논문 15-40-10-22 The Journal of Korean Institute of Communications and Information Sciences '15-10 Vol.40 No.10
http://dx.doi.org/10.7840/kics.2015.40.10.2047

데이터 플로우 기반의 네트워크 서비스 자율 설정을 위한 사물인터넷 플랫폼

권 기 덕◆, 유 영 환°

IoT Platform for Network Service Self-Configuration Based on Data Flow

Kideok Kwon◆, Younghwan Yoo°

요 약

본 논문에서 제안하는 사물인터넷 브로커 (게이트웨이) 플랫폼은 개발자와 사용자 측면을 고려하여서 설계하였다. 제안 플랫폼은 주변 디바이스를 Virtual-Entity로 서비스화하고 온톨로지를 이용하여 소유자를 추론함으로써 개발자가 쉽게 서비스를 조립할 수 있게 하는 한편 사용자 입장에서는 서비스가 자동으로 설치 및 설정될 수 있도록 한다. 또한 로컬 네트워크의 다양한 플랫폼과 프로토콜을 통합하는 방안으로 개발자 어댑터를 개발하여 플러그인 할 수 있게 하였다. 실험을 통하여 AllJoyn 플랫폼 플러그 인을 Node-RED에 추가하여 서비스 플로우를 테스트 하였다.

Key Words : Internet of Things, IoT Platform, Ontology

ABSTRACT

Our research proposese an IoT Broker (gateway) platform which considers the both sides of a service developer and a user. The platform associates the devices around us with virtual-entities and infers the owner of the devices through the ontology scheme. These functionalities can make the service developer easily design a new service. For the user, the service can be automatically installed and deployed under this platform. We also make it possible for the IoT broker to be plugged-in by integrating various communication platforms and protocols into that platform. Our expreriment with the AllJoyn plug-in on Node-RED substantiated that various service flows can be easily deployed and work on our platform.

Ⅰ. 서 론

최근 정보기술 분야 에서는 사물인터넷이 중요한 화두가 되고 있다. 사물인터넷이란 실생활에 존재하는 모든 객체들이 인터넷으로 서로 연결되어 정보를 주고받는 환경을 말한다. 여기서 객체는 사람, 사물, 사이버 자원 등 모든 것을 의미한다. 더 나아가 이러한 정보공유를 통하여 사람들의 삶의 질이나 편리함을 서비스로 제공해주는 패러다임이다.

그림 1과 같은 사물인터넷 청사진으로부터 사물인터넷 실현을 위해서는 3가지 과제의 해결이 선행되어야 한다. 첫 번째는 사물인터넷 생태계의 구축이다.

※ 이 논문은 2015년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.10043907, 개방형 고성능 표준 IoT 디바이스 및 지능형 SW 개발)
◆ First Author : Pusan National University Department of Computer Science & Engineering, merlin00@pusan.ac.kr, 학생회원
° Corresponding Author : Pusan National University Department of Computer science & Engineering ymomo@pusan.ac.kr, 종신회원
논문번호 : KICS2015-09-278, Received September 1, 2015; Revised October 20, 2015; Accepted October 20, 2015

그림 1. 서비스 마켓 기반의 사물인터넷 형사진
Fig. 1. IoT (Internet of Things) Platform based on the service market

애플이나 안드로이드 앱 시장이 활성화 될 수 있는 계기는 이러한 앱 생태계를 구축함으로써 개발자와 사용자가 마켓을 통해 쉽고 편리하게 자신의 앱을 구입하고 판매할 수 있기 때문이다[1-2]. 따라서 사물인터넷도 활성화를 위해서는 개발자가 쉽게 서비스를 만들고 판매 할 수 있어야 하고 또한 사용자가 그 서비스를 구입하여 편리하게 설치 할 수 있어야 한다. 이러한 사물인터넷 생태계가 갖추어진다면 한 시대의 패러다임이 아니라 실생활에 존재하는 기술로서 인정될 것이다. 그러므로 생태계 구축은 사물인터넷에서 중요한 하나의 과제이다.

두 번째는 쉬운 서비스 개발 환경 지원이다. 사물인터넷의 서비스 개발자는 안드로이드 개발자와 같이 특정 기업이 서비스를 개발 할 수 도 있고 개인이 홀로 서비스를 개발 할 수도 있다. 복잡한 서비스는 기업이 자신의 빅데이터를 이용하여 분석하고 서비스를 제공하는 반면에 실생활에 필요하고 간단한 창의적인 서비스는 개인 개발자가 더 유리할 수 있다. 또한 디바이스의 다양성을 지원할 수 있는 개발환경도 필요하다. 따라서 사물인터넷을 현실화하기 위해서는 개발자 측면도 고려한 서비스 개발 환경 지원이 요구된다.

세 번째는 편리한 사물인터넷 서비스 배치 및 설치 환경 지원이다. 시중에는 여러 기업이 개발한 다양한 디바이스가 존재하며 각 사용자마다 사용하는 디바이스 모델과 제조업체도 다르다. 예를 들어 헬스케어 사물인터넷 서비스를 가정하면 이 서비스는 사용자의 활동량, 몸무게 등 데이터 수집을 요구한다. 그러나 사용자 주위에는 많은 센서가 존재하므로 적절한 센서를 이 서비스에 연결하는 것은 어려운 과제이다. 따라서 사용자 측면에서 사물인터넷을 활성화하기 위해서는 서비스 배치와 설치가 자율적으로 이루어져 사용자 개입을 최소로 하는 것이 중요하다.

제안 플랫폼은 제시한 실현과제를 고려하여 설계하였다. 1) 사물인터넷 생태계와 친화적인 사물인터넷 플랫폼 구현이다. 본 논문에서는 사물인터넷 생태계를 현재 애플 또는 안드로이드 마켓과 유사할 것이라고 가정하였다. 즉 사용자는 패키지화된 서비스를 구매하여 사용할 것이다. 따라서 서비스에 필요한 요구사항을 하나의 사물인터넷 서비스 패키지에 포함시키는 형태로 설계하였다. 즉, 사물인터넷 생태계가 서비스 패키지를 지원한다면 어떠한 시스템에서도 사용가능할 것이다. 2) 다양한 디바이스가 존재하는 환경에서 쉬운 서비스 개발 지원이다. 사물인터넷 플랫폼에서는 개발자가 일반화된 정보를 이용하여 디바이스 연결과 데이터 흐름을 정의할 수 있게 하였다. 예를 들어 서비스 개발자가 A사 또는 B사 온도 센서가 아닌 일반적인 온도센서라고 개발자가 명시하는 방안을 제시한다. 그리고 이와 같은 정보를 사물인터넷 패키지에 포함시킴으로써 쉬운 개발 환경을 지원한다. 3) 편리한 사물인터넷 서비스 배치 및 설치 방법을 제시한다. 먼저 서비스 매쉬업 기능이 강화된 게이트웨이인 사물인터넷 브로커를 제안한다. 브로커는 서비스 패키지를 활성화하고 서비스 개발자가 요구한 디바이스들을 자동으로 연결시켜준다. 이러한 목적을 위하여 제안 플랫폼은 Virtual-Entity Association과 Flow Auto Configuration 기능을 구현하였다. 자세한 사물인터넷 플랫폼은 3장에서 설명한다.

제안 플랫폼 구현은 현재 사물인터넷을 위해 개발된 툴을 최대한 활용하였다. Node-RED[3]는 IBM에서 개발한 대표적인 툴로서 제안 플랫폼의 기능과 플로우 매쉬업을 시각화하여 사용자에게 보여준다. 또한 현재 로컬영역의 사물인터넷 플랫폼인 AllJoyn을 통합하여 로컬에 연결된 디바이스 간 서비스 플로우 설정이 가능하다.

2장에서는 구현에 사용된 플랫폼과 Front-End 소프트웨어 대한 관련 연구를 살펴본다. 3장은 제안 사물인터넷 브로커 플랫폼의 구조와 아키텍처의 구성요소에 대해서 상세히 기술하고, 4장은 제안 플랫폼 아키텍처를 기반으로 실제 구현방법과 간단한 응요 실험을 설명한다. 마지막으로 5장에서는 본 논문의 향후 과제에 대하여 논의 할 것이다.

Ⅱ. 관련 연구

AllJoyn[4] 플랫폼은 퀄컴에서 먼저 개발한 후 Allseen Alliance를 결성하여 오픈소스로 개방하였다. Allseen Alliance는 현재 170여개의 기업들이 참여하

고 있으며 대표적인 기업으로 LG, Microsoft, Qualcomm, Sony 등이 있다. 또한 C, C++, Java, Javascript 등 다양한 언어로 AllJoyn 플랫폼을 이용할 수 있게 지원하고 있다. AllJoyn 플랫폼은 로컬 네트워크 내의 디바이스가 자신의 기능들을 알리고 이용할 수 있게 하는 플랫폼이다. 플랫폼의 기본 개념은 RPC (Remote Procedure Call)와 D-Bus를 함께 사물인터넷에 적용한 것이다. 간단히 AllJoyn 동작을 보면 먼저 로컬 내의 디바이스가 접속하게 되면 자신의 연결되었다는 버스 정보를 주위 디바이스들에게 알린다. 이러한 정보를 알리는 역할은 AllJoyn router가 실행한다. 만약 다른 디바이스가 버스 정보를 받고 그 디바이스의 인터페이스를 알고 있다면 버스 정보를 이용하여 인터페이스의 메소드, 시그널, 속성을 원격에서 호출 가능하다. 그러나 AllJoyn 플랫폼은 로컬 디바이스들 간의 통신만 가능하고 인터넷을 통한 외부 디바이스와의 통신 기능은 제공하지 않는다.

OIC (Open Interconnect Consortium)는 또 다른 하나의 사물인터넷 플랫폼 연합으로서 IoTivity[5]라는 플랫폼 아키텍처 개발 프로젝트를 지원하고 있다. 이 프로젝트는 소스코드를 제공하고 있으나 AllJoyn에 비해 많은 기능과 개발 문서를 제공하지는 않는다. IoTivity 아키텍처 역시 사물인터넷을 위해 만들어졌기 때문에 AllJoyn과 비슷한 기능 또는 동일한 기능들이 많이 포함되어 있다. 아키텍처를 보면 AllJoyn과는 다르게 네트워크 프로토콜을 플러그인 할 수 있게 설계된 것을 알 수 있다. 이것은 AllJoyn이 로컬 네트워크에서 사용되는 RPC 프로토콜만을 포함하는 반면 IoTivity는 처음부터 외부 인터넷과의 연결을 고려하여 설계가 이루어지고 있음을 알 수 있다.

Node-RED는 IBM에서 개발하였으며 사물인터넷을 위한 Front-End 비주얼 툴이다. Node-RED 에서는 각 디바이스 또는 프로토콜이 입력과 출력을 가지는 노드로 추상화되어 있다. 사용자는 웹을 이용하여 이러한 노드를 서로 연결하여 새로운 서비스 또는 데이터 처리 루틴을 만들 수 있다. 또한 Node.js 기반으로 개발된 Node-RED는 Node.js가 제공하는 많은 라이브러리를 개발자가 사용 가능하고 쉽게 새로운 노드를 추가 할 수도 있다.

Ⅲ. 사물인터넷 브로커 플랫폼

근래에는 사용자 주변에 많은 유무선 디바이스들이 존재하고 이용되어지고 있다. 그러나 각 디바이스는 개발 제조사와 자신이 수행하는 역할도 다르며 제조사는 디바이스를 특정 응용분야에 맞게 프로토콜이나 표준을 결정하여 개발한다. 이종 프로토콜과 표준을 사용하는 디바이스들은 서로 통신을 할 수 없고 이는 새로운 서비스를 구성할 경우 기존의 디바이스가 있으면서도 호환성을 위해 사용자가 새로운 디바이스를 구매해야 하는 이유가 된다. 다시 말하자면 이종 디바이스 간의 매쉬업을 할 수가 없다. 따라서 사물인터넷 플랫폼은 이종 디바이스가 서로 정보를 교환할 수 있는 환경을 조성해야 한다. 이러한 환경을 지원하기 위한 제안 플랫폼은 그림 2와 같다.

플랫폼의 기본 구성 요소는 디바이스와 가상자원을 일반화된 객체로 관리하는 것이다. 사물인터넷 분야의 연구자들은 사물들을 대부분 서비스로 일반화하고 있다[6-8]. 서비스화 하는 것은 모든 사물들을 구조화된 특정형태로 정의함으로써 관리나 매쉬업을 쉽게 만들어 준다. 제안 플랫폼 또한 앞선 연구와 유사하게 사물과 가상자원들을 서비스화 한다. 서비스 구조는 서비스 매쉬업과 단순화를 고려하여 정의하였다. 그리고 이 서비스 구조는 플랫폼에서 Virtual-Entity로 사용된다. 그림 3은 Virtual-Entity를 자세히 살펴보기 위해 도식화 하였다. 최대한 단순화를 위해 입출력만을 가지는 블랙박스 형태의 구조로 정의하고 각 입출력은 토픽을 명시함으로써 어떠한 입출력인가를 식별할 수 있게 하였다. 이와 같은 형태로 구조화 하는 것은 플랫폼 내의 매쉬업을 Virtual-Entity 입출만을 가지고 레고 블럭을 조립하는 것과 같은 효과를 준다[그림 3].

사물인터넷 브로커 플랫폼은 다양한 통신표준과 플랫폼으로 개발된 디바이스를 통합하여 동일한 방법으

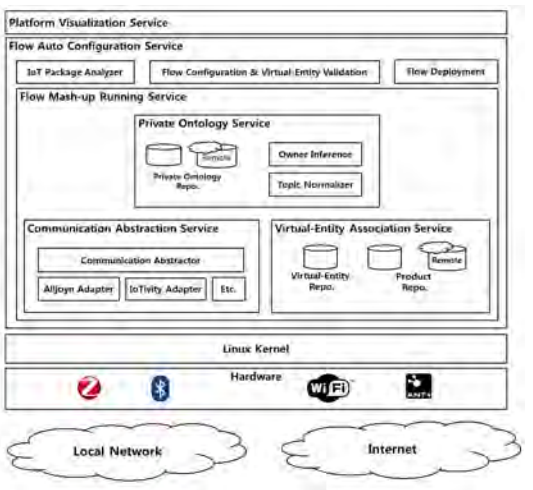

그림 2. 사물인터넷 브로커 플랫폼
Fig. 2. The overview of IoT broker platform

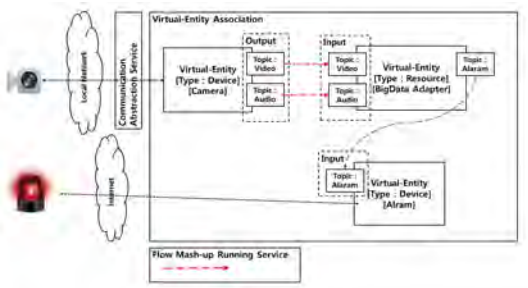

그림 3. Virtual-Entity 활용 구상도
Fig. 3. The usage of Virtual-Entity

로 Virtual-Entity 와 연결이 가능해야 한다. 각 디바이스는 WiFi, Zigbee, 블루투스와 같은 통신 프로토콜을 사용하고 또는 AllJoyn, IoTivity와 같은 플랫폼을 이용하여 제안 사물인터넷 플랫폼에 연결된다. 이러한 프로토콜과 플랫폼의 다양성을 모두 지원하는 것은 어렵고 많은 노력이 필요하다. 사물인터넷 브로커 플랫폼에서는 Communication Abstractor를 두어 다양한 프로토콜과 플랫폼을 Virtual-Entity에 맵핑 가능한 형태로 일반화 하였다. 그러나 각 프로토콜과 플랫폼은 기능과 특징이 차이가 있어 모두 동일한 방법으로 통신을 일반화 할 수 없다. 그러므로 각 개발자가 플랫폼에 확장할 수 있게 어댑터를 사용하는 구조로 설계하였다. 따라서 Virtual-Entity는 Communication Abstraction Service 통해 자신의 Virtual-Entity 입출력 통신을 맵핑할 수 있다.

Virtual-Entity는 동일한 형태의 디바이스를 일반화하고 서로 식별이 가능하여야 한다. 예를 들어 그림 4와 같이 각 A사, B사, C사 의 온도측정 센서가 있다고 가정하자 이러한 센서들은 모두 온도값을 측정하고 그 측정값을 전송한다. 그러나 제조사들은 각자의 모델명이나 명칭을 이용하여 디바이스들을 구분하고 그 정보를 사용자나 개발자들에게 제공한다. 제조사의 모델넘버 또는 디바이스 명칭과 관계없이 이들 센서의 본질적인 명칭은 온도센서이다. 만약 디바이스 명칭이 일반화 되지 않을 경우 서비스 개발자의 입장에서 모든 디바이스를 고려하여 서비스를 개발해야 한다. 이것은 매우 비효율적인 일이다. 개발자에게 필요한 것은 어떤 제조사의 온도센서인가가 아니라 누구 소유의 온도센서인가 그리고 어디에 위치하는가이다. 따라서 시중에 판매하고 있는 디바이스 정보들을 플랫폼 내의 데이터베이스에 저장하고 만약 디바이스로부터 연결 요청이 있다면 데이터베이스를 확인하여 디바이스와 연관된 해당 Virtual-Entity에 일반적인 명칭을 부여한다.

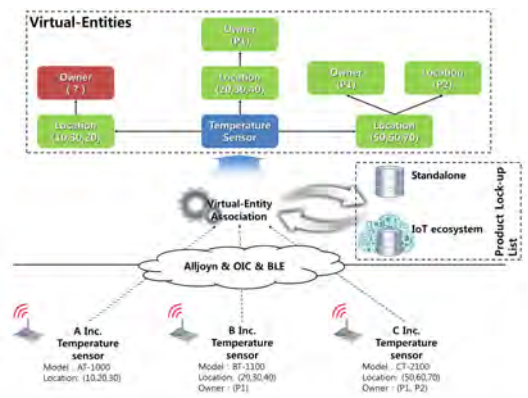

그림 4. 다양한 제조사의 디바이스를 Virtual-Entity로 연결
Fig. 4. Associating the various manufactures' devices with a Virtual-Entity

앞서 언급하였듯이 사용자나 개발자 입장에서는 디바이스의 위치와 소유자가 누구인가가 중요하다. 만약 플랫폼을 기반으로 서비스를 개발한다면 개발자는 서비스에 필요한 센서와 디바이스를 열거하고 누구 소유인가를 명시할 필요가 있다. 이러한 정보가 있어야지 자동으로 사용자의 디바이스를 검색하여 연결할 수 있기 때문이다. 그런데 디바이스들이 자신들의 정보를 완전히 알려주지 못하거나 정보를 알려주는 기능이 없을 경우가 있다. 이 경우 서비스 동작에 필요한 요구사항을 만족 할지라도 그 디바이스는 사용되지 못한다. 그러므로 플랫폼이 소유자를 추론하고 그 정보를 알려줄 수 있다면 사물인터넷에서의 디바이스 이용가능성을 최대화 할 수 있다. 이 기능은 제안 플랫폼에서 Private Ontology Service (POS)가 담당한다. Virtual-Entity Association Service(VEAS)처리 중에 Virtual-Entity의 소유자 정보를 알 수 없는 경우 POS에게 현재 알고 있는 디바이스 정보를 모두 알려주고 POS로부터 추론한 소유자 정보를 얻는다. 그림 5는 POS의 간단한 구상도를 보여 준다. 추론을 위해서 가장 먼저 온톨로지 초기에 사용자가 자신의 정보

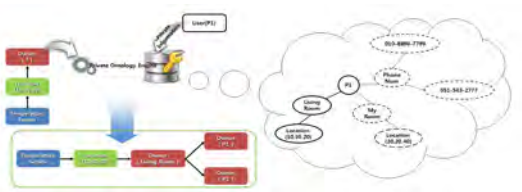

그림 5. 온톨리지 시스템 기반으로 하는 Virtual-Entity 소유자 추론
Fig. 5. Inferring the owner of a Virtual-Entity based on ontology system

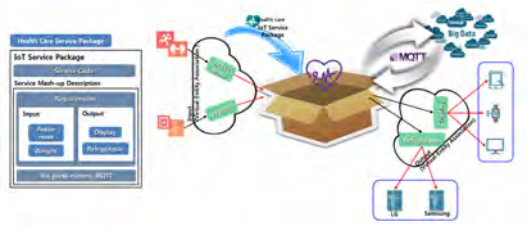

그림 6. 간략한 사물인터넷 서비스 패키지 구조 및 사물인터넷 브로커 플랫폼에서의 동작 구조
Fig. 6. A simple structure of IoT service package and the working process under IoT broker platform

를 플랫폼에 입력하게 한다. 이 정보를 기반으로 온톨로지를 구성하고 POS는 개인 온톨로지 정보로부터 Living Room과 P1이 특정 Virtual-Entity 소유임을 추론하여 VEAS에게 알려준다.

사물인터넷 브로커 플랫폼은 서비스 개발자 측면과 사용자 측면에서 사물인터넷을 쉽게 개발하고 편리하게 사용할 수 있는 기능들을 제공해야 한다. 개발자 측면에서는 주변 디바이스를 편리하게 탐색하고 연결할 수 있어야 한다. 탐색과 연결의 편의를 위하여 앞서 Virtual-Entity를 정의하였다. 이후 이러한 Virtual-Entity들을 열거하고 연결할 수 있게 요구사항을 서비스와 함께 하나의 패키지로 묶어 배포하면 개발과 사용의 편리성을 쉽게 제공할 수 있다. 즉, 개발자는 서비스코드와 서비스에 필요한 입출력을 열거하여 패키지화한다. 이것을 다운받은 플랫폼은 IoT Service Package Analyzer를 이용하여 패키지를 분석하고 해당하는 입출력을 가지는 Virtual-Entity의 유효성 검사를 한다. 만약 서비스에 필요한 디바이스가 없다면 사용자에게 경고 메시지 보낼 필요가 있기 때문이다. 이 기능은 플랫폼의 Virtual-Entity Validation통해 이루어진다. 따라서 그림 6의 사물인터넷 서비스 패키지 구조는 서비스 코드, 입력, 출력, 그리고 제3자(3rd party) 서비스 사용을 명시하게 하였다. 패키지 분석과 유효성 검사가 끝나면 Flow Deployment에 의해 실제 Virtual-Entity 간의 연결이 만들어지고 Flow Mash-up Running Service가 각 Virtual-Entity의 입출력이 일어날 때 데이터를 전달해주는 역할을 수행한다. 마지막으로 Platform Visualization Service는 개발자와 사용자가 현재 연결 디바이스와 서비스 플로우들을 시각적으로 확인하고 수정할 수 있는 서비스를 제공한다.

Ⅳ. 구현 및 실험

본 장에서는 앞서 제안한 플랫폼의 실현 가능성을 확인하기 위해 각 기능들을 구현하고 실험결과를 보인다. 플랫폼의 모든 기능을 처음부터 구현하고 실험하기에는 많은 시간이 필요하다. 그러므로 본 연구에서는 최대한 제안플랫폼과 유사한 개념의 플랫폼을 선택하여 수정하였다. Platform Visualization Service는 IBM의 Node-RED를 이용하였고 제안 플랫폼에 필요한 부분에 대해서는 직접 수정하였다. Communication Abstraction Service의 실험을 위해서 AllJoyn 플랫폼과 블루투스 통신을 선택하였고 플랫폼에 AllJoyn과 블루투스 어댑터를 구현하였다. 그리고 제안 플랫폼의 Flow Mash-up Running Service와 Flow Deployment 기능은 Node-RED의 개발 개념과 동일하므로 Node-RED에 이 기능이 존재한다. 따라서 개발자가 Node-RED 개발 가이드라인을 따르면 두 기능을 직접 구현하지 않고 손쉽게 이용가능하다.

그림 7은 헬스케어 서비스 위해 제안 플랫폼에 블루투스 어댑터를 구현하고 데이터 플로우를 설정한 실험화면이다. 실험을 위해 헬스케어 서비스를 간단하게 구성하였다. 우선 사용자가 몸무게와 활동량을 측정하면 그 정보를 TV를 통해 보여주는 시나리오이다. 이 서비스는 세 개의 모듈 즉, 블루투스 어댑터를 이용하는 ANDWeight 블루투스 체중계, 체중계의 측정값 가공하는 BitWeightParse, TV에 경고 메시지를 알리는 uNudge로 구성된다. debug 모듈은 실제 데이터를 Node-RED에서 확인하기 위해 필요하다. 그림 7의 오른쪽 그림에서 보는바와 같이 사용자의 몸무게가 실제로 전달되는 것을 확인하였다. 그러나 아직은 온톨로지를 통한 자동연결 기능이 완벽히 구현되지 않아 사용자가 직접 서비스 플로우를 연결하였다.

그림 8은 AllJoyn 플랫폼을 이용하는 디바이스가 제안 플랫폼에 연결되어 탐색된 결과 화면이다. 이 실험에서 AllJoyn 플랫폼을 사용하는 디바이스로는 안드로이드 스마트폰을 활용하였다. 먼저 플랫폼에 AllJoyn을 플러그인 하기 위해 어댑터를 구현해야한

그림 7. 블루투스 체중계를 이용한 헬스케어 서비스
Fig. 7. Healthcare service using a Bluetooth weight scale

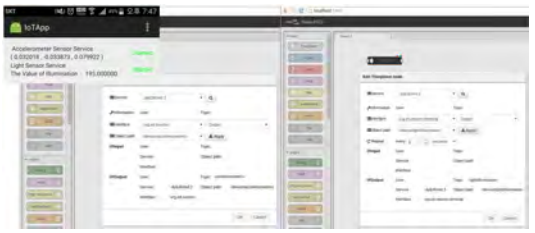

그림 8. AllJoyn Adapter 플러그 인 결과 화면(검색된 디바이스는 안드로이드 스마트폰)
Fig. 8. The test result for AllJoyn Adapter(The android cell phone which is found)

```
public class IoTInterfaces {
    @BusInterface(name = "org.iot.neuron", announced = "true")
    public interface IoTNeuron {
        @BusProperty
        public String getDeviceName();

        @BusProperty
        public String getOwner();
    }

    @BusInterface(name = "org.iot.neuron.dendrite", announced = "true")
    public interface IoTNeuronDendrite {
        @BusMethod
        public void dendrite(String str) throws BusException;
    }

    @BusInterface(name = "org.iot.neuron.terminal", announced = "true")
    public interface IoTNeuronTerminal {
        @BusMethod
        public String terminal() throws BusException;
    }
}
```

그림 9. AllJoyn 어댑터를 위한 인터페이스
Fig. 9. The interface for AllJoyn Adapter

다. AllJoyn 플랫폼은 오브젝트를 생성하고 이에 관련된 인터페이스만 정의하면 AllJoyn에 연결된 모든 디바이스는 이 인터페이스를 이용하여 통신이 가능하다. 따라서 제안 플랫폼은 주변 AllJoyn 디바이스가 그림 9의 org.iot.neurro 인터페이스만 지원해주면 플랫폼에 연결되어 사용될 수 있다. 즉, AllJoyn 어댑터는 org.iot.neurro 인터페이스를 통해 주변 디바이스와 통신을 한다. 그림 8에서 스마트폰의 가속센서와 조도센서의 연결과 그 디바이스들이 전송하는 데이터 토픽을 확인할 수 있다. 실험화면에서는 가속센서의 X축 값과 조명의 밝기 값을 토픽으로 되어 있고 그 데이터 값을 플로우 연결을 통해 전송할 수 있다.

V. 결론 및 향후 과제

본 연구는 사물인터넷 활성화를 위한 요구사항을 알아보고 요구조건을 충족할 수 있는 사물인터넷 플랫폼을 제안하였다. 제안한 플랫폼은 사물인터넷 활성화를 위한 서비스 마켓, 개발자와 사용자 측면을 고려하였다. 개발자 측면에서는 사용자 디바이스에 독립적인 개발이 가능하게 하였고 사용자 측면에서는 쉬운 설치를 가능하게 하였다. 이 기능들은 개발자가 개발한 서비스를 IoT Service Package로 패키지화하고 Flow Auto Configuration Service가 패키지를 분석하여 플랫폼에 설치한다. 또한 설계에 기본이 되는 Virtual - Entity 와 Private Ontology Service를 정의하였다. 마지막으로 Communication Abstraction Service의 가능성을 확인하기 위한 실험에서 제안플랫폼의 Front-End 서비스인 Node-RED를 이용하여 AllJoyn 디바이스가 연결되어 동작하는 것을 확인하였다. 현재까지는 플랫폼의Communication Abstraction Service만 구현되었지만 향후 나머지 부분을 구현할 것이다. 그리고 개인 온톨로지에 머신러닝 기능을 추가하여 소유자 추론을 향상시킬 필요가 있다. 마지막으로, 현재는 IBM의 Node-RED를 이용하였지만 플랫폼에 적합한 Front-End 시스템 개발이 필요하다.

References

[1] B. Jung, "The analysis of mobile app ecosystem for a developer's point of view (개발자를 중심으로 한 모바일 앱 생태계 분석)," *KISDI*, vol. 27, no. 7, pp. 1-25, Apr. 2014.

[2] S. T. Kim, J. S. Jeong, J. K. Song, and H. Y. Kim, "Trends of IoT device platforms and building its ecosystems," *Electron. Telecommun. Trends*, vol. 29, no. 4, pp. 82-90, Aug. 2014.

[3] Node-RED, *A visual tool for wiring the Internet of Things*, Available online: http://nodered.org/ accessed on 22 Aug. 2015.

[4] AllJoyn, *A platform for wiring the Internet of Things*, Available online: https://allseenalliance.org/ accessed on 19 Oct. 2015.

[5] IoTivity, *A platform for wiring the Internet of Things*, Available online: https://www.iotivity.org/ accessed on 19 Oct. 2015.

[6] J. Im, S. Kim, and D. Kim, "IoT mashup as a service: Cloud-based mashup service for the internet of things," in *Proc. Services Computing (SCC)*, pp. 462-469, Santa Clara(CA), USA, Jul. 2013.

[7] J. Gubbi, R. Buyya, S. Marusic, and M.

Palaniswami, “Internet of Things (IoT): A vision, architectural elements, and future directions,” *Future Generation Comput. Syst.*, vol. 29, no. 7, pp. 1645-1660, Sept. 2013.

[8] Q. Zhu, R. Wang, Q. Chen, Y. Liu, and W. Qin, “IOT Gateway: Bridging wireless sensor networks into internet of things,” in *Proc. Embedded and Ubiquitous Computing (EUC)*, pp. 347-352, Hong Kong, Dec. 2010.

권 기 덕 (Kideok Kwon)

2008년 8월 : 동의대학교 컴퓨터 공학과 졸업
2011년 2월 : 부산대학교 석사과정 수료
2011년 8월~현재 : 부산대하교 석박통합과정

<관심분야> 무선/이동네트워크, IoT

유 영 환 (Younghwan Yoo)

1996년 2월 : 서울대학교 전자공학과 학사
1998년 2월 : 서울대학교 컴퓨터 공학과 석사
2004년 2월 : 서울대학교 전기컴퓨터공학부 박사
2004년 5월~2006년 12월 : 미국 신시내티 대학교 컴퓨터과학과 연구원
2007년 3월~현재 : 부산대학교 컴퓨터공학과 교수

<관심분야> 무선/이동네트워크, IoT, 수중네트워크, 네트워크융합