리케이션과 같이 완성 된 소프트웨어로서 존재하는 것이 아니고, IoT-API를 이용하여 사물과의 바인딩만 정의되어 있는 상태로 제공된다. 사용자가 애플리케이션을 실행하

고 런 타임에 OSGi 프레임워크 상에서 번들로 구동 중인 특정 주변 지능사물을 선택하였을 때, 선택된 지능사물에 해당하는 서비스 번들과 동적인 서비스 컴포지션을 통해서 애플리케이션을 완성하는 형태의 방식으로 제공되는 것이다.

3. 기존 안드로이드 애플리케이션과의 차이점

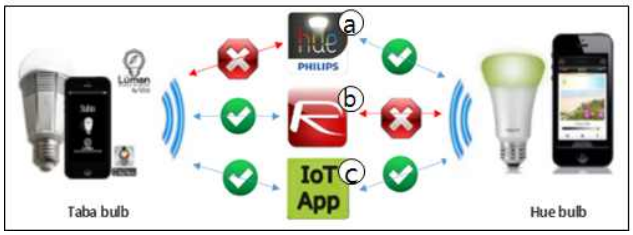

그림 6 특정 사물에 종속되는 기존 응용과, 종속되지 않는 사물인터넷 애플리케이션

기존의 안드로이드 애플리케이션의 경우에는 애플리케이션을 구현하는 시점에서 사물의 특정 모델과 정적으로 바인딩방식을 애플리케이션을 개발해야 한다. 현재 판매되고 있는 Philips사의 Hue-bulb 제품의 경우, 단지 Hue-bulb에 대해서만 그 애플리케이션의 기능이 제공되며 (그림6-a), Lumen사에서 제공하는 Taba-bulb의 경우에도 해당 애플리케이션에서만 동작한다(그림6-b). 하지만 위에서 제안한 사물인터넷 애플리케이션 구조를 도입할 경우, 사용자가 런 타임에 동적으로 사물을 선택하고, 해당 소프트웨어와 동적 선택적으로 서비스 컴포지션이 가능하기 때문에 특정 사물과 모델에 종속되지 않는다 (그림6-c). 이를 통해 스마트 폰 사용자들의 사물 이용 범위가 크게 확대 될 수 있다. 또한 애플리케이션 개발자들은 사물 각각의 세부사항보다는 사물의 기능에만 초점을 맞추어 개발할 수 있기 때문에, 사물을 이용한 애플리케이션을 개발할 때 개발 비용이 감소하게 된다.

4. 시나리오 및 구현

시나리오는 사용자 선택에 의하여 번들의 다운로드와 동적 구동 및 바인딩을 통해 애플리케이션이 완성되고 정상적으로 작동하는 것을 목표로 한다.

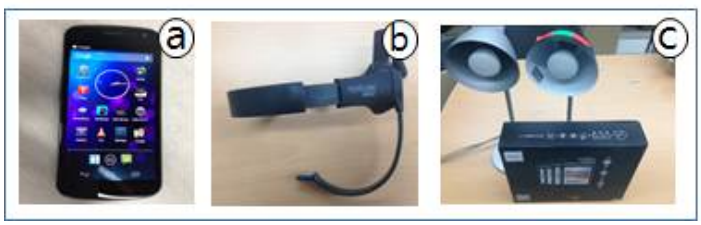

그림 7 시나리오에 이용 된 장비들

갤럭시넥서스, NeuroSky사의 EEG 센서인 Mindwave Mobile, 그리고 2개의 Philips Hue-bulb를 사용하여 시나리오를 고안하였다. 갤럭시 넥서스는 젤리빈 운영체제를 이용한다. Mindwave Mobile(그림7-b)은 헤드셋 형태로 사용자의 머리에 착용하며 뇌파감지센서가 사용자의 뇌파를

그림 8 애플리케이션 동작 화면 (a, b)와 눈 깜빡임에 따른 전구가 점등되는 사진 (c, d)

분석한 값이나 눈 깜빡임 간접적으로 측정하고 그 결과를 전송해주는 장치이다. Hue-Bulb(7-c)는 네트워크 전구로서 API를 통하여 전구에 Hue, Saturation, Brightness 값을 설정하고, 그에 따라서 그 색이 변화하는 전구이다. 시나리오는 다음의 순서로 진행된다. 사용자가 애플리케이션을 실행하면, 사물을 선택하는 화면이 나타난다(그림 8-a). 여기서 사용자가 애플리케이션의 동작을 위해 필요한 Mindwave Mobile과 Hue-bulb 버튼(그림 8-a의 그림 버튼)을 누르고, 바인딩 할 사물을 선택하면 (그림8-b) 선택한 사물에 해당하는 소프트웨어 번들이 동적으로 애플리케이션과 바인딩 된다. 같은 방법으로 바인딩이 필요한 모든 사물에 대해서 선택을 완료하면, 기존 애플리케이션 로직과 동적으로 조합되어 사용자가 이용하고자 하는 애플리케이션이 완성된다. 위에서 제시한 시나리오의 경우에는 Mindwave Mobile이 사용자의 눈 깜빡임 정보를 수신하여 해당 신호를 스마트 폰으로 보내면, 스마트 폰은 그 신호에 따라서 깜빡임 신호를 전구로 보내고, 전구도 깜빡이게 된다.

5. 결론 및 향후 연구

이 논문에서는 런 타임에 사용자 선택적으로 사물들을 애플리케이션과 조합하는 새로운 애플리케이션 구조를 제안하였다. 이는 안드로이드 애플리케이션에서 선택적 동적으로 서비스 컴포지션 기능을 제공했다는 점에서 의의가 있으나, 주변 사물을 자동으로 발견하고, 발견 된 정보를 통해 필요한 번들 들을 다운로드 하여 조합하는 데에는 이르지 못했다는 한계점이 있다. 향후 위에서 언급한 기능들을 제공할 수 있도록 스마트 폰 주변에 존재하는 이 기종의 사물을 동적으로 발견하고, 발견 된 사물들을 모델화 하여 효율적으로 관리하는 시스템 구조를 연구할 예정이다.

참 고 문 헌

[1] Teng-Wen Chang “Android/OSGi-based Vehicular Network Management System”

[2] Waylon Brunette, Rita Sodt, Rohit Chaudhri, Mayank Goel, Michael Falcone, Jaylen VanOrden and Gaetano Borriello “Open Data Kit Sensors: A Sensor Integration Framework for Android at the Application-Level” The 10th International Conference on Mobile Systems (MobiSys 2012)