KIPS Tr. Comp. and Comm. Sys.
Vol.5, No.1 pp.17~26 pISSN: 2287-5891

http://dx.doi.org/10.3745/KTCCS.2016.5.1.17

A Survey of the Self-Adaptive IoT Systems and a Compare and Analyze of IoT Using Self-Adaptive Concept

Seyoung Hwang† · Jangill Seo†† · Sungjun Park††† · Sangwon Park††††

ABSTRACT

IoT means things space networks that form the intelligent relationship such as sensing, networking, information processing about human being, things and service without explicit mutual cooperation of human being. Lately many IoT groups such as AllSeen Alliance, OIC launched a platform for IoT. Self-adaptive is aimed at implementation without the need for decisions of human being during the operation, so that the machine can respond to changes in its own determination. There is a need to apply the concept of self-adaptive to existing IoT and IoT platform. So In this paper, We look for trends of existing IoT, IoT platform and comparisons by applying a self-adaptive concept to IoT, IoT platform. In addition as an example of this paper, we suggest lacking self-adaptive elements to OIC.

Keywords : IoT, Self-Adaptive

자가적응형 IoT 시스템 개발 동향과 자가적응형 개념을 활용한 IoT 비교분석

황 세 영† · 서 장 일†† · 박 성 준††† · 박 상 원††††

요 약

IoT는 인간과 사물, 서비스, 이 세 가지 요소에 대해 인간의 명시적 개입 없이 상호 협력으로 센싱, 네트워킹, 정보처리 등 지능적 관계를 형성하는 사물 공간 연결망을 의미하고 현재 AllSeen Alliance, OIC 등과 같은 많은 IoT 그룹에서 IoT를 위한 플랫폼을 출시하고 있다. 자가적응형은 기기가 작동 중에 사람의 의사결정 없이도 가장 나은 대응을 스스로 판단하여 변화에 대처할 수 있도록 구현하는 것을 목표로 하는 것으로 기존 IoT와 IoT 플랫폼에 자가적응형 개념을 적용시켜 볼 필요성이 있다. 따라서 본 논문에서는 기존의 IoT, IoT 플랫폼, 자가적응형 연구와 개발 동향에 대해 살펴보고 IoT와 IoT 플랫폼에 자가적응형 분류체계를 사용한 비교 방식을 제안한다. 또한 이를 활용한 예시로 아직 개발 단계인 OIC의 IoT 플랫폼에서 부족한 자가적응형 요소에 대해 제안한다.

키워드 : 사물인터넷, 자가적응형

1. 서 론

IoT(Internet of Things)는 인간과 사물, 서비스, 이 세 가지 요소에 대해 인간의 명시적 개입 없이 상호 협력으로 센싱, 네트워킹, 정보처리 등 지능적 관계를 형성하는 사물 공간 연결망을 의미한다[1]. 현재까지 현실세계의 물리적 상황 정보를 수집하기 위한 연구는 RFID/USN 분야를 중심으로 연구되었지만, 수집된 정보를 가공하여 사용자에게 제공하는 수동적인 개념의 서비스가 주로 이루어졌다는 한계를 가지고 있다[2]. 이에 수동적인 형태의 서비스를 벗어나 사물 스스로 정보를 생성하고 공유하는 IoT 중심의 연구로 변화하고 있다. IoT를 구축하기 위한 기술에는 상황 인식 기술, 센싱 기술[3] 등이 있고, 이러한 기술을 이용하여 전용 개방형 플랫폼을 개발하여 IoT 환경을 구축하고 있다. 또한 여러 회사가 모인 커다란 IoT 그룹들이 존재하며 각 그룹은 각자의 IoT 플랫폼이 존재한다.

이러한 IoT 그룹으로는 현재 AllSeen Alliance, ANT+, Thread Group, oneM2M, OIC와 같은 많은 IoT 플랫폼 그룹이 존재하며, 각 그룹에서 제공하는 IoT 플랫폼을 사용한 많

※ 본 논문은 2015년 한국외국어대학교 교내학술연구비의 지원에 의하여 이루어진 것임.
† 준 회 원 : 한국외국어대학교 컴퓨터및정보통신공학과 석사과정
†† 비 회 원 : GS ITM 경영정보실 사원
††† 비 회 원 : 신도리코 경영정보실 사원
†††† 종신회원 : 한국외국어대학교 정보통신공학과 교수
Manuscript Received : September 21, 2015
First Revision : January 4, 2016
Accepted : January 5, 2016
* Corresponding Author : Seyoung Hwang(nahwasa@gmail.com)

은 시제품이 나오고 있다. 대표적인 예로 AllSeen Alliance의 AllJoyn 플랫폼을 사용한 전구, TV 등의 제품이 2013년부터 나오고 있으며, ANT+ 플랫폼을 사용한 스포츠웨어 제품 또한 이미 많은 시제품이 나와있는 상태이다.

'자가적응형'은 기기가 작동 중에 사람의 의사결정 없이도 가장 나은 대응을 스스로 판단하여 변화에 대처할 수 있도록 구현하는 것을 의미하며[4] 현재 자가적응형은 다양한 방면에서 많은 연구 및 조사자료들이 나오고 있다. 자가적응형에 관한 현재 연구들은 변화하는 환경에 대해 시스템이 스스로 어떻게 자가적응을 할 것인가에 대한 모델링을 정의하고 있다[5][6]. 이러한 모델링을 바탕으로 소프트웨어 분야에서 소프트웨어가 환경을 인지 또는 학습하여 스스로 적응하는 연구가 이루어지며, 운영체제 분야에서는 어플리케이션이 다수의 코어를 공동으로 사용하는 경우의 연구가 이루어지고 있다.

이러한 자가적응형 개념은 인간의 개입 없이 상호작용이 가능해야 하는 IoT 플랫폼에 적용 가능한 개념이다. 실제로 현재 나온 플랫폼 또한 각 기기간 메시지 전송과 같은 일반적인 서비스 외에도 새로운 기기가 IoT 네트워크에 연결될 경우 네트워크 상태에 맞게 자동으로 설정을 하는 등, 현재 기기나 사용자의 상황을 파악해 서비스하는 자가적응형 개념을 적용한 서비스를 제공하고 있다.

하지만 현재 IoT에 자가적응형 개념을 적용한 연구가 거의 없으며, 국내외의 IoT 플랫폼을 자가적응형 연구에서 제시한 분류 기준에 따라 비교, 분석한 자료가 없어 IoT 플랫폼들의 자가적응형 요소에 대한 비교, 분석이 힘든 상황이다. IoT 플랫폼에 자가적응 개념을 적용시킬 수 있으므로 각 플랫폼들이 자가적응형 요소를 충분하게 잘 지원하고 있는지 비교할 수 있는 방법이 존재해야 한다. 본 논문에서는 자가적응형 분류체계 중 '5W+1H' 분류체계를 사용한 비교 방식을 제안한다. 이 분류 체계를 IoT와 IoT 그룹, 자가적응형 플랫폼에 적용 결과 대부분의 시스템 요소에 대해 자가적응형과 관련된 사항을 빠짐없이 기술할 수 있었다. 이러한 비교를 통해 각 IoT 플랫폼들 간의 공통점과 차이점을 파악하여 기존의 IoT 플랫폼들의 부족한 부분을 발견하고 제안할 수 있다. 본 논문에서는 분류체계를 사용한 비교 방식 제안 외에도, 비교 분석을 활용한 예로써 분류체계에서 각 시스템 간에 적용되지 않는 부분을 통해 OIC IoT 통신 표준 공동 개발 컨소시엄에서 부족한 자가적응형 요소에 대해 제안을 한다.

본 논문에서는 2장에서 IoT, IoT 기술, IoT 플랫폼 등에 대해 개발 동향을 살펴보고, 3장에서 자가적응형의 연구 동향에 대해 살펴본다. 4장에서는 기존의 자가적응형 분류체계를 이용해 IoT와 IoT 플랫폼들을 분류하고 5장에서는 IoT 그룹 중 하나인 OIC에 부족한 자가적응형 요소에 대한 제안을 한다. 제안한 부분은 새로운 장치가 네트워크에 연결되었을 시 네트워크 상황에 맞게 설정이 진행되는 기능과 기기 자신이 수행 가능한 행동을 UI 형태로 만들어 다른 기기에게 자신을 동작할 수 있는 권한을 주는 기능으로 이 기능을 추가함으로써 자가적응형 분류체계에서 빠진 부분을 상당 부분 채울 수 있었다.

2. IoT 동향

2장에서는 IoT의 개념에 대해 설명하고, IoT를 위한 기술에 대한 기존 연구에 대해 알아본다. 또한 IoT 기술을 사용하거나 사용자가 사용할 수 있는 IoT 플랫폼에 대해 알아보고, 이러한 플랫폼을 제공하는 IoT 그룹들에 대해 간략히 설명한다.

2.1 IoT 개념

IoT(Internet of Things)란 인간과 사물, 서비스 세가지가 분산된 환경 요소에 대해 인간의 명시적 개입 없이 상호 협력적으로 센싱, 네트워킹, 정보처리 등 지능적 관계를 형성하는 사물 공간 연결망을 의미한다. 현재까지 현실세계의 물리적 상황 정보를 수집하기 위한 연구는 RFID/USN 분야를 중심으로 연구되었지만, 수집된 정보를 가공하여 사용자에게 제공하는 수동적 개념의 서비스 형태가 주로 이루어졌던 한계를 가지고 있다[2]. 이에 수동적인 형태의 서비스를 벗어나 사물 스스로 정보를 생성하고 공유하는 시스템을 구축하기 위해서 IoT 중심의 연구로 변화하는 중이다.

IoT 시스템을 구축하기 위한 가장 핵심이 되는 3대 주요 요소는 인간, 사물, 서비스이다. 첫 번째 요소로 인간은 독립적 주체로 사람과 그 사람의 사고, 행동 양식을 의미한다. 두 번째로 사물은 물리적 객체로서의 유형의 사물과 가상의 객체로서 IT 서비스에서 어떤 기능을 수행하는 함수, 객체인 무형의 사물을 의미한다. 세 번째로 서비스는 특정 목적을 위해 구현된 프로세스와 동작 메커니즘의 집합을 의미한다. IoT의 주요 구성요소 간의 소통, 즉 인간과 사물, 사물과 서비스, 인간과 서비스 또는 같은 요소 간의 소통을 통해 IoT를 구성하게 된다.

2.2 IoT를 위한 기술의 종류

IoT 시스템을 위한 여러가지 기술들이 연구되어 왔지만, 본 논문에서는 중요하다고 생각되는 몇 가지 기술에 대해서 설명하고자 한다.

1) 3대 주요 기술

2011년에 발간된 USN 산업발전 전략에 따르면 IoT의 3대 주요 기술을 센싱 기술, 유무선 통신/네트워킹 기술, USN 서비스 인터페이스 기술로 정의하였다[5]. 각 기술에 대해 살펴보면 다음과 같다.

a) 센싱 기술 : 유형 사물이나 온도, 습도, 가스 센서와 같은 물리적 센서로부터 센싱하는 것과 이미 센싱한 데이터로부터 특정 정보를 추출하는 가상 센싱이 포함된다.
b) 유무선 통신/네트워킹 기술 : 인간과 사물, 서비스를 연결시킬 수 있는 모든 유무선 네트워크를 의미한다.
c) USN 서비스 인터페이스 기술 : 정보를 센싱, 가공, 추출, 처리, 저장, 판단, 상황 인식하는 등의 서비스 제공을 위한 인터페이스 역할을 수행한다.

2) 상황 인식 기술

IoT의 식별과 시맨틱 중심의 관점에서 상황 인식은 센싱된 데이터로부터 의미있는 정보를 추출하는 중요한 기술이다[7]. 상황 인식 시스템은 상황 정보에 따른 라이프 사이클을 가지고 있는데, 상황 수집(Context Acquisition), 상황 모델링(Context Modelling), 상황 추론(Context Reasoning), 상황 배포(Context Dissemination)의 4단계로 이루어져있다. 이 사이클은 물리, 가상, 논리적 센서를 통해 데이터를 수집하고, 상황 정보를 수집 및 추론한다. 이를 통해 새로운 지식을 추론하고, 사용자에게 결과를 전달하는 기술이라 할 수 있다.

3) 스마트 데이터 로거 기술

스마트 데이터 로거는 M2M/IoT 분야에 있어서 상용 네트워크의 안전성을 기반으로 이기종의 센서 및 레거시 시스템을 지원할 수 있는 범용성을 지닌 반응형 소프트웨어를 의미한다[8]. 주변 환경에 대해서 효과적인 모니터링을 위해서 4가지의 핵심적인 기능들을 지원해야 한다. 첫 번째 기능은 원격에 위치한 사용자의 명령에 따라 프로그램의 설정 정보를 변경할 수 있는 기능이다. 두 번째는 센서의 고장이나 관측 데이터의 오류 등 예외상황에 대해서 사용자나 관리자의 개입 없이도 기기 스스로 예상, 진단하여 대응하는 것이다. 세 번째는 사용자가 사전에 정의한 특수한 상황이 발생했을 때, 사용자 또는 서버에게 능동적으로 알려주는 기능이다. 네 번째는 이미 구축된 센서 시스템들을 M2M 노드와 연결하여 새로운 시스템 구축 없이 환경 센서 네트워크 구축을 지원하는 것이다.

4) 수집된 센서의 정보가 스마트 기기에서 어떻게 관리될 수 있는지에 대한 방법 및 기술 〔9〕

스마트 기기와 외부센서의 융합에서 외부 센서로부터 공급되는 정보를 수집하는 방법에는 인터넷 기반의 융합과 기기간 직접 융합의 두 가지 방법이 있고, 그 다음의 기술적인 문제를 처리하는 방법으로 센서 정보 관리 방법이 있다.

먼저 정보 수집의 방법에서, 첫 번째 인터넷 기반의 융합은 여러 개의 각각 독립된 센서 기기에 독립적인 IP 주소를 할당하고, 각 센서 기기를 하나의 인터넷 객체로 인식하게 하여 주소 체계를 정리하는 것이다. 두 번째로, 기기간 직접 융합 방법은 스마트 기기와 센서에 내장된 네트워크 인터페이스를 이용하여 두 기종간 직접 통신을 가능하게 하는 방식이다.

센서 정보 관리는 외부 센서로부터 공급되는 정보를 수집하고, 그 다음 기술적인 문제는 어떻게 처리할 것인가에 대한 방법이다. 그 방법으로 ODK(Open Data Kit) Sensor Framework는 센서 관리 계층을 두어 외부로부터 유입되는 센서 정보를 저장하고, 관리 받아온 정보를 여러개의 응용서비스가 공용으로 활용하여 효율적 자원 관리 환경을 조성하는 방법이다.

2.3 IoT 플랫폼

IoT를 위한 디바이스가 IoT의 종단 단말이나 신규 서비스로 발전하기 위해서 통합적인 차원에서 연결해주는 플랫폼의 필요성이 제기되었다. 그에 따라서 전용 플랫폼과 개방형 플랫폼이 개발되었다.

1) 전용 플랫폼

전용 플랫폼은 특정 기기에 특화된 폐쇄형 플랫폼 이지만, 해당 기기를 중심으로 다양한 외부 서비스와 연동하여 환경을 구축할 수 있으며, 일부 플랫폼은 서드파티(3rd party) 기기의 환경에 편입도 가능하다는 특징을 가지고 있다. 전용 플랫폼의 예로 Wi-Fi가 내장된 콘센트형 스위치로 스마트 폰을 통해서 On/Off 할 수 있는 'Wemo'가 있다.

2) 개방형 플랫폼

개방형 플랫폼은 사물의 구성에 제한을 두지 않고, 프로토콜과 Open API를 이용하여 여러 사물들을 이용한 독자적인 서비스를 구축하는 플랫폼이다. 일반 사용자도 IoT 구축이 가능하며, 플랫폼을 구축할 만한 역량이 부족한 독립개발자나 중소기업에 서도 쉽게 사용할 수 있다. 개방형 플랫폼의 예로 인터넷을 통해 센서 데이터를 저장하여 웹 서비스와 연동하고 사용자에게 데이터 시각화를 제공하는 'ThinkSpeak'나 '2.4 IoT 그룹'에서 설명할 AllJoyn, oneM2M, OIC 등의 그룹에서 개발한 IoT 플랫폼이 있다.

2.4 IoT 그룹

IoT 그룹은 여러 기관 혹은 회사가 상업적 목적 또는 IoT 발전을 위해 설립한 것으로 현재 많은 IoT 그룹이 존재한다. 그 중 AllSeen Alliance, oneM2M, Thread Group, ANT+, OIC의 설립 년도, 설립 목적, 주요 참여 기업, 제공 플랫폼은 Table 1과 같다.

이러한 IoT 그룹에서 제공하는 플랫폼에서 제공하는 서비스는 몇 가지 부분에서 차이점을 나타낸다. 원하는 기기를 검색하는 기능의 경우 Thread 플랫폼에는 존재하지 않고 다른 플랫폼들에는 모두 존재한다. 이 기능을 통해 각각의 기기들은 유기적으로 원하는 서비스를 가진 기기를 검색하여 사용할 수 있다. 메시지를 네트워크상의 다른 모든 기기에게 알림 형태로 보낼 수 있는 기능은 AllJoyn, oneM2M, OIC 플랫폼에 존재하며, 다른 기기의 메시지를 받고자 자신을 등록해두는 기능 또한 이 세 플랫폼에서 지원한다. 주변의 해당 플랫폼 네트워크로 자동 접속하는 기능은 AllJoyn, ANT+, OIC에서 제공하며 네트워크에 연결된 다른 기기를 원격으로 설정하는 기능은 AllJoyn, Thread, oneM2M에서 지원한다. 해당 프레임워크를 사용하지 않는 기기도 어느 정도 제어 가능한 플랫폼은 Thread와 OIC이다. 예를 들어 OIC의 경우 플랫폼이 설치된 별도의 장치를 플랫폼 미사용 기기에 연결하는 형태로 전원을 켜고 끄는 등의 행동이 가능하다. 메모리나 연산 기능이 부족한 기기를 위해 더 작은 리소스를 사용하는 분할된 형태의 라이브러리는 AllJoyn, ANT+, oneM2M에서 제공한다. 분리된 라이브러리를 사용할 경우 전구와 같이 작은 기기에서 플랫폼 사용을 위해 필요로 하는 별도의 메모리를 더 적은 용량을 가진 것으로 사용할 수 있다.

Table 1. Comparison of IoT Systems

IoT 그룹 명	AllSeen Alliance	oneM2M	Thread Group	ANT+	OIC
제공 플랫폼 명	AllJoyn	oneM2M	Thread	ANT+	OIC
설립 년도	2013년 12월	2012년 1월	2014년 7월	2012년	2014년 7월
설립 목적	IoT 확산을 통해 IoT를 통한 삶의 많은 부분에 대한 유연함을 제공하기 위해	M2M 시스템의 연합과 상호 운영을 활성화 하기 위해	가정에서 새로운 무선망의 필요성에 따라	스포츠, 피트니스 및 건강 분야에서 업계를 선도하기 위해	운영체제와 서비스 공급자가 달라도 기기간 정보 관리,무선공유가 가능하도록 하기 위해
주요 기업	Qualcomm 등[1]	Samsung, LG 등[2]	ARM, Samsung 등[3]	ST Ericsson 등	Samsung, Intel 등[4]

3. 자가적응형 동향

3장에서는 자가적응형(Self-Adaptive)에 대한 개념을 설명하고, 기존에 연구된 자가적응형 분류체계에 대해 설명한다. 또한 자가적응형 프레임워크와 소프트웨어에 대해 알아본다.

3.1 자가적응형 개념

자가적응형은 변화하는 환경에 시스템 스스로가 판단을 하고 알맞게 적응하는 것을 의미한다. IoT 기술이 주목을 받으면서 환경에 대한 인지와 적응을 위해 자가적응형 역시 주목을 받고 있다. 이러한 이유로 각 분야에서는 다양한 개념과 적응방식에 대한 연구 및 조사 자료가 나오고 있다. IT에서의 자가적응의 개념을 살펴보면 '환경을 시스템 스스로 인식하여 그들의 행동에 대한 답을 내린다'[10]라고 정의를 내렸으며 '사람이 실행 중에 사람의 의사결정 없이도 가장 나은 대응을 스스로 판단하여 변화에 대처할 수 있도록 구현하는 것'[11]이라고도 한다. 또한, 그밖에 '유동적인 동작 환경에서도 주어진 목적을 달성하기 위해 소프트웨어 스스로 판단하여 행동방식을 변경하는 소프트웨어'[12], '소프트웨어가 운용되는 동안에 변화된 환경에 스스로 반응하며 소프트웨어 자체가 환경을 모니터링하고 변화를 탐지하며 어떻게 반응할 것인지를 결정하고 이러한 결정을 어떻게 실행시킬 것인지에 대한 요구'[13], '환경에 대한 인지를 바탕으로 스스로의 행동 또는 구조를 재구성할 수 있는 소프트웨어'[11] 등 다양한 정의가 제시되어 있다.

여러 논문들에서 제시한 자가적응형 개념들의 공통점을 살펴 보았을 때 중요한 키워드는 '변화하는 환경', '시스템 또는 소프트웨어가 스스로 판단 또는 행동', '변화에 대처'이다. 즉, 자가적응형은 사용자의 개입이 없이 소프트웨어 또는 시스템이 변화하는 환경을 인지 또는 모니터링 하고, 변화한 환경을 분석하여 알맞은 계획을 세우고 그에 맞게 시스템 또는 소프트웨어가 스스로 수행하는 것을 의미한다.

3.2 자가적응형 분류체계

자가적응형 분류체계란 자가적응을 포함할 수 있는 전체적인 체계를 설계한 것이다. 본 논문에서는 5W+1H 분류체계를 주요 분류체계로 설명한다.

5W+1H 분류체계는 5W인 When, Why, Where, Who, What과 12H인 How로 구성된다[13]. Table 2는 자가적응형을 분류할 수 있는 항목을 자세히 나타낸 표이다. 이하 5W+1H 각 항목에 대한 간략한 설명이다.

Table 2. 5W+1H taxonomy

When (Time)	Proactive	
	Reactive	
Why (Reason)	Change in the Context	
	Change in the Technical Resources	
	Change Caused by the User(s)	
Where (Level)	Application	Single Application
		Ensemble of Application
	System Software	Middleware
		(Operating) System
	Communication	Network Infrastructure
		Communication Pattern
	Technical Resource	
	Context	
What (Technique)	Parameter	
	Structure	
	Context	
How (Adaptation Control)	Approach	Internal
		External
	Adaptation Decision Criteria	Models
		Rules/Polices
		Goals
		Utility
	Degree of Decentralization	Decentralized
		Hybrid
		Centralized

a) When : 언제 자가적응이 일어나야 하는지에 대한 항목이다. 이 항목은 미리 알고 있는 상황과 예측하지 못하는 상황으로 분류된다.

b) Why : 왜 자가적응이 일어나야 되는가에 대한 항목으로 문맥이 변화할 경우, 하드웨어 장치의 변화에 대한 적응이 필요할 경우, 마지막으로 사용자 및 사용자 환경이 변화할 때로 구분한다.

c) Where : 어느 수준에서 자가적응이 일어나는가에 대한 항목으로 크게 응용프로그램, 시스템 소프트웨어, 통신,

1) QUALCOMM, LG, Microsoft, Panasonic, SHARP, SONY, ADT, HTC 등
2) Samsung, LG, ETRI, Adobe Systems, Hewlett Packard, HTC, IBM, Intel, Qualcomm, SKT, SONY 등
3) ARM, SAMSUNG, nest, PHILIPS, SILICON LABS 등
4) Samsung, Intel, ADT, DELL, HP, acer, ETRI 등

하드웨어, 문맥으로 나누어지고 각 항목은 각 항목에 세분화 되어 다시 나눠진다.

d) What : 무엇을 이용해서 자가적응을 하는 가에 관한 항목으로 매개변수, 구조, 문맥으로 나누어 설명이 가능하다.

e) How : 적응 조정에 관한 것으로 적응의 접근은 내부적인 접근인가 외부적인 접근인가로 구분되며 적응 결정 요인은 모델, 규칙/정책, 목적, 유틸리티로 적응이 되었나를 판단한다. 또한 적응이 일어날 경우 분산처리 시스템, 중앙집중 시스템, 모두 가능한가로 구분된다.

3.3 자가적응형 프레임워크

자가적응형 프레임워크는 자가적응형 소프트웨어의 설계와 구현에 있어, 재사용과 유지 보수의 효율성을 보장해준다. 대표적인 자가적응형 프레임워크로는 Rainbow 프레임워크, Tropos 프레임워크가 있다.

1) Rainbow 프레임워크

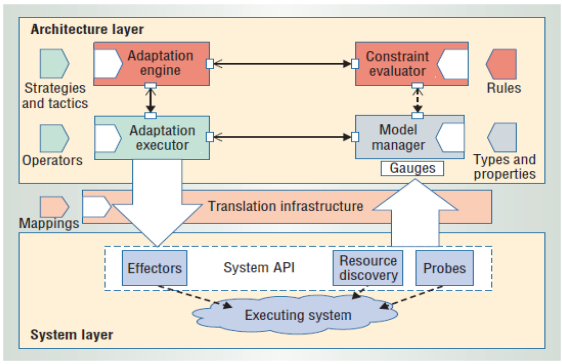

Fig. 1. Rainbow Framework [6]

외부적으로 적응을 조정하는 메커니즘은 분석, 수정, 확장 및 다른 시스템에서 재사용 할 수 있으며, 분리될 수 있는 모듈에서 문제를 감지하고 해결하기 때문에 내부적으로 적응을 조정하는 메커니즘보다 효율적이다[6]. 이러한 외부적인 접근을 구조 기반 자가적응이라 한다. 자가적응은 다른 시스템의 각기 다른 구조, 메커니즘을 다룰 수 있어야 하고 모니터링, 모델링, 문제 감지 메커니즘을 통해 최선의 적응 방법을 찾아야 한다. Rainbow는 모니터링과 모델을 평가하기 위한 프레임워크이다.

Fig. 1은 Rainbow 프레임워크의 동작방식 및 원리를 나타내는 그림이다. Rainbow 프레임워크의 'System layer'는 시스템 접근 인터페이스를 정의 한다. 'Probes'는 다양한 시스템의 상태를 측정하고 관찰하며 'Resource Discovery'는 자원 유형, 다른 결정 기준에 기반한 새로운 리소스에 대해 질의를 담당하고 있다. 또한, 'Effector'는 실제 시스템 수정에 대한 수행을 담당하며 'Translation infrastructure'는 아키텍처 요소 식별자를 IP주소로 바꾸거나 아키텍처 수준 정보의 중재를 돕는 기능을 한다. 'Gauges'는' 'Probes'로부터 정보를 모으고, 해당 아키텍처 모델에서 적절한 속성을 업데이트하며, 'Model manager'는 시스템 아키텍처 모델의 접근을 제공하는 역할을 한다. 'Constraint evaluator'는 모델을 주기적으로 검사하고 제약 조건에 어긋나면 적응을 트리거하는 기능을 제공한다. 'Adaptation engine'은 행동을 결정하고 필요한 적응을 적용하는 역할을 한다. 'Knowledge'는 시스템 구성 요소의 종류와 특성, 제약 조건 등의 정보로써 그림 양쪽에 있는 오각형의 모양에 해당된다. 이러한 기능들은 시스템이 자가적응을 하기 위해 상호협력을 통하여 적응을 가능토록 한다.

[4]에서는 자가적응 복합 시스템의 예시를 드는 경우에서 Rainbow 프레임워크를 사용하여 자가적응 시스템을 설명하였다.

2) Tropos 프레임워크

'Belief-Desire-Intention' 에이전트 모델을 통하여 프로세스와 자가적응시스템을 개발하기 위한 프레임워크를 정의하는 것을 목표로 한다[14]. 방법론적 프레임워크인 Tropos는 지능형 에이전트로서 초기 요구사항, 나중 요구사항, 구조설계, 상세 설계, 구현으로 구성되며 이를 통해서 행동, 목적, 계획, 자원, 가용성의 역할을 한다. 에이전트의 계획, 목표, 능력과 그들의 의존성에 대한 명세를 작성하고 이를 바탕으로 시스템을 구현한다. 그러나 작은 단위의 행동은 표현이 가능하나 규모가 커지고 다양한 에이전트의 관계에서는 설계와 예측의 어려움이 있다. [5]에서는 Tropos 프레임워크의 문제점을 해결하고자 새로운 관계를 도입하여 우선순위 형태로 표현하였다.

3.4 자가적응형 소프트웨어

자가적응형 소프트웨어는 '유동적인 동작 환경에서도 주어진 목적을 달성하기 위해 소프트웨어 스스로 판단하여 행동방식을 변경하는 소프트웨어'이다[12]. 즉, 변화하는 환경에 스스로 대처하여 알맞은 결정을 내리는 소프트웨어이다. 본 논문에서는 소프트웨어 내부에서의 자가적응, 환경인지를 통한 자가적응에 대해 살펴본다.

1) 소프트웨어 내에서의 자가적응

소프트웨어 내에서의 자가적응은 소프트웨어 스스로가 자신의 상태 및 상황을 인지하여 알맞게 적응 및 대처하는 것을 의미한다. 소프트웨어 내에서 자가적응의 대표적인 예로는 자가 치유를 볼 수 있다[15]. 소프트웨어 아키텍처 상에서 오류 탐지의 중요도는 컴포넌트마다 다르다. 각 컴포넌트마다 발생하는 오류의 심각도와 빈도가 다르기 때문에 모니터링 중요도가 높은 컴포넌트에는 강도가 높고 모니터링 중요도가 낮은 컴포넌트에는 강도가 낮도록 모니터가 적응한다면 오류탐지 부하를 줄일 수 있다. 자가치유는 이러한 신뢰성을 보장하기 위한 기술 중 하나이며, 소프트웨어 스스로 자신의 오류를 탐지하고 오류를 회복하는 것을 의미한다.

2) 환경인지를 통한 자가적응

IoT와 같은 기기들은 소프트웨어 스스로 사용자 및 사용자 환경을 인지하여 변화된 환경에 알맞게 소프트웨어의 상

태를 변화시켜 적응이 가능해야 한다. 변화하는 환경에 따른 대표적인 예로 웨어러블 기기가 있다. 웨어러블 기기는 사람의 신체에 착용하는 기기로서 사용자의 상황, 환경의 변화에 따라 소프트웨어 스스로 판단을 하여 적응을 하여야 한다. 환경인지에 관한 논문을 살펴보면, 웨어러블 기기간의 통신에 관한 자가적응을 설명한다[16]. 단말 조작이 서툰 사용자들의 경우 서비스를 이용하는데 제약이 있기 때문에 사용자들의 특별한 설정 없이 이웃하는 단말들의 자율적인 통신을 하기 위한 논문을 제시한다.

또한, 자가적응 소프트웨어의 변화하는 환경 설계 논문인 [11]에서는 스마트그리드를 예로 들어 변화하는 환경에 따른 자가적응형 소프트웨어에 관해 설명한다. 스마트그리드 시스템, 발전소, 가정집의 에이전트를 도출하고 시스템이 환경에 대한 인지를 바탕으로 전력 수급에 관련한 관리를 한다. 전력량을 인지하고 소프트웨어는 전력량에 따라 각 에이전트에 속한 시스템을 관리하면서 전력 부하를 안정적으로 제어한다.

4. 자가적응형 개념을 활용한 IoT 비교분석

4장에서는 3.2.1의 5W+1H 자가적응형 분류체계에 따라 Table 3에서 보는 바와 같이 IoT와 각 IoT 플랫폼을 분류하였다. IoT는 단순한 센싱기능을 하는 단일센서와 스마트폰과 같은 복합센서, 각 센서들의 데이터를 가공하고 처리하여주는 서비스, IoT기기들이 연결된 네트워크로 분류하였다. 단일센서, 복합센서, 서비스, 네트워크 모두 해당하는 특성을 가지고 있으면 O, IoT 내의 기준에 따른 내부 분류에서 모두 해당되지 않고 일부만 해당사항이 있을 경우 △로 표시하고 단일센서, 복합센서, 서비스, 네트워크를 각각 (단), (복), (서), (네)로 표시하였다. IoT 플랫폼의 경우 해당 IoT 플랫폼의 구성에 대해 간략히 설명한 후 Table 3에서 분류한 항목에 대해 설명한다.

4.1 IoT

IoT기기의 환경이나 사용자 환경이 변화하면 IoT 기기는 새로운 환경에 대한 네트워크에 연결을 하거나 현재 처한 상황에 대해 적응을 즉각적으로 해야 한다. 복합센서(스마트폰)나 서비스 같은 경우 미리 변화할 환경을 예정하는 것으로 설정이 가능하다. 단일센서는 자신의 콘텍스트(Context)를 가질 수 없기 때문에 콘텍스트에 따라 자가적응 하지 않는다. 하지만 복합센서나 서비스, 네트워크는 상태가 변경됨에 따라 자가적응이 일어나야 한다.

기술 자원(Technical Resource)은 하드웨어의 변화에 따라 자가적응이 일어나야 되는가에 대한 부류로 IoT 기기들은 새로운 센서가 부착되거나 새로운 복합센서가 추가 될 경우 서비스나 네트워크에서 이를 스스로 알고 적응을 시켜야 한다. 그러므로 하드웨어의 변화에 따른 자가적응은 모두 일어난다. 또한 사용자나 사용자 환경에 따른 환경변화에 대하여 모든 부분에서 자가적응이 필요하다.

어느 계층에서 자가적응이 일어나야 되는가에 대해서는 복합센서와 서비스는 어플리케이션(Application) 계층에서 일어난다. 네트워크와 단일센서는 자신의 응용프로그램을 가지고 있지 않기 때문에 응용계층의 자가적응이 일어나지 않는다. 또한, 시스템 소프트웨어 계층에서는 단일 센서를 제외한 복합센서, 서비스, 네트워크 모두 자가적응이 일어난다. 단일센서는 자신만의 운영체제나 미들웨어가 없기 때문에 시스템 소프트웨어 계층에서 자가적응이 일어나지 않는다.

IoT는 사물간의 통신을 해야 한다. 그러므로 통신계층에

Table 3. Self-Adaptive Classification

Self-Adaptive 분류		IoT	AllJoyn	ANT+	Thread	oneM2M	OIC
Proactive		△(복, 서)					
Reactive		O	O	O	O	O	O
Change in the context		△(복, 서, 네)					
Change in the Technical Resources		O	O	O	O	O	O
Change caused by User(s)		O	O	O			O
Application		△(복, 서)	O	O		O	O
System S/W		△(복, 서, 네)	O				O
Communication		O	O	O	O	O	O
Technical Resource		△(복, 네)		O	O	O	
Context		△(복, 서, 네)					
Parameter		△(단, 복, 서)	O	O	O	O	O
Structure		△(복, 서, 네)	O				
Context		△(복, 서, 네)					
Approach	Internal	△(복, 서. 네)	O			O	O
	External	O	O	O	O	O	
Adaptation Decision Criteria	Model	△(복, 서. 네)	O		O	O	O
	Rule/Policies	O				O	
	Goals	O		O		O	
	Utility functions	△(복, 서)					
Degree of Decentralization	Decentralized	△(복, 서. 네)	O		O		
	Hybrid	△(복, 서, 네)					
	Centralized	O	O	O			O

서는 모든 영역에서 자가적응이 일어난다. 기술 자원 계층에서는 새로운 하드웨어의 추가나 변경이 가능한 복합센서나 네트워크 같은 경우 자가적응이 일어난다. 콘텍스트 계층에서의 자가적응은 자신의 상태를 가지고 있는 단일센서를 제외한 복합센서, 서비스, 네트워크가 해당된다. 또한, IoT는 센싱된 정보나 가공하여 만들어낸 정보를 파라미터(Parameter)로 하여 자가적응이 가능하다. 또한 복합센서나 네트워크는 새로운 하드웨어를 통하여 자가적응을 할 수 있다. 마찬가지로 자신의 상태를 가지고 있지 않는 단일 센서는 포함되지 않는다.

IoT는 또한 내부적, 외부적 접근이 가능하다. 하지만 단일센서는 내부적으로 접근이 불가능하다. 적응의 결정 요인 중 모델은 단일센서를 제외하고 가능하다. 단일센서는 자신만의 모델을 정의하지 않고 단순한 기능만을 하는 센서기 때문에 제외된다. 하지만 변화한 환경에 따라 작동하는 규칙이나 목적이 같아진다면 적응을 하였다고 볼 수 있다. 단일센서는 또한 중앙 집중 처리 방식만을 선택한다. 단일센서는 중앙서버나 HUB를 통하여 자가적응을 한다.

4.2 AllJoyn

AllJoyn은 오픈소스로 소스가 공개되어있고, OS와 개발언어, 통신망에 독립적이다. 또한 RMI (Remote Method Invocation) 방식을 사용한다. AllJoyn의 주요 구성으로는 마샬링된 데이터를 분산 시스템으로 보내는 빠르고 간편한 방법을 제공하는 소프트웨어 버스나, 각각의 기기에 분산된 버스인 라우터가 있다. 각 기기나 어플리케이션은 라우터와 버스를 통해 서로 간 소통이 가능하다. 개발을 위한 프레임워크는 개발한 실제 코드인 'AppCode', 서로 다른 종류의 기기에 대한 인터페이스를 제공하는 'Service Frameworks', AllJoyn 네트워크와 상호작용을 위한 가장 기본이 되는 API를 제공하는 'Core Library'로 이루어진다.

AllJoyn의 경우 외부에서 새로운 기기가 AllJoyn 네트워크에 연결될 경우 자가적응이 일어난다. 핸드폰의 경우 사용자의 움직임에 따라 연결되는 네트워크가 변화하므로 유저에 의해 일어난다고 할 수 있고, 네트워크 망 자체가 변화할 수도 있으므로 기술 자원의 변화 때문이기도 하다. 이러한 변화는 어플리케이션, 프레임워크, 통신망에 걸쳐 이루어지며 파라미터를 통해 이루어진다. 또한, AllJoyn의 'Onboarding' 서비스를 통해 다른 기기를 자신의 네트워크로 데려올 수 있으므로 네트워크의 구조 변화도 있을 수 있다. 설정 변화는 내부적으로 스스로 처리되거나, 네트워크 망의 다른 기기가 처리할 수 있고 정해진 모델에 따라 이루어진다. 내부적으로 처리되거나 외부적으로 처리 가능하므로 중앙집중적일 때도 있고 분산적으로 처리될 때도 있다.

4.3 ANT+

ANT+ 네트워크는 전 세계로 퍼져있는 작은 네트워크들로 구성되어 관리된다. 네트워크들은 ANT+ 센서와 리시버만 있으면 어디서든 존재하고 찾을 수 있다. ANT+ 장치는 ANT+ 네트워크에 접근할 수 있는 키를 사용하여 네트워크에 접근할 수 있고 ANT+ 장치 프로파일 중 적어도 하나를 구현한다.

ANT+는 ANT 프로토콜을 사용하는 기기간의 통신이 가능하다. ANT+는 사용자나 사용자 환경이 변화하면 즉각적인 적응이 필요하다. ANT+는 예를 들어 자전거, 심박수 등을 측정하는 센서로 되어있다. ANT+가 자가적응을 해야 하는 이유는 사용자가 기존 ANT+가 연결되어있는 네트워크를 벗어날 경우 다른 ANT+ 네트워크에 연결이 되어야 하며, 새로운 센서들과의 통신이 필요하기 때문이다. 그러므로 ANT+의 응용프로그램 역시 센서에 따라 적응이 필요하며 기기들과의 통신을 위해 커뮤니케이션 수준에서도 적응이 필요하다. 또한 새로운 센서의 추가는 기술 자원 수준에서 일어나게 된다. ANT+는 기기의 정보와 센싱된 데이터들을 매개변수로 하여 전송을 하며, 이로 인하여 새로운 환경에 적응이 가능하다. ANT+는 기기들의 통신이 필요하다 그렇기 때문에 기기들은 외부적으로 연결이 된다. 또한, 각기 다른 센서들이 모여 같은 목적을 수행하게 된다. 다른 용도로 사용되던 센서가 현재 상황을 스스로 인지하여 목적을 바꾼다면 자가적응이 되었다고 볼 수 있다. 또한 각 ANT+기기들은 서버나 HUB로 센싱된 데이터를 보내어 데이터를 공유한다. 이러한 이유로 ANT+는 중앙집중 시스템이다.

4.4 Thread

Thread의 기본 시스템 구성 요소는 집에 있지 않아도 클라우드를 통해 기기 제어를 하기 위한 'Cloud Connectivity', 전달할 데이터를 클라우드를 통해 전달하는 'Border Router', 집에서 기기와 기기 사이의 통신을 지원하는 'Device Communication'로 이루어진다.

Thread는 가정 기기들을 네트워크에 연결하여 통신이 가능하게 한다. 그러므로 가정 기기가 연결되면, 사용자의 특별한 설정 없이 즉각적으로 자가적응이 일어나야 한다. Thread는 새로운 하드웨어 기기가 추가될 때 적응이 필요하며, 기기간의 통신과 새로운 하드웨어 기기에서 적응이 일어나므로 커뮤니케이션 수준과 기술 자원 수준에서 자가적응이 일어난다. 기기들은 기존의 파라미터를 변경하여 자가적응을 한다. 기기들 간의 연결을 외부적으로 하고, 모델을 기준으로 자가적응을 한다. 또한, 가정의 홈 네트워크에 연결된 기기들은 서버를 통하여 통신을 하게 되므로 중앙집중 시스템이다.

4.5 oneM2M

oneM2M의 주요 구성은 oneM2M 어플리케이션과 관련된 비즈니스 및 운영 로직으로 구성된 'Application Layer', oneM2M 어플리케이션에서 이용 가능한 oneM2M 서비스 함수로 구성된 'Common Services Layer', 전송 및 연결 서비스 함수를 제공하는 'Network Services Layer'로 구성된다.

oneM2M은 다른 기기와 연결되었을 때, 즉시 자가적응을 한다. 또한 응용 객체와 일반 서비스 객체를 통해 자가적응이 일어나므로 어플리케이션과 커뮤니케이션 수준에서 자가적응이 일어난다고 할 수 있다. 또, 기본적으로 기기간의 연결을 기본으로 하기에 기술 자원 수준에서도 일어난다고 할

수 있다. OneM2M 어플리케이션에서는 다른 기기와의 통신을 하기 때문에 외부적인 접근방식을 취한다. 또, 기기의 특성에 따라서 서버역할을 하는 기기를 중심으로 하는 중앙집중적인 구성과 그 기기들을 여러 개로 이은 시스템에서는 분산적인 시스템도 가능하다.

4.6 OIC [5]

OIC 프레임워크의 핵심 기능 중 하나인 'Identification and Addressing'은 OIC 식별자와 어드레싱능력, 결합 메커니즘을 제공한다. 'Discovery'는 사용할 수 있는 OIC 장치와 리소스에 대한 학습을 한다. 'Resource Model'은 리소스 측면에서 객체의 능력과, 조작을 위한 메커니즘이다. 'CRUDN'은 Create, Retrieve, Update, Delete, Notify로 리소스 조작을 담당한다. 'Device Management'는 OIC 장치의 기능을 관리, 설정, 모니터 하는 등의 기능을 수행한다. 'Group Management'는 OIC 리소스 그룹 생성 및 관리를 담당한다.

OIC의 자가적응은 AllJoyn의 방식과 유사하다. 네트워크 망에 새로운 기기가 연결되었을 때 자가적응이 일어나며 파라미터를 사용해 상태를 변화시킨다. 이 때 자가적응은 AllJoyn의 방식과 다르게 외부의 기기에서 설정이 불가하므로 내부적인 수준에서 일어나며, 이에 따라 분산적으로 처리된다고 할 수 있다.

5. OIC 자가적응형 요소 제안

5장에서는 4장의 비교 분석을 활용하는 예시로 OIC에서 부족한 자가적응형 요소에 대한 제안을 한다.

5.1 IoT 플랫폼 제공 서비스 분류

4장의 자가적응형 분류에 따른 비교는 직관적으로 이해하기에 어려움이 있다. 따라서 본 절에서는 4장에서 비교한 자가적응형 분류를 각 IoT 플랫폼이 제공하는 서비스를 기준으로 좀 더 구체적으로 명시한다. Table 4는 임의의 분류기준에 따라 비교한 것이다. 분류 기준은 각 IoT 플랫폼에서 지원해주는 서비스를 모두 목록화하여 자가적응형 분류 중 포함될 수 있는 것을 골라 각 시스템이 제공하는 서비스를 표시한 것이다. 용어가 다르더라도 동일한 서비스가 존재할 경우 하나의 시스템에서 사용된 용어 중 가장 명확한 명칭을 기준으로 정의한다. 제공하는 서비스가 해당 IoT 플랫폼에 명시적으로 설명되어 있을 경우 O, 명시적으로 제공하지 않는다고 되어 있을 경우 X, 알 수 없을 경우 빈칸으로 두었다. 구체적인 각 분류의 정의와 해당하는 자가적응형 분류는 Table 4와 같다.

기기 검색은 원하는 서비스를 가진 기기를 찾을 수 있는지에 대한 구분이고, 알림(Notification)은 알림을 모든 기기에 브로드캐스트할 수 있는지에 대한 부분이다. 네트워크 자동 접속은 주변의 해당 프레임워크의 네트워크로 기기나 어플리케이션이 자동으로 접속할 수 있는지에 대한 것이고, 원격 기기 설정은 프레임워크 네트워크 상의 기기를 외부의 기기에서 설정할 수 있는지에 대한 부분이다. 마지막으로 컨트롤 패널(Control Panel)은 자신을 제어할 수 있는 행동을 다른 기기에게 보내 자신을 동작할 수 있는 권한을 주는 것이다. 예를 들어 스토브의 경우 한 쪽 레버에 대한 동작을 UI 형태로 만들어 핸드폰과 같은 기기에서 UI를 통해 동작시킬 수 있도록 한다.

Table 4. Classification of Services Related to Self-Adaptive

구분	AllJoyn	Thread	ANT+	oneM2M	OIC
기기 검색	O		O	O	O
알림	O			O	O
네트워크 자동 접속	O		O		O
원격 기기 설정	O	O		O	X
컨트롤 패널	O				X

AllJoyn의 경우 'Unique Name'을 통해 특정 기기를 검색 가능하고, 두 가지로 세분화된 알림 서비스를 지원하며 메시지를 받는 쪽에서 무시할 수 있는 기능도 존재한다. 또한 'Onboarding' 서비스를 통해 네트워크 자동 접속을 지원하며 'Configuration' 서비스를 통해 원격 기기 설정을 지원한다. 컨트롤 패널 서비스 또한 존재한다.

Thread의 경우 스마트폰에서 한 기기에 명령을 보내 전원 제어가 가능하므로 원격 기기 설정을 일부지만 지원한다.

ANT+는 ANT 네트워크에 기기가 접속되면 자동 접속이 되며, 이를 통해 기기 검색 또한 가능하다.

OneM2M의 경우 네트워크에 연결된 장치에 대한 검색이 가능하며 알림의 경우 알림을 받기를 원하는 구독자(subscriber)에게 해당 주소로 알림을 전송한다. 'DMG CSF'는 네트워크 내에 있는 장치 'MN, ASN, AND'의 기능뿐만 아니라 장치의 관리 기능도 제공하므로 원격 기기 설정이 가능하다.

OIC의 경우 'Retrieve' 기능을 통해 원하는 리소스 검색이 가능하고 알림 서비스 또한 제공한다. 'Onboarding' 서비스를 통해 네트워크 자동 접속 또한 지원하며, 원격 기기 설정과 컨트롤 패널 기능은 제공하지 않는다.

5.2 OIC 자가적응형 요소 제안

5.1과 같은 구체적인 분류를 통해 OIC에서 부족한 부분에 대한 제안을 한다. OIC와 나머지 IoT 플랫폼의 기능을 종합적으로 비교한 결과, 나머지 IoT 플랫폼에서 지원하는 기능을 OIC에서 지원하지 않는 기능은 총 2가지로 분석되며 이를 제안한다.

첫 번째로, 새로운 장치가 네트워크에 연결되었을 시 네트워크 상황에 맞게 외부에서 설정이 진행되는 기능을 제안한다. 새로운 장치가 네트워크에 접속되었을 경우 스스로 설정이 불가능한 작은 센서나 구동물품은 네트워크 상에 존재하는 다른 기기에서 원격 설정이 가능할 수 있어야 한다. 예를 들어 단순히 빛의 세기가 조절 가능한 전구가 네트워크에 접속되었을 때 현재 방의 밝기에 맞춰 너무 밝거나 너무 어둡지 않게 조절하는 경우가 있다. 이 때 전구는 현재

5) 논문 작성 시점에서 개발 진행중인 내용을 기준으로 한다.

방의 밝기를 알 수 없으므로 스스로 설정을 할 수 없다. 따라서 조도 센서가 달려있는 네트워크 상의 핸드폰 등의 기기가 정보를 주어 설정이 진행될 수 있어야 한다. AllJoyn 프레임워크의 경우 'Onboarding'과 'Configuration' 서비스를 통해 이와 같은 동작이 가능하다.

첫 번째 제안의 경우 Fig. 2와 같이 동작한다. 자신을 설정해주길 원하는 기기는 알림 서비스를 사용해 브로드캐스트로 네트워크 상 다른 기기에게 자신의 설정 데이터를 알린다. 알림을 받은 기기는 해당 기기를 자신이 지원 가능한지 체크한 후 해당 기기에 접속해 설정을 진행한다. 최종적으로 설정을 변경하여 과정이 끝나게 된다.

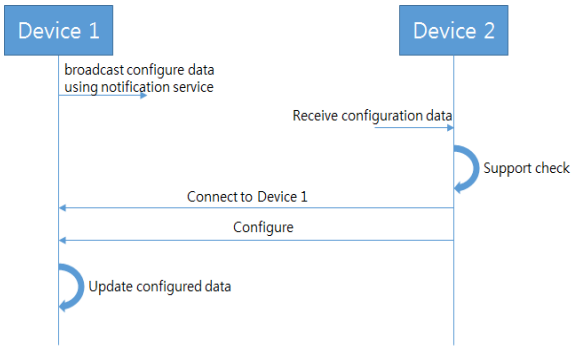

Fig. 2. Configuration Other Device Flow Diagram

두 번째로, AllJoyn의 컨트롤 패널과 유사한 형태의 서비스를 제안한다. 기존 OIC 시스템 프레임워크에서 제안하는 요소를 적용한 프레임워크는 Fig. 3과 같다. 이 때 OIC Framework 그룹에서 기존에 제공되는 요소는 흰색 네모 박스로, 추가된 요소는 회색 네모 박스로 나타낸다. 흰색 박스로 되어 있는 기존의 시스템은 OIC 식별자와 어드레싱능력 결합 메커니즘을 정의하는 'ID & Addressing', 사용할 수 있는 OIC 장치와 리소스에 대한 학습을 위한 'Discovery', 리소스의 측면에서 개체의 표현 능력을 명시하고 리소스를 조작하기 위한 메커니즘을 정의하는 'Resource Model', 리소스 관리를 위한 메시지 교환에 대한 일반적인 방법을 제공하는 'CRUDN', 메시지 프로토콜을 제공하는 'Messaging', 단방향 스트리밍 방식을 제공하는 'Streaming' 장치의 준비 및 초기 설정을 진행하는 'Device Management', 그룹을 만들고 관리하는 'Group Management', 보안 액세스에 필요한 기능과 프로세스를 제공하는 'Security'로 이루어진다. 회색으로 된 추가 요소를 OIC Framework에 도입함으로써 본 논문에서 제안하는 두 가지 사항을 만족할 수 있다.

두 번째 제안의 경우 Fig. 4과 같이 동작한다. 자신에 대한 조정 정보를 담은 메시지를 브로드캐스트를 통해 네트워크 상 다른 기기에게 알리고 이 메시지를 받은 기기는 이를 지원할 것인지 확인한다. 지원할 경우 사용자가 조정할 수 있도록 화면상에 띄우거나, 어플리케이션에서 바로 조정 가능하다. 이 때 사용자나 프로그램이 해당 기기를 사용하고자 하면 명령 메시지를 보내 해당 기기에서 실행될 수 있도록 한다.

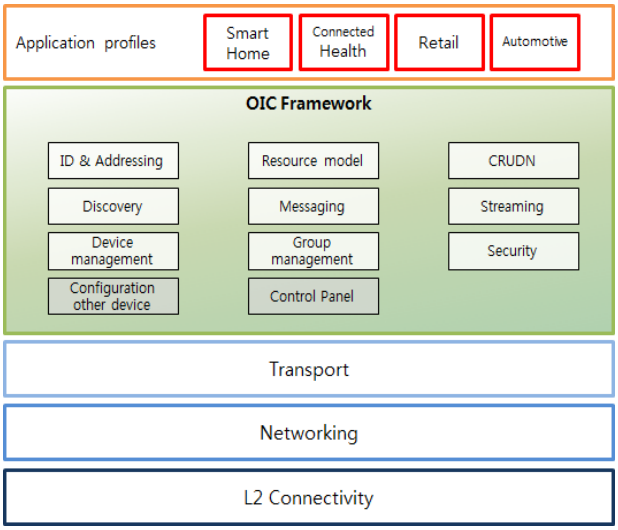

Fig. 3. Modified OIC Framework Architecture

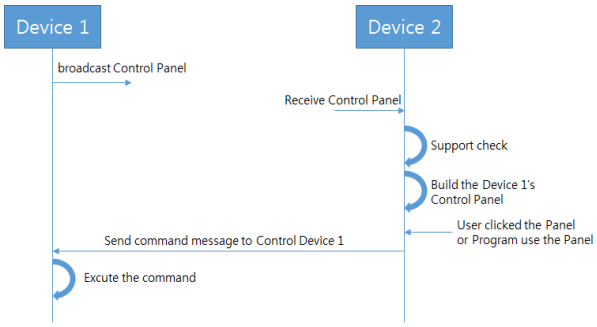

Fig. 4. 'Control Panel' Flow Diagram

6. 결론 및 향후 연구

본 논문에서는 기존의 IoT의 연구 동향과 AllJoyn, oneM2M, ANT+, Thread, OIC의 개발 동향에 대해 알아보았다. 또한 자가적응형 연구와 개발 동향에 대해 살펴보고 IoT와 IoT 플랫폼에 자가적응형 분류체계를 사용한 비교 방식을 제안했다. 이러한 적용 방식을 적용해 본 결과 자가적응형과 관련된 사항을 빠짐없이 기술하고 비교할 수 있었다.

이를 활용한 예시로 개발 단계인 OIC의 IoT 플랫폼에서 부족한 자가적응형 요소에 대한 제안을 하였다. 제시한 요소는 새로운 장치가 네트워크 상에 연결되었을 경우 다른 기기에서 원격으로 설정을 진행할 수 있도록 해주는 기능과 서비스를 제공하는 기기가 능동적으로 자신의 서비스를 제공하는 기능이다. 이러한 기능을 추가함으로써 자가적응형 분류체계에서 빠진 부분을 어느 정도 보완할 수 있게 된다.

IoT와 자가적응형은 현재도 지속적으로 연구가 진행되고 있는 분야로 새로운 사항에 대한 지속적인 비교, 분석이 필요하다. 또한 OIC는 아직 개발중인 IoT 플랫폼으로 지속적인 제안과 연구가 필요하다.

References

[1] D. Bandyopadhyay and J. Sen, "Internet of things: applications and challenges in technology and standardization," *Wireless Personal Communications*, Vol.58, No.1, pp.49-69, 2011.

[2] L. Atzori, A. Iera, and G. Morabito, "The internet of things: a survey," *ComputerNetworks*, Vol.54, No.15, pp.2787-2805, 2010.

[3] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelffle, "Vision and challenges for realizing the internet of things," *Cluster of European Research Projects on the Internet of Things*, pp.229, 2010.

[4] M. W. Kim and H. P. In, "The Concept and an Example of Self-Adaptive System of Systems," *College of Information and Communications*, 2014.

[5] M. Morandini, L. Penserini, and A. Perini, "Towards Goal-Oriented Development of Self-Adaptive Systems," *SEAMS '08 Proceedings of the 2008 International Workshop on Software Engineering for Adaptive and Selfmanaging Systems*, pp.9-16, 2008.

[6] S. W. Cheng, A. C. Huang, and B. Schmerl, "Rainbow: Architecture-Based Self-Adaptation with Reusable Infrastructure," *IEEE on Computer*, Vol.37, No.10, pp.46-54, 2004.

[7] C. Perer and D. Georgakopoulos, "Context aware computing for the internet of things : A survey," *IEEE Communications Surveys & Tutorials*, 2013.

[8] W. J. Joe, M. Jiang, and K. Jeong, "An M2M/IoT based Smart Data Logger for Environmental Sensor Networks," *Journal of KIISE*, Vol.20, No.1, pp.1-5, 2014.

[9] J. Ko, S. G. Hong, B. B. Lee, and N. S. Kim, "Trends of Converging Smart Devices with IoT Technology," *Electronics and Telecommunications Trend, ETRI*, Vol.28, No.4, 2013.

[10] H. C. Betty, R. Lemos, H. Giese, P. Inverardi, and J. Magee, "Software Engineering for Self-Adaptive Systems: A Research Roadmap," *Berlin: Springer-Verlag*, pp.1-26, 2009.

[11] S. H. Kim and S. W. Lee, "Designing behavior of self-adaptive software in aspect of agent-based software engineering," *Journal of KIISE*, pp.451-453, 2014.

[12] W. E. Hong, D. H. Kim, and H. P. In, "Situation-based Behavioral Modeling for Self-Adaptive Software," *College of Information and Communications*, 2014.

[13] S. Cho and H. In, "A M&S based system configuration method for Self-Adaptive System of Systems," *College of Information and Communications*, 2014.

[14] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini. "Tropos: An Agent-Oriented Software Development Methodology. Autonomous Agents and Multi-Agent Systems," *Autonomous Agents and Multi-Agent Systems*, Vol.8, No.3, pp.203-236, 2004.

[15] Team Barrelfish, "Barrelfish Project," *Systems Group: Department of Computer Scinece ETH Zurich*, 2013.

[16] H. Hoffmann, J. Holt, G. Kurian and E. Lau, "Self-aware Computing in the Angstrom Processor," *IEEE on Design Automation Conference*, pp.259-264, 2012.

황 세 영

e-mail : nahwasa@gmail.com

2007년 한국외국어대학교 정보통신공학과 (학사)

2013년~현 재 한국외국어대학교 정보통신공학과 DISLab 연구원

2014년~현 재 한국외국어대학교 컴퓨터 및정보통신공학과 석사과정

관심분야 : Database, Mobile Application, Context Awareness, Flass Memory

서 장 일

e-mail : duty5208@naver.com

2010년 한국외국어대학교 정보통신공학과 (학사)

2014년~2015년 한국외국어대학교 정보통신공학과 DISLab 학부연구원

2016년~현 재 GS ITM 경영정보실 사원

관심분야 : IoT, Database System

박 성 준

e-mail : apdanum@naver.com

2010년 한국외국어대학교 정보통신공학과 (학사)

2014년 한국외국어대학교 정보통신공학과 DISLab 학부연구원

2015년~현 재 신도리코 경영정보실 사원

관심분야 : IoT, Database System

박 상 원

e-mail : swpark@hufs.ac.kr

1994년 서울대학교 컴퓨터공학과(학사)

1997년 서울대학교 컴퓨터공학과(석사)

2002년 서울대학교 컴퓨터공학과(박사)

2002년~2003년 세종사이버대학교 디지털콘텐츠학과 전임강사

2010년~현 재 한국외국어대학교 정보통신공학과 교수

관심분야 : Flash Memory, Embeded Database, Mobile Computing