碩士學位論文

OSGi를 위한 실시간 데이터 처리 방법

濟州大學校 大學院

컴퓨터工學科

車 祉 潤

2010年 02月

OSGi를 위한 실시간 데이터 처리 방법

指導敎授 邊 暎 哲

車 祉 潤

이 論文을 工學 碩士學位 論文으로 提出함

2010年 02月

車祉潤의 工學 碩士學位 論文을 認准함

審査委員長 _____ 印

委 員 _____ 印

委 員 _____ 印

濟州大學校 大學院

2010年 02月

Method of Real-time Sensor Data Stream for OSGi Frameworks

Ji-Yun Cha

(Supervised by professor Yung-Cheol Byun)

A thesis submitted in partial fulfillment of the requirement for the degree of Master of Computer Engineering

2010. 02.

This thesis has been examined and approved.

Thesis director, _____

Thesis director, _____

Thesis director, _____

February 2010

Department of Computer Engineering

Graduate School

Jeju National University

감사의 글

2년 전 연구실에 처음 발을 내딛을 때부터 계절이 바뀌는 것도 잊은 채 시작한 2년 동안의 석사과정이 지나 어느덧 석사논문의 한쪽에 감사의 글을 전하게 되었습니다. 돌이켜 생각해보면 2년이라는 시간이 살아온 지난날 중에서 가장 짧은 순간이 아닐까 생각합니다. 시간은 짧았지만 앞으로 살아 갈 날을 위한 소중한 경험들이였고 추억들을 만들었습니다. 제가 그동안 잘 생활 할 수 있도록 많은 도움을 주신 분들께 감사의 마음을 이 좁디좁은 종이에 표현 할 수 없지만 진심으로 감사하다는 말을 전하고 싶습니다.

무엇 보다 철없고 반항적이며 비판적인 저를 이해 해주시고 또한, 학문의 길을 터주신 탐라대학교 컴퓨터공학과에 현창문 교수님께 감사의 마음을 전하고 싶습니다. 부족한 저에게 격려와 용기를 주시고 논문이 완성되기 까지 세심힌 배려와 지도를 아낌없이 해주신 저의 지도교수님께 감사드립니다. 더불어 많은 연구와 동기와 지식을 전달 해주신 김장형 교수님, 안기중 교수님, 곽호영 교수님, 변상용 교수님, 이상준 교수님, 송왕철 교수님, 김도현 교수님께 감사드립니다.

아무것도 모르던 저에게 학업에 대해 조언과 도움을 주신 박사과정 노영식 선배님, 양문석 선배님, 변지웅 선배님, 그리고 연구실에서 같이 생활하면서 많은 도움을 준 있었던 고기붕, 한대오와 학부생 후배님들에게 감사의 말을 전합니다. 그리고 저를 물심양면으로 많은 도움을 주신 송수경 선배님, 한경복 박사님, 김정희 박사님, 김남식 선배님, 김용우 선배님, 부문수 형, 석사동기 이창영과 컴퓨터공학과 사무실에 정은경 선생님, 이정하 선생님께 감사의 말을 전합니다.

끝으로 힘든 일이 있으면 항상 도와주었던 저의 누님인 인선, 지선 누님과 15년 죽마고우인 학이, 정우, 성우, 정철이, 노을이 학부생 선배이면서 저에게 많은 도움을 주신 송수경 선배님, 학부생 때 함께 공부하며 밤을 새웠던 동훈이 형, 경보, 봉석이에게 감사하며 끝으로 어머님께 사랑하고 감사한다는 말을 남깁니다.

목 차

그림 목차

표 목차

국문초록

OSGi를 위한 실시간 데이터 처리 방법

컴퓨터공학과 차지윤
지 도 교 수 변영철

유비쿼터스 컴퓨팅을 실현하기 위하여 센서 네트워크, 홈 네트워크, U-City, 텔레매틱스 등 다양한 분야에서 연구가 진행되고 있다. 특히 홈 네트워크 환경에서는 이질적인 하드웨어 플랫폼, 네트워킹, 프로토콜, 미들웨어 등 다양한 기술들이 존재한다. 홈 네트워크에서는 RFID 및 환경정보 센서들이 추가되어 실시간 센서 데이터 스트림을 활용한 방범, 화재경보, 가스누출, 침입탐지 등의 응용에 대한 연구가 진행되고 있다. 또한, 다양한 디바이스들이 존재하는 홈 네트워크에서 OSGi는 로컬 네트워크상의 상호 호환성을 보장하고, 각 하드웨어에서 관리되는 서비스들의 배포 및 공유에 대한 플랫폼을 제공하기 때문에 현재 실시간 센서 데이터를 활용한 OSGi 기반 홈 네트워크 응용에 대한 연구가 진행되고 있다. 그러나 현재 OSGi를 이용한 홈 네트워크 응용에서는 RFID/USN 실시간 센서 등을 이용한 실시간 센서 데이터스트림 처리에 대한 고려가 미진하다. 또한, OSGi 기반 응용 서비스들은 실시간 센서 데이터 처리하여 사용하기 위해 특정 응용에 종속적인 미들웨어를 사용하고 있다. 따라서 OSGi를 이용한 홈 네트워크 환경에서 개발자로 하여금 실시간 센서 데이터스트림 생성 장치를 효과적으로 활용하여 OSGi 응용을 개발할 수 있도록 하기 위한 아키텍처 및 API 제공 방법 등에 관한 연구가 필요하다.

본 논문은 특정 OSGi 기반 응용 서비스에 종속적이지 않으며, OSGi 프레임워크로 제공되는 다양한 형태의 실시간 센서 데이터 스트림을 필터링, 그룹핑, 그리고 카운팅 등 효과적으로 처리하기 위한 방법을 제안한다. 본 논문의 연구를 통하여 홈 네트워크에서 효과적으로 처리하지 못했던 실시간 센서 데이터 스트

림 처리를 가능하게 하고 홈 네트워크에 다양한 실시간 센서가 도입되거나 확장될 때 별도의 시스템 도입 없이 기존의 국제 표준 미들웨어를 사용하여 현재 홈 네트워크의 단점을 보안하고 경제적인 효과를 가질 수 있다.

ABSTRACT

Method of Real-time Sensor Data Stream for OSGi Frameworks

CHA, JI-YUN
Department of Computer Engineering
Graduate School
Jeju National University

Nowadays, to realize ubiquitous computing environment, many research activities have been going on within various kinds of research domains including sensor network, home network, U-City, and telematics. Specially, in an environment of home network where a number of technologies including heterogeneous hardware platforms, networking and protocols, middleware systems, and etc., exist, OSGi provides a platform for deployment and sharing of services managed in hardware and guarantees compatibility among applications. However, only simple control and processing of event data is considered in a home network using OSGi, and the consideration about real-time processing of data stream generated by sensors is not enough. Additionally, OSGi-based application services using a particular application are dependent on the middleware. Therefore, researches allowing users to effectively develop OSGi applications by using various kinds of sensors generating data streams are needed. In this paper, and we propose an effective method of processing various types of real-time sensor data streams in OSGi applications. The proposed system has some operations including filtering, grouping, and counting, and does not depend on a specific OSGi-based application service, as well.

약어표

ALE	Application Level Event
API	Application Program Interface
EPC	Electronic Product Code
GSM	Global System for Mobile Communications
HAVi	Home Audio Video Interoperability
IrDA	Infrared Data Association
MIDP	Mobile Information Device Profile
OMA-DM	Open Mobile Allience for Device Management
OSGi	Open Services Gateway initiative
PLC	Power Line Communication
PML	Physical Markup Language
RCP	Rich Client Plaform
RFID	Radio Frequency Identification
SNMP	Simple Network Management Protocol
SOA	Service Oriented Architecture
SSH	Secure Shell
UPnP	Universal Plug and Play)
USB	Universal Serial Bus
USN	Ubiquitous Sensor Network
WAP	Wireless Application Protocol
XML	eXtensible Markup Language

I. 서 론

1. 연구 배경

유비쿼터스 컴퓨팅은 다양한 컴퓨터가 현실 세계의 디바이스, 환경 및 사물들 속으로 스며들어 언제, 어디서나, 어떠한 기기로도 통신 서비스를 이용할 수 있는 인간, 사물, 공간 간의 최적의 컴퓨팅 및 네트워킹 환경을 구축함으로써 생활 속에서 자연스럽고 편리하게 컴퓨터를 사용할 수 있는 기술을 의미한다. 실생활에서 유비쿼터스 컴퓨팅의 대표적인 예가 홈 네트워크이다[1][2].

홈 네트워크는 가정과 정보 네트워크를 연결함으로써 정보를 효율적으로 사용할 수 있도록 지원하며, 각종 공공 서비스 및 사회 서비스(금융·의료 등)에 직접적으로 연결하여 서비스를 제공할 수 있도록 하고 홈 게이트웨이나 홈 서버를 이용하여 정보통신기기, 디지털AV기기 및 기존 가전기기 등을 통합적으로 제어함으로써 가정생활의 편리함과 효율성을 극대화 했다[3].

홈 네트워크의 하드웨어 플랫폼, 컴퓨팅, 네트워킹, 프로토콜, 미들웨어, 서비스 등 이질적인 환경에서 OSGi(Open Service Gateway Initiative)는 로컬 네트워크 상에서 상호 호환성을 보장하고, 각 디바이스에서 관리되는 서비스들의 배포 및 고유에 대한 공개스펙을 정의하며, 장비 연결 및 제어로 얻을 수 있는 유효 서비스의 배포 문제를 해결하고 서비스가 작동하기 위한 제반환경을 제공하여 홈 네트워크를 위한 서비스들을 쉽게 개발할 수 있게 함으로써 이질적인 환경에서 발생하는 문제점들을 극복할 수 있다[5].

홈 네트워크의 유비쿼터스 환경을 실현하기 위하여 RFID(Radio Frequency Identification)/USN(Ubiquitous Sensor Network)등의 관련기술 활용이 필수적이다. RFID는 각종 물품에 소형 칩을 부착하여 사물의 정보와 주변 환경정보를 무선 주파수로 전송·처리하는 비접촉식 인식기술이며, USN은 필요한 모든 곳에 센서노드를 부착하고, 이를 통해 사물의 인식 정보를 기본으로 주변의 환경 정보(온도, 압력, 오염, 균열 등)까지 각종 센서를 통해 실시간 수집하여 관리, 통제할

수 있도록 구성한 네트워크이다[4].

RFID/USN 기술을 이용한 유비쿼터스 응용서비스가 홈 네트워크에 적용됨에 따라 객체와 응용 서비스 사이에서 교량 역할을 하는 미들웨어의 필요성이 증대되고 있다. RFID/USN 미들웨어는 이기종 운영체제 간 상호협력이 가능하며, 분산처리의 신뢰성, 네트워크의 독립성, 응용 프로그램 및 서비스간의 상호운용성 및 투명성을 지원한다. 또한 여러 가지 센서를 관리하며 센서의 프로토콜을 이용하여 데이터를 수집하고, 수집되어 가공되지 않은 데이터로부터 의미 있는 정보, 혹은 응용이 사용하기 쉬운 형태의 정보를 추출하여 응용 서비스에 전달하는 기능을 수행 한다.

홈 네트워크에서는 RFID 및 환경정보 센서들이 추가되어 실시간 센서 데이터 스트림을 활용한 방범, 화재경보, 가스누출, 침입탐지 등의 응용 서비스를 제공할 수 있다. 홈 네트워크에서 방범, 화재경보, 가스누출, 침입탐지와 같은 응용 서비스들은 초당 수백 건의 실시간 데이터를 처리해야 한다. 하지만 OSGi 기반의 기존 홈 응용 서비스들은 단순한 제어와 이벤트성 데이터 처리에 대해서만 고려되고 있어, 실시간 센서 데이터 스트림을 이용한 다양한 응용에 대한 연구가 미진하다. 또한, 특정 OSGi 기반 응용 서비스에 종속되는 비표준 형태의 실시간 센서 데이터 스트림 처리만 고려되고 있다. 즉, 홈 네트워크 내·외부에서 RFID/USN를 이용한 OSGi 기반 응용 서비스에서 표준을 따르는 실시간 센서 데이터 스트림 처리에 대한 고려가 미미한 상태이다.

따라서 OSGi 플랫폼과 국제표준을 따르는 실시간 센서 데이터 스트림 처리 미들웨어의 연동을 통해 OSGi를 이용한 홈 네트워크 환경에서 OSGi 응용 서비스 개발자로 하여금 실시간 센서 데이터 스트림 처리 장치를 효과적으로 활용하여 OSGi 플랫폼 기반 응용을 개발할 수 있도록 하기 위한 아키텍처 및 API 제공 방법 등에 관한 연구가 필요하다.

2. 연구 목적 및 방법

현재 홈 네트워크에서 OSGi와 관련된 응용 서비스들은 실시간 데이터 처리에 대한 고려가 미진하다. 따라서 본 논문에서는 OSGi 플랫폼과 국제 표준을 따르는 실시간 센서 데이터 스트림 처리 미들웨어를 연동하여 OSGi 기반 홈 네트워크 응용 서비스에서 실시간 센서 데이터를 효과적으로 처리하기 위한 방법을 제안한다. 제안하는 방법은 OSGi 플랫폼과 ALE 기반의 실시간 센서 데이터 스트림 처리 미들웨어 사이의 계층을 설계하고 OSGi 기반 응용 서비스들이 이 계층 위에 존재하여 OSGi 응용 서비스가 원하는 실시간 센서 데이터를 효과적으로 획득할 수 있도록 하는 것이다. 즉, 다양한 OSGi 응용 서비스들이 본 논문에서 제시한 OSGi 응용 프레임워크를 사용함으로써, 국제 표준에 따라 처리된 실시간 센서 데이터를 효과적으로 사용할 수 있도록 하는데 그 목적이 있다. 번들 형태로 존재하는 OSGi 응용을 위한 프레임워크 구현을 위하여 다음과 같은 방법으로 연구한다.

첫째, 기존의 사실상의 국제 표준을 따르는 ALE 기반 실시간 센서 데이터 스트림 처리 미들웨어의 구조를 분석하여 미들웨어 내부의 실시간 센서 데이터 스트림 처리 과정을 파악한다. 이는 미들웨어를 새롭게 구현하는 것은 아니며 이미 구현되어 있는 미들웨어에 대한 분석이다. 둘째, 미들웨어 분석결과를 바탕으로 미들웨어의 여러 모듈 중 데이터 처리의 핵심 모듈인 실시간 센서 데이터 요청 명세서를 받아들여 분석하는 모듈, 실시간 센서 데이터를 수집하여 필터링, 그룹핑 하는 모듈, OSGi 플랫폼 기반 홈 네트워크 응용에게 전송하기위해 결과 명세서를 생성하는 모듈을 분석한다. 이는 OSGi 플랫폼으로 제공되는 다양한 형태의 실시간 센서 데이터를 응용 개발자가 원하는 형태로 쉽게 가공할 수 있도록 필터링, 그룹핑 등을 효과적으로 수행할 수 있는 방법에 대한 연구이다. 셋째, 실시간 센서 데이터 스트림 처리 미들웨어와 OSGi 플랫폼 기반 응용 시스템이 효율성과 유연성을 가질 수 있도록 API 및 인터페이스 방법에 대해 연구한다. 최종적으로 실시간 센서 데이터를 이용하여 간단한 제어 기능을 지닌 홈 네트워크 기기 제어 응용을 OSGi 번들 형태로 구현하고, 이 OSGi 기반 응용에서 실시간

센서 데이터 스트림 처리 미들웨어에게 데이터를 요청하고, 그 결과 값을 이용하여 홈 네트워크 기기를 제어하는 시나리오를 바탕으로 테스트 및 최종 결과를 분석 정리한다.

3. 논문 구성

본 논문의 구성은 다음과 같다. II장에서는 관련 연구 및 기술로 OSGi 프레임워크 플랫폼의 동향과 특징, RFID/USN 미들웨어의 기술, 사실상의 국제표준 스펙인 EPCglobal의 ALE 미들웨어 와 ALE 미들웨어에서 다양한 유형의 실시간 센서 데이터 처리, 연구되고 있는 OSGi 기반 응용 서비스와 이 응용에서 사용하고 있는 미들웨어에 대해 설명하며, III장에서는 실시간 센서 데이터 스트림 처리 미들웨어를 분석하여 OSGi 플랫폼과 연동하기 위한 방법에 대해 설명한다. IV장에서는 III장에서 제안한 방법의 구현 및 실험한 결과를 살펴보고, 마지막 V장에서는 본 연구의 결론에 대하여 설명한다.

II. 관련 연구 및 고려사항

1. OSGi 서비스 플랫폼

1) OSGi 배경과 현황

1990년대 후반에 홈 네트워크가 확산되면서 이를 위한 각종 장비들이 출시됨에 따라 집 내부와 같은 로컬 네트워크(Local Network) 환경에서 각각의 장비들이 인터넷에 연결되게 되었다. 장비들이 많아지면서 각각의 장비들을 연동할 필요가 생기면서 이런 장비들 간에 통신을 할 때의 호환성 문제가 대두 됐다. 그런 장비들 간의 상호 호환성을 보장하고, 나아가 공개적인 명세가 필요하다는 의견이 모여 탄생된 기술이 바로 OSGi 이다. 1999년 3월에 IBM, 썬, 노키아, 삼성전자 등의 업체들이 모여 OSGi Alliance(http://www.osgi.org)라는 비영리 단체를 만들었다. OSGi는 네트워크상에 연결된 디바이스들이 다양한 서비스를 공유할 수 있도록 하는 자바 언어 기반의 동적인 플랫폼을 만들기 위해 'Open Services Gateway initiative'라는 이름으로 시작되었고, 현재 OSGi라고 불리고 있다. 하지만, 이 약어 이름 자체로도 OSGi라는 단어가 어떤 기술을 의미하는지 알 수가 없어 종종 'Dynamic Module System for java'라고 하는 부제가 따라 붙는다. OSGi Alliance 단체를 통해 OSGi 공개 표준 명세(Specification)를 만들기 시작하였다. OSGi는 스펙이 공개된 기술이며, 모든 개발 및 스펙 제정 활동은 OSGi Alliance에 의해서 승인되고 진행된다. 현재 홈 네트워크의 기능을 크게 가전기기의 상태 정보/모니터링, 기기의 원격 제어컨트롤, A/V 및 주방가전의 홈 솔루션의 통합으로 구분하고 있는데, 가정의 구성된 네트워크에 속한 단말 또는 가전에 접근하기 위해서는 디바이스에 대한 표준화된 네트워킹과 범용 미들웨어가 필요하다. 이로 인해 나온 것이 오디오/비디오 기기에서 IEEE1394 선을 근간으로 필립스가 주도하는 HAVi(Home Audio Video Interoperability), PC와 같은 정보기기에는 HomePNA나 Ethernet 라인을 통하여 정보를 주고받을 수 있게 하는

MS에서 주창한 UPnP(Universal Plug and Play), 냉장고나 마이크로웨이브 오븐 같은 간단한 컨트롤만 요하는 가전제품에는 에셜론 사의 론워크 같은 PLC(Power Line Communication), Jini 등의 표준이다. 이러한 장비 연결 및 제어로 얻을 수 있는 유효한 서비스로는 그림 1과 같은 다양한 서비스들이 있으며, OSGi의 경우 그런 것을 다 지원해줄 수 있다. 즉, 이러한 서비스의 배포문제와 서비스가 작동하기 위한 제반 환경 제공을 목표로 하는 것이 OSGi의 기본 목표라고 할 수 있다. 또한, OSGi는 기존의 자바 플랫폼이 제공하지 못하는, 동적 컴포넌트 모델을 지원하는 프레임워크라고 볼 수 있다. 초기에는 홈서비스 게이트웨이에 집중되었지만 최근에는 특정 네트워크 환경에 국한하지 않고 유비쿼터스 환경까지 확장해 가고 있다[4][5][6].

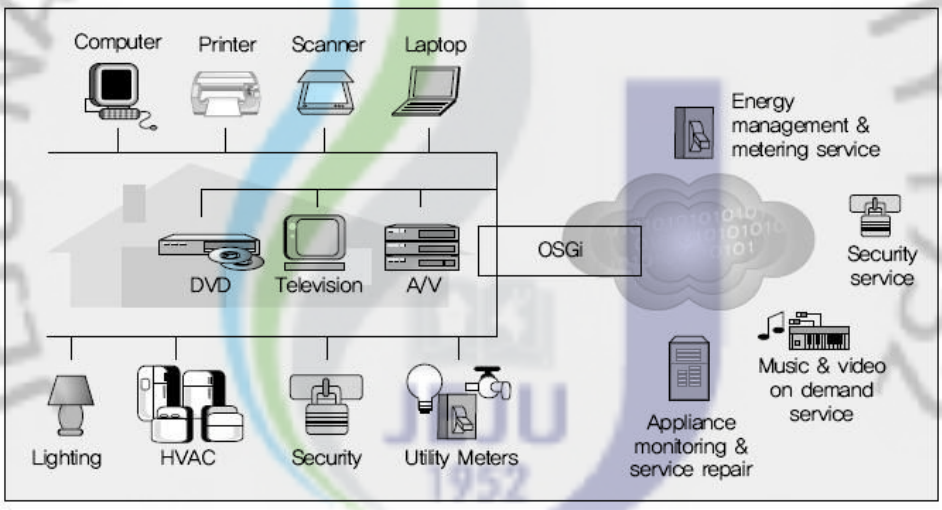

그림 1. OSGi와 홈 솔루션 서비스 개념도

OSGi는 번들(Bundle)이라는 모듈 단위를 기반으로 동작한다. 번들은 실제로 기존의 자바 플랫폼에서 사용하는 JAR 파일과 같은 형식이지만 JAR 파일들이 가지는 몇 가지 제약사항을 해결한 것이다. 즉, OSGi는 한 개의 번들 또는 여러 개의 번들로 이루어진 애플리케이션 자체를 언제든지 동적으로 프레임워크 상에 설치, 실행, 업데이트, 중단, 제거하는 것을 가능하게 하는 매우 유연한 유니버설 미들웨어 프레임워크이다.

OSGi Alliance 창단 이후 2005년 5월에 OSGi Release 1.0(R1) 스펙이 만들어졌고, 이후 계속적인 확장으로 R2와 R3를 거쳐 현재는 R4.2 Early Draft 버전 스펙까지 나와 있는 상태이다. 초기 R1에서는 기본적인 프레임워크 정의 및 동적으로 디바이스에 대한 지원을 확장할 수 있는 발판을 마련한 것으로 기초적인 정보기기의 연동과 상태 모니터링 기능 탑재, 그리고 표준화에 주력했다. R2 에서는 애플리케이션의 설정을 저장하기 위한 Configuration, Preference 같은 서비스와 Permission Admin 같은 기초적인 보안 서비스와 운영과 관리 기능 등을 추가 했다. R3 에 오면서 OSGi는 큰 확장을 하게 되어 UPnP와 Jini 표준이 서비스로 추가되었고, Star Level과 Execution Environment의 지원, 외부의 오픈소스를 이용해 처리하던 XML 파싱, Wire Admin, URL Handler등의 기능이 표준 서비스로 채택되어 명실상부하게 모바일, 임베디드, 데스크톱 애플리케이션, 클라이언트/서버 환경을 모두 아우르는 미들웨어 프레임워크로 자리 잡았다. R4에 이르러서는 좀 더 세분화 되면서 카테고리별로 디아비스 특성에 맞는 콘텐츠와 시스템 서비스들이 발전하게 된다. 특히 휴대폰과 스마트 홈 컨트롤러, 빌딩 자동 컨트롤러, 자동차용 내비게이션과 텔레매틱스 솔루션 등의 폭발적인 성장에 힘입어 모바일, 임베디드 시스템에 대한 많은 요구 사항들이 직접 기본 서비스에 선택되어 탑재되었다. R4는 프레임워크 레이어(Layered) 구조로 바뀌어 좀 더 이해하기 쉽고, 잘 분리된 구조로 바뀌었으며 프레임워크 전체에서 사용 가능한 이벤트(Event) 시스템이 포함되어 컴포넌트 간의 통신을 더욱 편리하면서도 강력하게 지원해주게 된다. R4 초기에는 모바일 쪽 서비스들이 따로 분리되어 있었으나, R4.1에서는 기본 플랫폼 서비스로 포함되었다. 현재 제정중인 R4.2에서는 Blueprint라는 서비스에 의해 Component Model이라는 개념이 소개되어 SpringDM에서 지원해주고 있는 개념을 OSGi의 기본 기능으로 추가 할 예정이다. 또한 Distributed OSGi를 지원하게 되어, 원격에 있는 OSGi 프레임워크 간의 Service 공유 및 통신이 가능해 질 것으로 보인다[5][6][7]. 그림 2는 OSGi 스펙 버전별 서비스 내용을 보여준다.

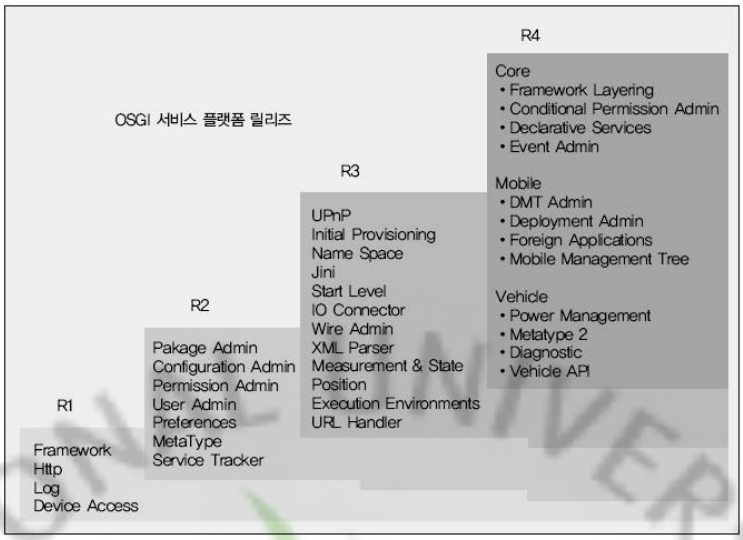

그림 2. OSGi 버전별 서비스 내용

2) OSGi 특징

(1) Service Component Management

OSGi는 서비스가 작동하고 운영되는 서비스 환경에 관한 표준 기술이다. 과거 OSGi R1.0 버전이 발표될 때만 하더라도 OSGi는 홈 네트워크의 네트워크 표준에 국한된 부분이 많았으나, 지금은 홈 네트워크와 모바일, 임베디드, 텔레매틱스, PC 애플리케이션(이클립스 기반의 RCP)은 물론이고, 엔터프라이즈 환경의 프레임워크에까지 확장되고 있다. 그만큼 기존의 미들웨어에 비해 많은 장점을 가지고 있는 셈이다. 또한, 과거 초기 단계의 홈 네트워킹 및 빌딩제어, 자동화 시장에는 디바이스 제어 기술이 주로 쓰였지만 현재는 그보다 훨씬 다양한 디바이스간의 상호작용을 요구하고 있고, 나아가 콘텐츠 및 솔루션 서비스가 점차 고도됨에 따라 OSGi와 같은 서비스 플랫폼 환경이 꼭 필요하게 될 것이다. 현재 다양한 장치들 간의 상호운용성을 위해 UPnP, HAVi, Jini 같은 표준이 만들어져 주로 장비의 제어나 데이터 전달 등을 처리하므로, 이는 OSGi 기술과 상호 보완적인 위치에 있다고 할 수 있다. OSGi를 사용하거나 구현하는 쪽에서는 이러한 미들웨어 지원이 또 다른 고려 사항임을 유의해야 한다[5].

위에서 말한 것처럼 OSGi는 Dynamic Module System for Java'이다. 다시 말해 바이트코드(ByteCode)와 가상 머신(Virtual Machine) 기수를 이용하여 코드 호환성을 보장하는 자바 플랫폼 위에서, 각 애플리케이션이 번들이라 불리는 작고 재사용 가능한 컴포넌트로부터 조립될 수 있도록 도와준다. 번들은 OSGi에서 이야기하는 각각의 컴포넌트 또는 애플리케이션을 가리키는 단위를 의미 한다[6]. 여러 개의 컴포넌트(Bundle)로부터 조합된 애플리케이션들은 OSGi 프레임워크가 설치된 곳은 어디든지 배포될 수 있다. 이러한 애플리케이션들은 시스템의 재시작 없이 컴포넌트의 연결구조등을 동적으로 변경할 수 있으며 이를 위해 OSGi는 서비스 지향 아키텍처(Service Oriented Architecture, SOA)를 사용한다[5][6]. 즉, 서비스(애플리케이션)는 모두 번들의 물리적 묶음에 포함되며, 복수개의 OSGi 서비스가 하나의 번들에 포함될 수도 있으며, 번들은 배포와 관리의 기본 단위를 형성한다. 이 번들들을 관리하는 것이 바로 프레임워크(Framework)이다[5][7].

프레임워크는 서비스에 대한 등록/관리기(Service Registry)를 가지고 있어 서비스에 대한 등록, 조회, 실행, 삭제 등을 수행한다. 즉, 각각의 컴포넌트/애플리케이션들은 OSGi 프레임워크에서 제공하는 OSGi에서 제공하는 Service Registry에 자신의 서비스를 등록하여 OSGi를 통해 서비스를 익스포트(Export)할 수 있고, 이 서비스를 사용하고자 하는 컴포넌트들은 서비스 레지스트리로부터 손쉽게 임포트(Import)하여 연결 및 사용이 가능하다.

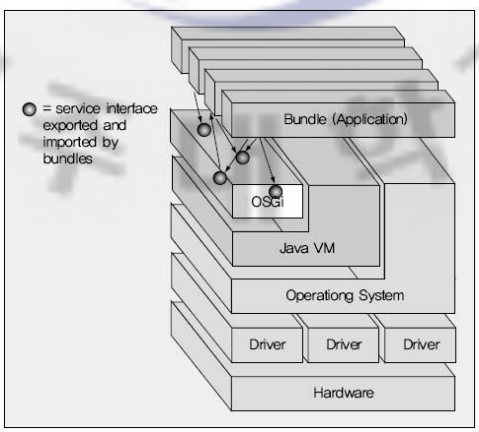

그림 3. OSGi 프레임워크와 번들

그림 3에서 볼 수 있듯 OSGi는 자바 가상머신에서 돌아가는 하나의 프레임워크이며 사용자가 개발한 응용 서비스는 번들 형태로 선언되어 OSGi 내부에서 실행된다. 동적으로 번들이 추가/삭제되고 서로간의 호출이 일어날 수 있기 때문에, OSGi는 동적으로 애플리케이션 추가/변경이 필요한 오픈 플랫폼에서 많이 사용되고 있다.

(2) Remote Component Management

OSGi는 원격관리를 지원한다. OSGi는 번들 단위로 서비스를 형성하고 운영하는데, 번들을 업데이트하거나 원격에서 업데이트를 관리 및 제어할 수 있다. 자바 VM(Virtual machine)을 재부팅 할 필요 없이 사용 중에 원격으로 업데이트 할 수 있는 것은 매우 큰 특징이다. 이를 위해 OSGi는 원격관리 표준 프로토콜을 제정했으므로, OSGi가 탑재되고 운영되는 환경에 맞춰 이용하면 된다.

원격관리 프로토콜로는 대표적으로 OMA-DM, SNMP, CMSE 그리고 Telnet/SSH 등이 사용된다. OSGi가 탑재된 컨트롤러나 디바이스가 인터넷이나 로컬 네트워크에서 운영된다면 주로 과거에는 SNMP, Telnet/SSH 등을 사용했다. 그러나 최근 무선 환경이 급격히 확산되고 모바일/무선 디바이스의 사용이 늘어나면서 새롭게 탄생한 프로토콜이 바로 OMA-DM(Open Mobile Alliance for Device Management)이다. 다시 말해 OMA-DM은 모바일 디바이스를 위한 유무선 통합 관리 프로토콜이다. 물리적인 레이어(Physical Layer)에서 유무선 포트라면 USB, RS-232C를 사용하고, 무선모드에서 GSM, CDMA, IrDA, 블루투스를 사용해 데이터를 전송한다. 또한 전송 레이어(Trans port Layer)에서는 HTTP, WAP, OBEX(Object Exchange)를 사용한다. 데이터 전송 언어로는 SyncML을 상용해 데이터 호환 및 규격을 통일했다. 마지막으로 Binary Transmission과 Session Management를 통해 SyncML의 텍스트 포맷(Text format)을 압축/암호와 하고 HTTP/WAP의 세션을 관리한다. 필립스의 가정용 컨트롤러인 iPronto, 노키아의 스마트폰, BMW 5/6 시리즈에 탑재된 텔레매틱스 등이 OMA-DM을 활용해 콘텐츠 번들을 관리하고 업그레이드 하는 대표적인 제품들이다.

(3) Application collaboration

수많은 자바 애플리케이션 서버 환경에서 구동하는 자바 애플리케이션들은 독립성을 보장하기 휘해 극히 폐쇄적인 컨테이너 환경에서 작동된다. 따라서 다른 자바 애플리케이션과의 연동이나 통합이 이뤄지려면 라이브러리 코드를 각각 가져와서 구동해야 하므로 오버헤드가 필연적으로 발생한다. 예를 들어 엔터프라이즈 환경에서 사용되는 JMS 서비스 API는 백 앤드 서버 라이브러리에 위치하며 그 크기가 대략 30~40KB 정도다. 만약 MIDP 애플리케이션이 JMS를 이용해 메시징 서비스를 하려한다면, 백 앤드 서버의 라이브러리 코드를 사용해야한다. 이때 다수의 MIDP 애플리케이션이 구동된다면 n개의 JMS-API가 동작하는 오버헤드가 생긴다. 이런 문제점들을 해결하기위해 나온 솔루션 서비스가 바로 SOA이다. OSGi는 이러한 SOA의 기본적인 구조 즉, 공통으로 사용되기 위한 서비스 또는 라이브러리 API를 서버나 공간의 어느 디바이스에 등록해(Registry) 배포 및 공유하면(Contribute) 접근 가능한 어떤 애플리케이션이라도 사용할 수 있는 연결 구조(Loosely Coupled)를 지향한다. 특히 OSGi는 엔터프라이즈 서버 환경보다 매우 가볍고, 빠르게 접근해 바인딩 할 수 있는 OSGi Framework Service Registry 구조를 가지고 있다. 이러한 OSGi 서비스간의 협업 구조는 리소스가 한정적인 모바일이나 임베디드 환경에서 매우 강력한 위력을 발휘했다. 현재는 데스크톱 애플리케이션에까지 그 영역이 확장되어 이클립스 기반의 RCP와 IBM Expeditor 솔루션으로 나타나고 있다[5].

3) OSGi 아키텍처

(1) OSGi 프레임워크

OSGi 프레임워크는 R1부터 정의되고 다듬어져 왔다. OSGi 구조는 그림4와 같이 몇 개의 계층으로 구성되어 있다[8].

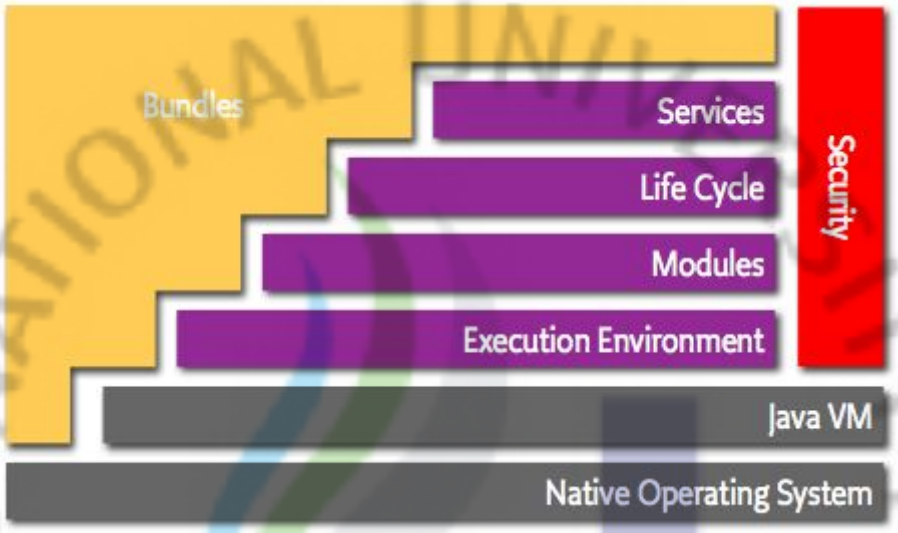

그림 4. OSGi 프레임워크 아키텍처

OSGi는 우선 자바 런타임(Runtime) 환경에서 구동된다. 자바 런타임은 하드웨어 환경에 의해 J2ME(CDC/CLDC), J2SE, J2EE로 구성되며, 하나의 JVM에서 서로 다른 복수 개의 클래스 로더들에 의해 각각의 OSGi 애플리케이션을 실행한다. 자바 런타임 환경 위에서는 OSGi 프레임워크가 실행된다. OSGi 프레임워크는 크게 번들의 실행주기(설치, 시작, 중단, 제거, 업데이트)와 OSGi 기본 실행 단위인 번들(Bundle)과 서비스에 대한 운영 관리, 이어서 리소스와 서비스 레지스트리 등을 담당하게 된다. 복수 개의 클래스 로더에 의해 각기 다른 OSGi 애플리케이션이 독립성을 가지고 실행되지만, OSGi Framework Service Registry에 등록된 Sharing Code와 Address에 의해 서로 다른 번들간의 리소스 공유와 연동/통합으로 무수히 많은 서비스들을 생성하고 실행 할 것이다. 이러한 OSGi 프레임워크 구조는 JES(Java Embedded Server)에 의해 영향을 받았다.

표 1. 프레임워크의 주요 레이어

Layer	주요 특징 및 역할
Bundles	왼쪽 위에 있는 번들은 아래에 설명될 레이어와 같은 개념이 아닌, OSGi의 레이어를 통하여 작성되고 프레임워크에 올려진 실제 번들들을 의미하는 것으로, 개발자가 개발해서 프레임워크에 올리는 번들을 말한다.
Security Layer	자바의 보안(Security) 구조에 기반하고 있으며, 패키지나 서비스에 대한 권한(Permission)을 관리하거나, Digital Signed JAR 파일에 대한 지원을 해주는 레이어 이다.
Service Layer	모듈 레이러를 통해 작은 컴포넌트 단위인 번들을 만들어 재사용할 수 있게 되고, 라이프 사이클을 통해 번들을 관리한다. 이들 레이어의 상위인 서비스 레이어는 이런 번들들이 서로 동적으로 협동하여 작업 할 수 있도록 하는 방법을 제공한다. 즉, 서비스 레지스트리(Service Registry)를 통해 서비스를 등록하고 찾을 수 있도록 지원하는 레이어 이다.
Life Cycle Layer	모듈 레이어 상단에 위치하는 라이프 사이클 레이어는 번들이 어떻게 동적으로 설치되고 관리될 수 있는지를 정의하는 레이어로 번들 내부에서 어떻게 외부 OSGi Context에 접근할 수 있는지를 정의한다.
Module Layer	OSGi의 근간이 되는 번들을 정의하는 레이어 이다.
Execution Environment	번들이 실행될 수 있는 환경을 말하는 것으로, J2ME, J2SE, 와 같은 환경을 말한다.

(2) Bundle

OSGi의 가장 기본적인 실행 단위인 번들은 OSGi 프레임워크에서 수행되는 어떤 S/W 컴포넌트의 resources, 동작을 위한 Java Classes, 번들 정보를 담고 있는 Manifest file, service를 포함하는 JAR 파일 등이다. OSGi는 단 하나의 JVM 인스턴스 위에서 동작하고, 복수 개의 클래스 로더를 수행해 독립된 Namespace를 가진다. 또한, 번들은 동적 라이프 사이클(Dynamic Install, Start, Stop, Delete, Update)을 갖고 수행한다. 자바 애플릿처럼 서버에서 다운로드 하는 것이 아니라 로컬 디바이스에서 상주하는 방식인 것도 번들이 가지는 특징이다. 그림 5는 이러한 번들의 동적 라이프 사이클을 나타낸다.

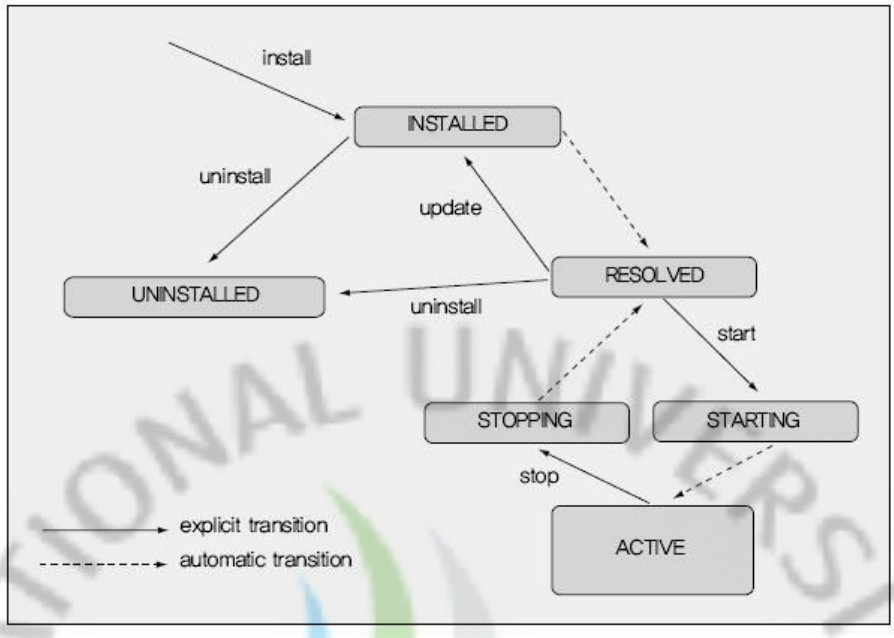

그림 5. OSGi 번들 다이나믹 라이프 사이클[5]

(3) OSGi Service

OSGi Service는 프레임워크에서 공통/반복적으로 사용될 수 있는 서비스들을 정의해서, 각 번들이 서비스를 쉽게 사용할 수 있도록 해준다. 이 서비스들은 마치 자바 언어가 버전이 올라갈 때마다 기능이 추가되는 것처럼, OSGi 버전이 올라 갈 때마다 새로운 서비스가 추가되고 있다. OSGi 서비스는 자바 오브젝트로서 하나의 번들에 의해 등록되며, 다른 번들에 의해 사용되거나 복수 개의 번들이 연동 및 통합해 독자적인 서비스를 만들기도 한다. OSGi의 각 릴리즈들 간에 추가된 서비스들 가운데 몇 개의 서비스는 OSGi 프레임워크 내부에서 사용자가 코드 상으로 직접 호출해서 쓸 수 있는 것도 있고, 몇 가지는 프레임워크 자체를 구성하는 기초 서비스로 작동하기도 한다. 아래 그림 6은 OSGi 번들과 서비스의 생성을 나타낸다.

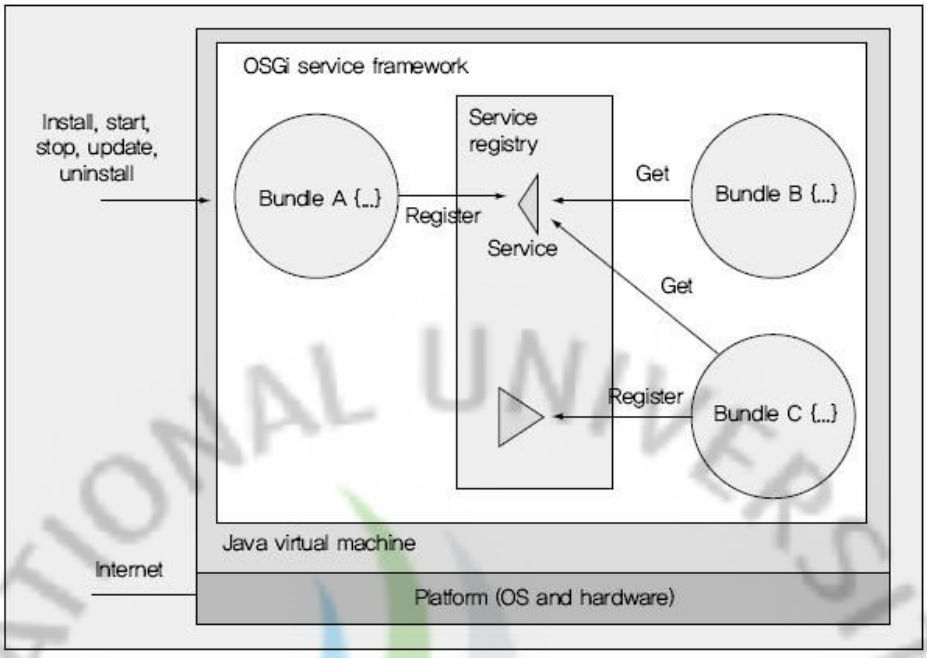

그림 6. OSGi 번들과 서비스 생성 과정[5]

앞서, OSGi는 동적으로 연결구조가 변경될 수 있도록 서비스 기반 구조(SOA)를 지향 한다고 하였다. 서비스의 제공자가 서비스중개자에 자신의 서비스를 등록하고, 서비스 사용자가 이를 찾아서 서비스 계약(인터페이스)을 통해 서비스를 사용하는 형태이다. OSGi는 이 형태의 모델을 따르고 있다. 위 그림에서 번들 A는 자신의 서비스를 서비스 레지스트리(Service Registry)에 등록한다. 이것은 보통 비즈니스 로직(Business Logic)이라고 부르며, 정의된 인터페이스를 가지는 독립적인 서비스를 의미한다. 이 인터페이스를 통하여 새롭게 구현된 번들 B, 번들 C는 다시 하나의 서비스로 등록이 가능하다. 클라이언트(Client)에서는 번들 A에만 연결하면 다양한 서비스(번들 A, 번들 B 서비스)를 제공 받을 수 있다. 이는 SOA의 특징인 서비스와 이를 조합하여 하나의 애플리케이션을 구축하는 것이다. 기존의 재사용 가능한 소프트웨어나 컴포넌트 모델 방법론과 같이 소프트웨어를 각각의 개별단위로 만들어 조합하거나 재사용 한다는 점에서는 같으나, 네트워크 통한 원격협업이 가능해지고, 인터넷의 발달로 HTTP와 XML이 표준적으로 사용되면서 웹을 이용한 쉬운 연동방법들이 많이 등장하게 되어 웹 서비스 기반의 SOA가 사용되고 있다.

2. 실시간 센서 데이터 스트림 처리 미들웨어

최근에는 RFID/USN 기술을 이용하여 물류, 소매업, 의료, 공장·가정·사무실 자동화, 보안, 재난 방지, 재산 관리 등 다양한 서비스를 제공하기 위한 응용 및 기술들이 연구 개발되고 있다. RFID/USN 기술을 이용한 유비쿼터스 응용 서비스를 쉽게 구축할 수 있으려면 RFID/USN 센서가 부착된 객체와 응용 서비스 사이에서 교량 역할을 하는 미들웨어가 필요하다.

RFID/USN 미들웨어는 이기종 운영체제 간 상호협력이 가능하며, 분산처리의 신뢰성, 네트워크의 독립성, 응용 프로그램 및 서비스간의 상호운용성 및 투명성을 지원한다. 여러 가지 센서를 관리하고 센서의 프로토콜을 이용하여 데이터를 수집하며, 또한 수집되어 가공되지 않은 데이터로부터 의미 있는 정보, 혹은 응용이 사용하기 쉬운 형태의 정보를 추출하여 응용 서비스에 전달하는 기능을 수행한다. RFID와 USN은 기술적 유사성 때문에 RFID도 하나의 센서로 취급되어 RFID/USN이 동시에 처리가 가능한 실시간 센서 데이터 스트림 처리 미들웨어에 대해 연구가 진행되고 있다[12].

1) RFID/USN 미들웨어

(1) RFID 미들웨어

일반적으로 RFID 시스템은 기본적으로 사물에 부착되고 이를 유일하게 구분할 수 있는 정보를 담고 있는 RFID 태그, 태그에 담겨진 정보를 인식하는 RFID 리더기, 마지막으로 리더기로부터 인식된 태그정보를 처리하는 호스트 시스템으로 구성된다[11].

호스트 시스템이란 RFID 리더기로부터 인식된 태그인식정보를 처리하고 응용 프로그램이 이를 활용하는 전반적인 소프트웨어 시스템을 지칭하며, 이중에서 미들웨어는 리더기로부터 인식된 RFID 태그정보를 수집하고 중복된 정보들을 제거하여 의미 있는 정보만을 응용 프로그램에 전달한다. 이러한 RFID 미들웨어는 단순한 태그 인식정보의 전달뿐만 아니라, 서로 다른 형태의 통신 방식 및 프로

토콜을 지원하는 리더기를 일관된 형식으로 통합적으로 관리하고, 모니터링 할 수 있어야 한다[18].

RFID 미들웨어는 리더에서 계속적으로 발생하는 식별코드 데이터를 수집, 제어, 관리하는 기능을 하며, 모든 구성요소와 연결되어 계층적으로 조직화되고 분산된 구조의 미들웨어 네트워크를 구성하여 서로 통신한다. 미들웨어는 다양한 형태의 리더 인터페이스, 다양한 코드 및 네트워크 연동, 여러 가지 응용 플랫폼에 대해서도 상호운용성을 보장할 수 있어야 한다[19]. 이를 위해 적어도 다음의 요건들이 만족되어야 한다.

가. 이기종 RFID 리더 시스템 지원 및 관리

RFID 미들웨어는 다수의 이기종 RFID 리더 시스템간의 이질성(예를 들면, 지원하는 태그와 리더간의 프로토콜의 상이함, 리더와 호스트 시스템간의 네트워크 인터페이스의 다양성 등)이 존재하는 환경 하에서, RFID 하드웨어 시스템을 상위계층에서 일관되게 접근이 가능하도록 기능을 제공해야 한다. 리더인식영역에 존재하는 태그정보 수집, 리더기 설정 및 원격제어, 리더 시스템 모니터링 등의 관리가 이뤄질 수 있도록 기능이 제공되어야 한다.

나. RFID 태그 데이터 처리

RFID 태그 데이터가 RFID 리더기로부터 반복적으로 대량의 정보가 유입됨에 따라서, RFID 미들웨어는 이러한 중복된 정보 및 응용 시스템 계층에 불필요한 정보들을 필터링하고 요약하는 기능을 제공해야 한다.

다. 응용 시스템과의 연동

RFID 미들웨어는 정제, 요약된 태그데이터를 데이터 수요자인 기존 응용 시스템에 신뢰성 있게 전송할 수 있는 기능을 제공해야 한다. 또한, 이상의 정제, 요약 과정을 최종 인식된 RFID 태그 정보를 이용하여 태그가 부착된 개별사물에 대한 상세정보를 제공하기 위한 시스템 또한 필수적인 요소라 할 수 있으며, RFID 미들웨어 시스템의 고려 대상이 된다.

RFID 미들웨어는 RFID 정보가 다양한 응용 서비스에서 사용되도록 RFID 태그에 저장되어 있는 데이터를 적절한 장소와 적절한 시간에 응용 서비스로 전달하는 기능을 제공한다. 이때 RFID 리더기로부터 수집된 정보 중 응용 서비스가 필요한 데이터의 형식 및 전달 조건에 따라 요구되는 기능이 달라진다. 이것은 다양한 조건에 따른 필터링 기법, 처리 대상 데이터의 양과 동시에 처리되어야 하는 조건의 수 등을 고려하여 실시간 센서 데이터 처리가 손실 없이 이루어져야 한다. 또한 응용 서비스로 정보를 제공하는 다양한 인터페이스 기능이 가능해야 한다.

(2) USN 미들웨어

USN 기술은 모든 사물에 컴퓨팅 능력과 통신 기능을 부여해 환경과 상황을 자동으로 인지하도록 함으로써 사용자에게 최적의 서비스를 제공할 수 있게 하는 기술이다[20]. 즉 일상생활에서 사용되는 가전기기, 사무용품등에 컴퓨팅 능력이 있는 센서들을 장착해 이 센서들의 주변의 온도, 습도 등의 환경정보와 사용자의 정보를 인식해 사용자가 필요한 서비스를 제공하도록 한다.

일반적으로 USN은 센서노드, 싱크노드, 미들웨어로 구성되어 있다. 센서 노드는 센서를 통해 주변 환경의 정보를 수집하여 가공·데이터화해 싱크노드에 전송한다. 즉, 센서노드는 서비스의 요구에 따라, 또는 이미 설정한 조건의 이벤트 발생에 따라 수집된 정보를 싱크노드에 전달한다. 센서노드로부터 데이터를 수집한 싱크노드는 수집한 데이터를 미들웨어를 통해 원격지의 위치한 서버나 사용자에게 전달하여 데이터를 활용할 수 있게 한다[19].

미들웨어는 다양한 하드웨어 환경을 하나의 플랫폼을 보이게 하고, 이를 이용한 단일 플랫폼에서 다양한 애플리케이션을 구동할 수 있는 환경을 지원하는 소프트웨어를 말하며, USN 미들웨어는 물리적으로 USN 응용 서비스 시스템과 센서노드 하드웨어의 중간에 위치하여 그 둘 간의 통합이 유연하게 이루어지도록 하는 역할을 수행한다.

USN 미들웨어의 기본 기능은 센서노드들로부터 수집되는 수집 정보를 효율적으로 관리하고, USN 응용 서비스들로부터 주어지는 질의들에 대하여 신속히 응답하는 것으로 매우 단순한 것처럼 보인다. 그러나 USN 미들웨어는 수집 정보

를 관리할 때, 다수의 센서네트워크에 설치된 수많은 센서노드들로부터 연속하여 주어지는 대용량 센싱 정보를 효율적으로 관리하기 위한 방법과 연속적인 센싱 정보 요청을 센서네트워크 및 센서노드의 컴퓨팅, 통신 및 저장 능력 등을 고려하여 처리할 수 있는 방법도 제공하여야 한다. 또한, 주어지는 질의들에 대하여 실시간으로 응답할 수 있어야 하고, 유비쿼터스 환경에서 발생 가능한 일시성, 연속성, 이벤트 질의와 같이 다양한 형태의 질의도 처리할 수 있는 능력을 보유해야 한다. 이외에도 USN 미들웨어는 센서네트워크 구성에 대한 추상화 기능을 지원함으로써 이기종의 센서네트워크 구성에 독립적으로 응용 서비스가 가능하도록 해야 한다.

대부분의 USN 미들웨어는 응용 서비스를 지원하기 위하여 서버 시스템에 설치되고, 노드들의 원활한 동작과 성능 향상을 위하여 센서노드와 싱크노드에도 설치된다. 이러한 USN 미들웨어를 설치되는 위치에 따라서 구분하며, 서버 시스템에 설치되는 경우는 server-side 미들웨어, 그리고 노드에 설치되는 경우는 in-network 미들웨어 또는 센서노드 미들웨어라고 할 수 있다[20]. 일반적으로 server-side 미들웨어는 기본 기능으로 이기종 센서 네트워크로부터 수집한 센싱 데이터를 필터링, 통합, 분석해 의미 있는 상황정보를 추출, 저장, 관리, 검색하고 그 정보를 응용 서비스로 전달해, USN 서비스 간 연계, 통합을 용이하게 한다. 이에 반하여, in-network 미들웨어는 애플리케이션과 환경 변화에 따른 센서 노드의 재 프로그래밍, 센서 네트워크의 변화 지원, 센싱 데이터의 처리, 저장, 질의 처리, 이벤트 처리 기능 등을 제공하며, 주로 센서노드와 싱크노드 영역으로 국한된다.

2) EPCglobal의 ALE 미들웨어

EPCglobal에서 표준으로 제안한 RFID 미들웨어의 스펙인 ALE는 EPC 미들웨어에 대한 구체적인 구현 및 특정 소프트웨어 모듈 내에서의 내부 인터페이스를 기술하지 않고 외부 인터페이스만 정의하여 데이터를 수집하고 필터링 및 그룹핑하여 비즈니스 로직을 해석하는 이벤트를 생성하는 데에 초점을 둔다. 즉, 원

시 EPC 데이터를 획득하는 하부 구조 모듈과 그 데이터를 필터링 및 카운팅하는 구조적 모듈, 그리고 데이터를 사용하는 클라이언트 응용 간의 독립성을 제공한다[21].

RFID/USN 미들웨어에 대해 RFID 미들웨어는 사실상의 국제 표준인 ALE 스펙이 존재하여 RFID 미들웨어의 연구가 상당히 진행되고 있는 반면, USN 미들웨어는 그 복잡성 때문에 국제 표준이 명확하게 정해져 있지 않다 ALE는 리더기를 통해 수집되는 RFID 코드 데이터를 PML(Physical Mark-up Language) 문서로 바꿔서 처리를 한다. PML 문서로 바꾸는 과정 중에 원시 EPC 데이터를 URN 코드로 변환해야 한다. 즉, ALE 미들웨어는 리더기를 통해 수집되는 원시적인 RFID 코드 데이터를 필터링 및 그룹핑을 하는데 URN 코드로 바꿔서 처리하고 있다. 따라서 다양한 RFID 코드 데이터가 입력될 경우 URN 코드로 변환해야 한다[22]. 따라서, 다양한 실시간 센서들을 ALE 기반 RFID 미들웨어에서 처리가 가능하도록 센서 데이터를 EPC 포맷에 데이터를 변환한 후 URN 코드 체제로 바꾸어 ALE 기반 RFID 미들웨어에서도 USN 실시간 센서 데이터의 처리가 가능하도록 한 국제 표준을 따르는 실시간 센서 데이터 스트림 처리 미들웨어에 대한 연구가 진행되었다[12]. 이 실시간 센서 데이터 스트림 처리 미들웨어는 RFID 태그 정보를 통하여 사물의 인식 정보만을 처리하는 것이 아니라 RFID 이외의 센서 데이터를 처리할 수 있는 미들웨어를 개발할 수 있으므로 사물의 인식 정보와 센서 데이터를 활용하여 다양한 응용 서비스를 저비용으로 제공할 수 있다. 즉, 응용과 다양한 센서들의 표준화된 인터페이스를 미들웨어가 제공함으로써 이기종 센서를 저렴한 비용으로 쉽게 연결할 수 있는 범용성을 가질 수 있다.

3) 실시간 센서 데이터 스트림 처리 미들웨어

실시간 센서 데이터 스트림 처리 미들웨어는 기본적으로 ALE 기반의 RFID 미들웨어로 구성되었으며 여기에 온도, 습도, 조도 등의 센서 데이터의 처리가 가능하다. 실시간 센서 데이터 스트림 처리 미들웨어는 OSGi 응용의 요청에 따라 RFID/USN 실시간 센서 데이터 스트림을 처리하여 응용으로 전송한다. 아래 그림 7은 실시간 데이터 스트림 처리 미들웨어의 간단한 구성도를 보여준다.

그림 7. 실시간 데이터 스트림 처리 미들웨어 구성도

(1) 센서 연결 관리

여러 종류의 RFID Tag 및 실시간 센서 장치에 관련된 실시간 센서의 정보를 관리하여 실시간 센서의 환경에 따라 실시간 데이터를 전송받을 수 있다. 조도, 온도, 습도 센서 및 RFID 리더 장치, 그 외에 다양한 센서 장치 정보를 관리자 툴 및 미들웨어가 제공하는 인터페이스를 이용하여 센서 정보 및 센서 IP 주소와 포트 종류를 입력받으면, 센서 연결 관리 모듈은 하나의 실시간 센서 인터페이스를 스레드로 바꿔 메모리에 올려서 지속적으로 관리한다. 효율적인 센서 연결 관리를 위하여 물리적인 센서와 이를 그룹화한 논리 센서로 나누어서 관리하

며, 네트워크 장애 및 센서 장치의 오류로 인하여 실시간 센서 데이터를 입력 받지 못하면 이를 관리자에게 알리거나 Exception을 일으켜 로그에 기록한다.

(2) 센서 인터페이스

미들웨어 내부에서 사용하는 정보를 받기 위한 인터페이스로 여러 종류의 RFID 리더 와 센서 장치에 관련된 환경정보를 입력 받기 위해 실시간 센서들에 대한 공통적인 정보를 인터페이스를 정의한 것이다. 미들웨어에 실시간 센서들이 확장될 때 센서 인터페이스를 제공함으로써 해당 정보가 입력되면 위의 센서 연결 관리 모듈이 센서를 연결한다. 실시간 센서들에 제공되는 대표적인 인터페이스로는 실시간 센서의 고유번호, 회사, 이름 등을 입력 받기 위한 DeviceInfo Interface와 통신 방법 및 포트번호 데이터의 형식 등을 입력 받기 위한 DeviceTransInfo Interface가 있다.

(3) 센서 데이터 정규화 및 코드변환

다양하게 들어오는 각종 실시간 센서 데이터들은 서로 다른 데이터 형태를 가지고 있어 미들웨어에서 사용할 수 있는 표준 데이터 타입으로 정규화를 시켜준다. 센서 데이터 변환을 위해 센서 데이터 변환부에는 각각의 센서 데이터 변환 정보인 메타정보를 관리하는 센서 정보 관리자(Sensor Information Manager)와 센서 데이터를 EPC 코드와 유사한 형태로 변환하는 EPC 코드 생성기(EPC code Generator), EPC 코드 생성기에 의해 EPC 코드와 유사한 형태로 변환된 데이터를 URN 코드로 변환하는 URN 코드 생성기(URN code Generator) 그리고, 마지막으로 미들웨어 내에서 처리되어 지는 데이터인 PML로 변환하는 PML 코드 생성기(PML Generator)로 구성된다[15].

센서 데이터의 처리 과정은 센서 정보 관리자에 등록된 센서 변환 정보에 따라 EPC 코드 생성기에 의해 EPC 코드 형태의 센서 데이터로 변환하고, EPC 코드 형태로 변환된 데이터를 URN 코드 생성기에 의해 URN 형태로 변환한 후, 최종적으로 PML 코드 생성기에 의해 ALE 미들웨어에서 처리되는 데이터 형태인 PML로 변환한다[12].

(4) 요청 문서 분석

OSGi 응용 및 관리자, 개발자에 의해서 요청된 문서는 XML 형태로 전송된다. 요청 문서 분석을 통하여 ALE에서 SpecName이라 불리는 사용자가 요청한 요청문서의 이름, 물리적인 센서에 그룹화 되어 있는 논리적인 센서의 위치 및 정보센서 데이터가 언제부터 시작하여 얼마동안 센서 데이터를 수집할 것인지의 여부, 필터링 조건, 결과 값을 반환할 때 요청했던 문서의 첨부 여부, 받고자하는 데이터의 유형 등의 정보를 XML Parsing을 통해서 얻는다. 또한 수집된 데이터를 받고자 할 때 동기, 비동기 형식으로 받을 것인지도 정의할 수 있다.

(5) 기본 필터링

수집된 센서 데이터 중에서 노이즈 센서 데이터나 중복된 센서 데이터를 제거하는 역할을 한다. 센서들의 데이터는 리스너를 통하여 해당 큐에 저장되는데 이때 기본적으로 중복된 데이터 및 의미 없는 데이터를 삭제하는 역할을 한다.

(6) 요청 필터링

사용자의 요청 문서에 의해서 정의된 필터링 조건을 수행 하는 것으로 기본 필터링 모듈에서 기본적인 중복데이터와 노이즈 데이터를 제거한 뒤 사용자가 원하는 데이터만 추출 할 수 있도록 한다. ALE 내부의 URN 코드 형식에 의해서 필터링 된다.

(7) 결과 문서 생성

필터링된 센서 데이터의 결과 값을 데이터 지정 및 포맷 조건에 의해서 데이터를 변환 시키고 요청 문서 분석 모듈에 의한 각종 문서 정보를 결합하여 결과 문서를 생성하는 역할을 한다, 리포트를 생성할 때는 기본적으로 요청 문서 이름, 결과데이터의 논리 센서 정보, 그리고 요청 문서 첨부 여부, 마지막으로 전송할 센서 데이터가 필요하다. 결과 문서 생성 모듈은 이와 같은 입력 값을 받아서 XML 문서로 변환하여 전송한다.

이밖에도 실시간 센서 데이터 스트림 처리 미들웨어에는 원격으로 데이터를 전송하기 위한 원격 커넥션 관리, 다양한 DB의 연동, 스레드관리, 백업 등의 기능들이 존재 한다.

3. 기존 실시간 센서 데이터 OSGi 응용 서비스

현재 OSGi와 관련된 연구들을 살펴보면 '홈 네트워크 환경에서 상황 정보를 이용한 유아 위험 알림 시스템 설계', 건널목 사고예방을 위한 OSGi 기반 상황 모니터링 시스템', OSGi 기반 상황인지 모바일 헬스케어 시스템 설계 및 구현' 등의 관련 논문들에서는 이들 응용 서비스가 구체적으로 실시간 센서 데이터의 값을 센싱하고, 센싱된 데이터에 대한 처리 및 가공이 필요함에도 불구하고, 실시간 센서 데이터 스트림 처리에 대한 부분이 없거나 해당 연구에 맞게 정의해 온 상황(Context)에 따라 가공 처리된 실시간 센서 데이터의 획득이 가능하다는 가정 하에 연구를 진행하고 있다, 즉 실시간 센서 데이터 스트림이 어떤 미들웨어를 통하여 처리되고 전달되는지에 대한 부분이 명확하지 않다. 또한, "무선 센서 네트워크를 이용한 지능형 홈 네트워크 서비스 설계' 논문에 따르면 사용자 개인의 특성에 맞추어진 서비스를 제공하기 위해 사용자에 대한 정보와 환경데이터를 수집, 관리, 가공하는 미들웨어가 필요하다고 말하고 있다. 하지만 표 2와 같이 논문에서 제시한 홈 네트워크 서비스 기술에 대한 관련 연구 비교표를 보면 홈 네트워크 서비스의 표준인 OSGi에 대해서는 전혀 고려하지 않고 있으며 대부분 시스템의 경우 실시간 센서 데이터 스트림 처리 미들웨어를 자체 제작하여 사용하고 있다.

표 2. 홈 네트워크 서비스 기술에 대한 관련 연구 비교[14]

	순천향대	카톨릭대	원광대	순천대
상황정보	사용자 위치 사용자 식별 사용자의 기기 동작	조도 값	사용자 위치	사용자 상황, 물리적 환경 상황, 시공간 상황
위치인식 방법	기기 ON/OFF 및 마그네틱센서	없음	스위치 및 ON/OFF 센서의 동작	블루투스 통신 모듈
미들웨어	자체제작	자체제작	TMOSM	자체제작
응용서비스	개인화 서비스(기기 자동 제어), 방법	웹캠을 통한 댁내 모니터링, 전력관리	위치추적, 거주자의 헬스케어 정보	음악재생
제어	PLC, 자체제작보드	인터페이스 보드,(스텝모터)	없음	사운드 플레이어

이러한 자체제작 미들웨어 및 OSGi 표준을 따르지 않는 응용 서비스들은 향후 다양한 외부 비즈니스로직과 서비스 등의 추가와 다양한 디바이스들의 상호 운영이 필요한 경우 서로 다른 제약으로 인한 문제점 발생으로 인하여 비용 및 융통성 면에서 개선의 여지가 있다.

4. 고려사항

본 논문에서는 OSGi 기반 응용 서비스에서 효율적인 실시간 센서 데이터 스트림 처리를 위해 다음과 같은 사항을 고려하여야 한다.

RFID/USN 센서 처리가 가능한 ALE기반 실시간 센서 데이터 스트림 처리 미들웨어를 다양한 OSGi 기반 응용에서 사용이 가능하도록 해야 한다. 이에 따라 실시간 센서 데이터 스트림 처리 미들웨어에서 코어(Core) 모듈만을 사용할 필요가 있으며, OSGi 응용 서비스와 실시간 센서 데이터 스트림 처리 미들웨어와 연동이 가능한 OSGi 응용 프레임워크를 설계할 필요가 있다. 또한 OSGi 응용 프레임워크는 OSGi 기반 다양한 응용에서 사용이 가능하도록 번들 형태로 구현되어야 하며, ALE에서 요구하는 데이터 포맷, 교환 규약을 준수해야하고 이를 위해 OSGi Specification, ALE Specification 표준 등과 같은 참조 표현을 준용해야 한다.

III. OSGi를 위한 실시간 센서 데이터 스트림 처리

1. 개요

본 연구에서는 OSGi 프레임워크에서 실시간 센서 데이터 스트림 처리 방법을 제안한다. 기존의 OSGi 환경에서도 표준 실시간 센서 데이터 스트림을 효과적으로 처리하고 OSGi 응용 서비스에서 사용되는 다양한 RFID/USN 센서 및 홈 네트워크 기기들을 관리할 수 있도록 한다. 홈 네트워크 시스템 사용자에게 양질의 서비스를 제공하기위하여 OSGi 응용 프레임워크를 정의하여 OSGi 응용으로 하여금 실시간 센서 데이터 스트림을 효과적으로 처리할 수 있도록 하는 미들웨어 인터페이스를 번들 형태로 제공하고, OSGi 응용 서비스 개발자로 하여금 실시간 센서 데이터 스트림을 활용하여 손쉽게 효과적인 응용을 개발할 수 있는 API 제공 방법에 관한 연구이다.

이를 위해서는 OSGi 응용 프레임워크와 표준화된 방법으로 다양한 실시간 데이터를 처리 할 수 있는 미들웨어가 필요하다. OSGi 응용 프레임워크는 제공되는 표준 미들웨어와 효과적인 연동을 위하여 통신, 스펙정의 등의 인터페이스를 번들 형태로 제공해야한다. 미들웨어는 RFID/USN 등의 다양한 센서 장치들을 연결하고 제어할 수 있는 리더 인터페이스가 필요하며, 서로 상이한 데이터 포맷을 가지는 코드들을 하나의 데이터 포맷으로 변환하는 센서 데이터 변환 부분이 필요하다. 또한, 데이터의 가공을 위하여 센서 데이터 스트림을 위한 필터링 및 그룹핑 하는 부분과 사용자의 요청을 분석 할 수 있는 요청 스펙분석 부분, 미들웨어의 원격 지원을 위한 원격 서비스 전송 부분 등이 필요하다, 이에 따라서 본 논문에서는 ALE Specification Version 1.0을 기반으로 RFID 미들웨어에 다양한 실시간 센서 데이터 처리를 위한 기능과 모듈들을 확장한 실시간 센서 데이터 스트림 처리 미들웨어를 분석하여 연동 하였다.

2. 시스템 구성

OSGi 응용 서비스들을 위해 OSGi 응용 프레임워크를 정의하고 OSGi 응용 프레임워크는 실시간 센서데이터 스트림 미들웨어와 연동하여 데이터를 요청, 수집, 분석한다. OSGi 응용 프레임워크는 OSGi 응용 서비스와 실시간 스트림 처리 미들웨어 간의 인터페이스 역할을 한다. OSGi 응용 프레임워크를 이용한 OSGi 응용 서비스의 예로 실시간 센서 데이터를 바탕으로 간단한 홈 네트워크 기기들 제어하는 응용으로 구성하였다.

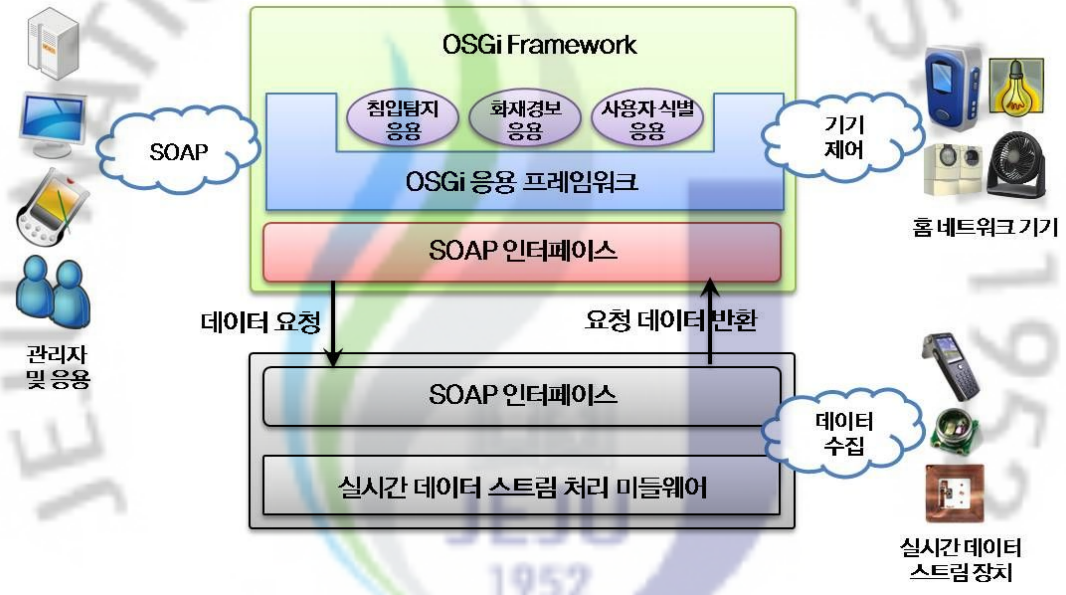

그림 8. 실시간 데이터 처리 개념도

그림 8은 OSGi를 위한 실시간 센서 데이터 스트림 처리를 위한 방법을 나타내는 개념도이다. 홈 네트워크 기기들을 제어하는 OSGi 응용 서비스들은 실시간 데이터 스트림 처리 미들웨어와 연동을 위하여 OSGi 응용 프레임워크 위에서 구성되고, OSGi 응용에서 실시간 데이터 스트림 처리 미들웨어로 데이터를 요청하면 수집된 데이터를 처리하여 OSGi 응용으로 전송한다. 전송된 데이터를 바탕으로 홈 네트워크 기기들을 제어한다. 시스템을 구성하는데 있어서 실시간 데이터 스트림 처리 미들웨어는 별도의 서버에 위치하여 OSGi 응용과 SOAP 프로토

콜을 이용하여 통신이 가능하도록 구성하였다.

OSGi 응용 서비스를 위한 효과적인 실시간 센서 데이터 스트림 처리를 위한 시스템의 관리자 및 OSGi 응용 부분은 OSGi 응용 프레임워크에서 제공하는 서비스를 통해 실시간 데이터 스트림 처리 미들웨어에 데이터를 요청하여 결과 값을 바탕으로 홈 네트워크 기기의 모니터링 및 제어를 한다. 또한 실시간 데이터 스트림 처리 미들웨어와 OSGi 응용이 서로 연동할 수 있는 환경을 제공하는 OSGi 응용 프레임워크와 표준화된 방법으로 데이터를 처리 가공하는 실시간 데이터 스트림 처리 미들웨어 가 있다.

3. OSGi 응용 프레임워크 설계

본 논문에서 제안하는 OSGi 응용 프레임워크를 이용하여 간단한 홈 네트워크 기기를 제어하는 홈 네트워크 응용 서비스이다. OSGi 응용에 포함된 OSGi 응용 프레임워크는 실시간 데이터 스트림 처리 미들웨어와 연동하고 홈 네트워크 기기 등의 장치를 컨트롤 하기위한 최소한의 모듈은 OSGi 기반 서비스 배포 단위인 번들 형태로 구성된다. OSGi 응용 프레임워크는 다시 표 3과 같이 크게 세 부분으로 구성된다.

표 3. OSGi 응용 프레임워크 구성

구분	주요 모듈
API 및 응용 서비스 인터페이스	· API 관리자 번들 · 상황인지 관리자 번들 · SOAP 인터페이스 번들
시스템 코어	· 데이터베이스 번들 · 자원 관리 번들 · 데이터 요청 처리 번들 · 데이터 결과 분석 번들
장치 연결 제어 인터페이스	· 장치 인터페이스 · 장치 연결 관리 번들 · 장치 제어 관리 번들

표 3. 과 같은 번들들을 기반으로 실시간 센서 데이터 스트림 처리 미들웨어와 연동 하고자 하는 OSGi 프레임워크 기반 응용 서비스의 구성도는 그림 9 와 같다.

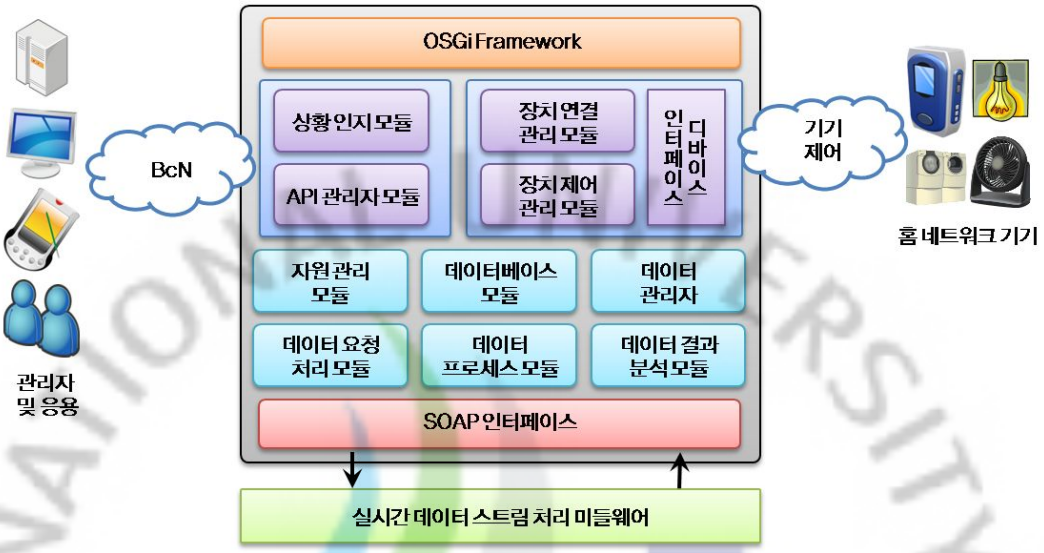

그림 9. OSGi 응용 프레임워크 개념도

데이터 관리자는 실시간 데이터를 요청, 분석, 처리 등 데이터를 관리한다. 실시간 데이터스트림 처리 미들웨어에 데이터를 요청하는 모듈, 요청한 데이터의 결과를 분석하는 모듈, 분석된 데이터를 관리하며, 데이터를 장치 제어 모듈에 전송하는 모듈로 구성된다.

관리자 및 응용에서 API를 이용하여 장치 제어 및 데이터가 요청되면 데이터 요청 처리 모듈에 의해 실시간 데이터 스트림 처리 미들웨어에게 해당 데이터를 요청한다. 데이터 요청처리 모듈은 응용의 요구 사항을 만족하는 스펙 문서를 작성하여 데이터를 요청하며, 그 결과 XML 요청 문서를 전송한다.

데이터 결과 분석 모듈은 스트림 처리 미들웨어로부터 전송된 데이터를 분석하여 사용자가 원하는 결과 값을 추출한다. 추출된 결과 값을 데이터 프로세스 모듈로 전송하면 결과 값을 바탕으로 홈 네트워크 기기 제어, 데이터베이스 저장 여부를 결정한다. 사용자 요청 사항이 미리 정의된 일련의 값들에 의해 홈 네트워크 기기의 제어가 필요할 경우 장치 제어 관리 모듈을 호출하여 장치를 제어하고, 그 결과를 장치 상태 관리자에게 알려 관리자가 결과를 알 수 있도록 한

다. 또한, 요청 결과나 홈 네트워크기기의 상태의 변화 등은 데이터베이스 모듈에 의해 데이터베이스에 저장된다.

장치 연결, 제어 관리 모듈은 장치와 센서들의 등록과 해제를 담당하며, 장치의 상태 제어와 관리자나 응용에게 장치 정보를 전달한다.

자원 관리 모듈은 제한된 임베디드 환경에서 자원 관리를 위한 모듈로써, OSGi 응용이 실시간 데이터 스트림 처리 미들웨어에게 데이터를 요청하거나 장치를 제어하고자 할 때 스레드를 관리하여 메모리 등 시스템의 자원을 관리한다. 데이터베이스 모듈은 다양한 유형의 데이터베이스 관리 시스템을 지원한다.

API 관리자 모듈은 다양한 API들을 제공하고 관리 한다. API를 제공하는 번들은 OSGi 응용 프레임워크에 존재하는 번들들의 실행/정지/업데이트/상태 등을 수행하기 위해 존재하며 또한, 응용 개발자나 시스템 관리자로 하여금 SOAP 인터페이스를 통하여 실시간 데이터스트림 처리 미들웨어를 접근하여 원하는 데이터를 획득할 수 있도록 한다. 응용 개발자 및 시스템 관리자에게 제공되는 효율적 센서 데이터스트림 처리 및 홈 네트워크 기기 제어를 위한 주요 API는 표 4와 같다.

표 4. 관리자 및 응용을 위한 주요 API

주요 API	기능
getDeviceNames():String	OSGi 프레임워크에서 현재 사용할 수 있는 장치 정보를 반환함.
getStateData():String	현재 제어 가능한 장치들의 상태 정보를 반환함.
setStateData(boolean state)	제어 가능한 장치를 제어함.
getDeviceBundleList()	프레임워크 내의 등록되어 있는 응용 번들의 정보를 반환함.
setBundle()	특정 번들의 상태를 설정함.
setRealData()	미들웨어에게 데이터 용청을 위한 스펙을 작성함.
requestRealData()	미들웨어에게 데이터를 요청함.

4. 주요 클래스 다이어그램

1) API 관리자 패키지

API 관리자는 응용 서비스 개발의 편의성을 제공하기위하여 웹 서비스 형태로 제공된다. 미들웨어의 호출을 위한 API들은 미들웨어 자체에서 WSDL 문서를 제공하고 있으며, 관리자는 SOAP 인터페이스를 통해 이들 API를 사용할 수 있다. 응용 사용자들이 API를 이용하여 시스템에 운영되는 각종 서비스에 대한 실행과 정보 이용이 가능하도록 하였다. 그림 10은 OSGi 응용 프레임워크에서 제공하는 API를 이용한 OSGi 응용 번들 중의 일부분을 보여준다.

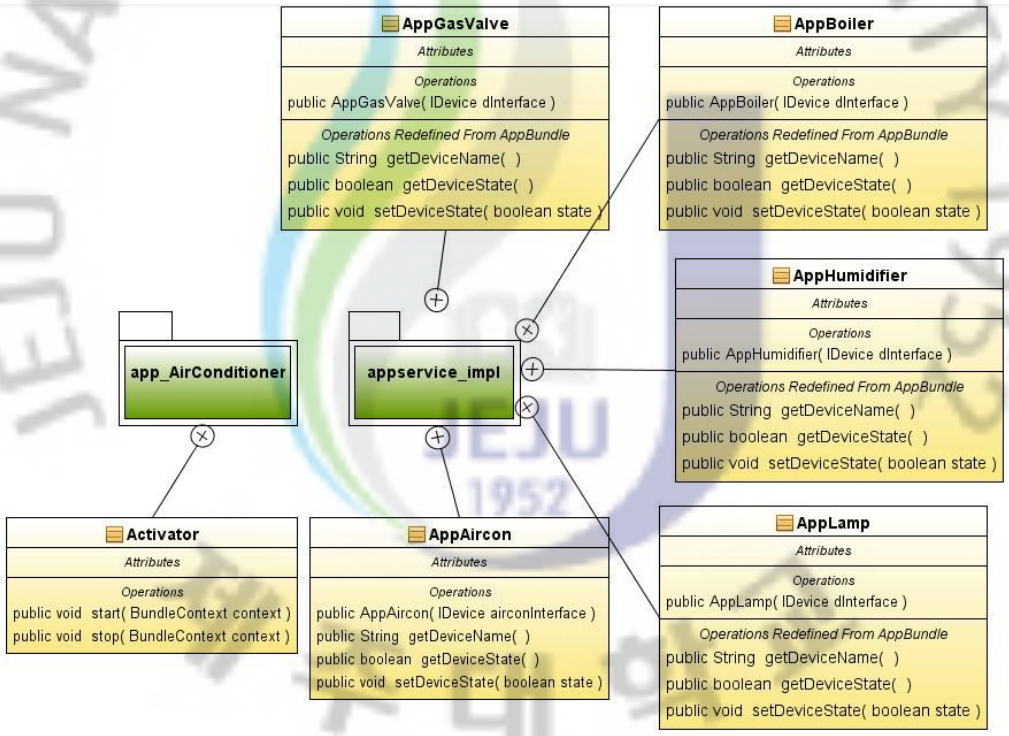

그림 10. OSGi 응용 번들 제어 API 클래스 다이어그램

또한, 그림 11은 SOAP을 이용하여 API를 제공하기 위해 SOAP 번들의 클래스 다이어그램을 보여준다.

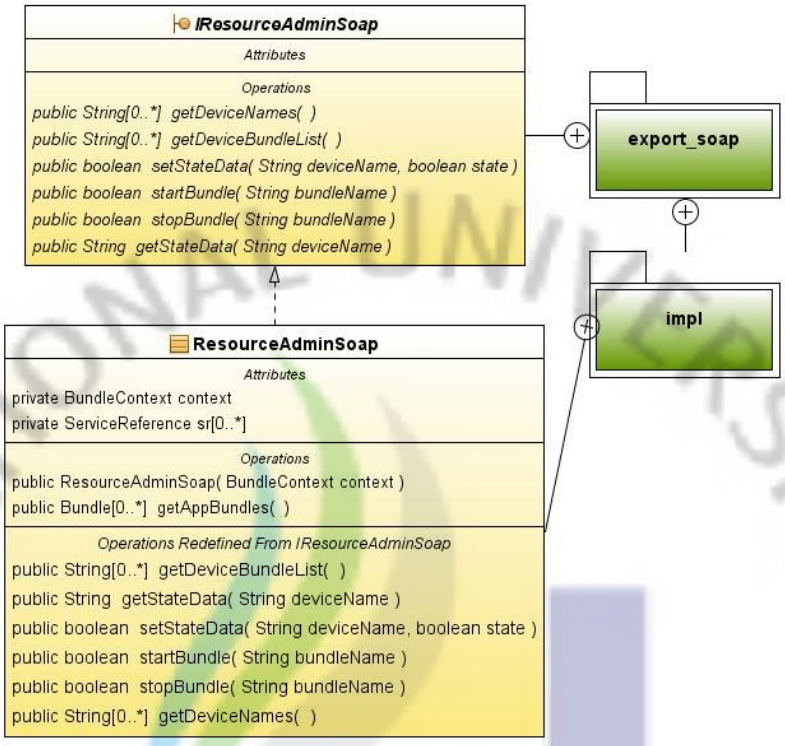

그림 11. SOAP 제공 API 클래스 다이어그램

위에 제시한 API 이외에도 다양한 API가 존재하며 표 5와 같다.

표 5. OSGi 번들제어 API 클래스

패키지	설명
appservice_Impl	각각의 홈 네트워크 기기를 제어하는 Device_Bundle를 제어하기 위하여 기기별로 제어 가능한 인터페이스를 정의한 패키지다. 이 패키지는 번들 형태의 서비스로 등록된다.
App_Airconditioner	각각이 홈 네트워크 기기를 컨트롤 하기위한 사용자 인터페이스로 사용자는 제공되는 API를 이용하여 번들들을 컨트롤 한다.
ResourceAddimSoap	관리자가 SOAP을 이용하여 번들 정보 데이터 수집 및 제어를 한다.

2) 데이터 요청 패키지

데이터 프로세스 패키지는 미들웨어에 데이터를 요청하기 위해서 ECSpec을 생성하고, 생성된 ECSpec을 XML 형태로 변환한 후 전송을 위한 번들들이다. 그림 12는 이들 패키지와 번들들을 보여준다.

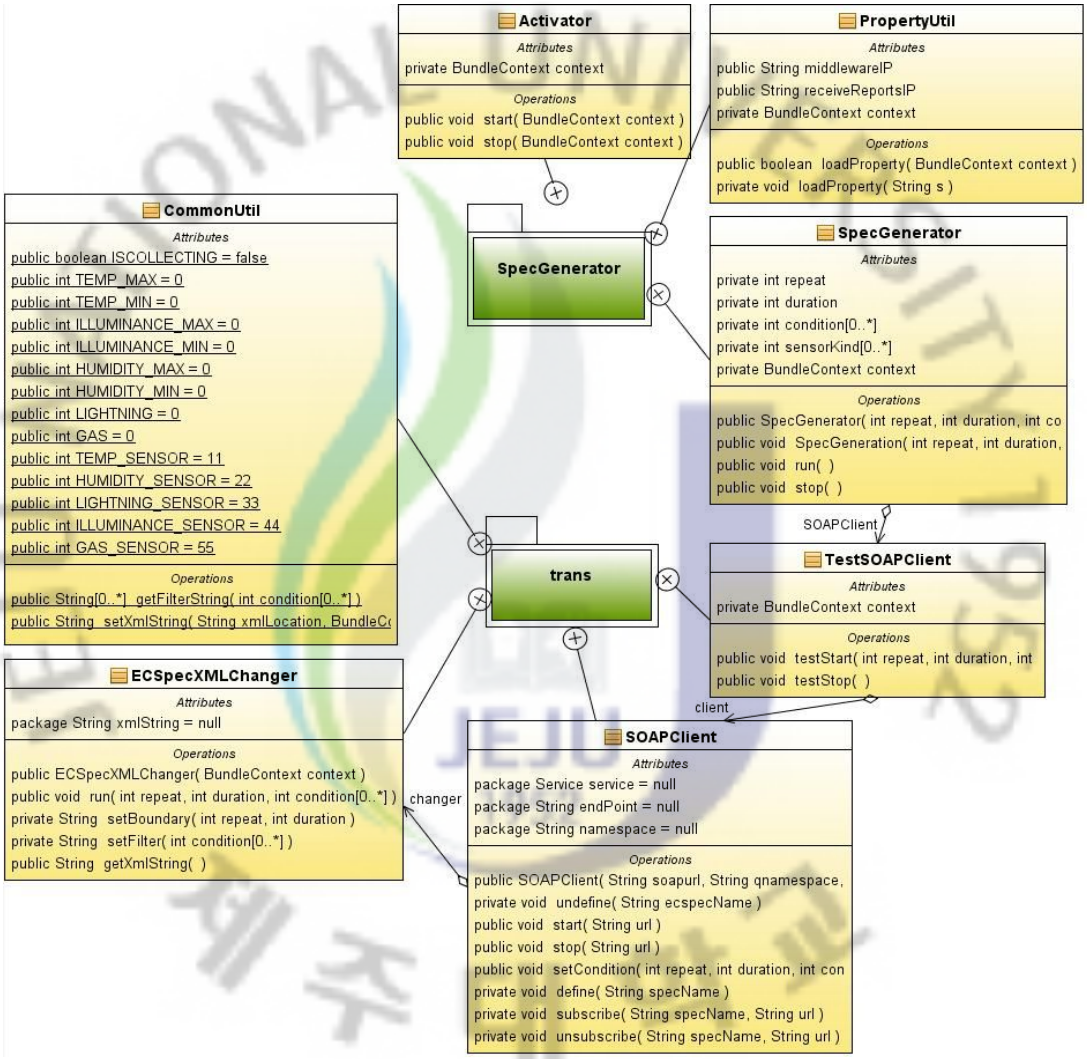

그림 12. 데이터 요청 처리 클래스 다이어그램

표 6. 데이터 요청 처리 주요 클래스

패키지	설명
CommonUtil	센서들의 수집 주기의 범위 및 센서들의 코드를 설정하는 클래스이다.
PropertyUtil	미들웨어로 전송될 ECSpec을 작성하기위해 미들웨어의 정보를 수집하는 클래스이다.
SpecGenerator	웹 서비스로 제공되는 API를 이용하여 ECSpec을 생성하는 클래스이다.
SOAPClient	생성된 XML문서를 미들웨어에게 전송한다.
ECSpecXMLChanger	ECSpec을 미들웨어에게 전송하기위해 XML 형태로 변경한다. 또한, 사용자의 필터링(요청 필터링) 조건을 추가한다.

3) 데이터 결과 분석 및 프로세스 패키지

미들웨어로부터 전송된 Reports를 분석하고 파싱하여 데이터를 처리하는 번들 패키지이다. 그림 13은 Reports를 수집하고 분석하는 클래스 다이어그램이다.

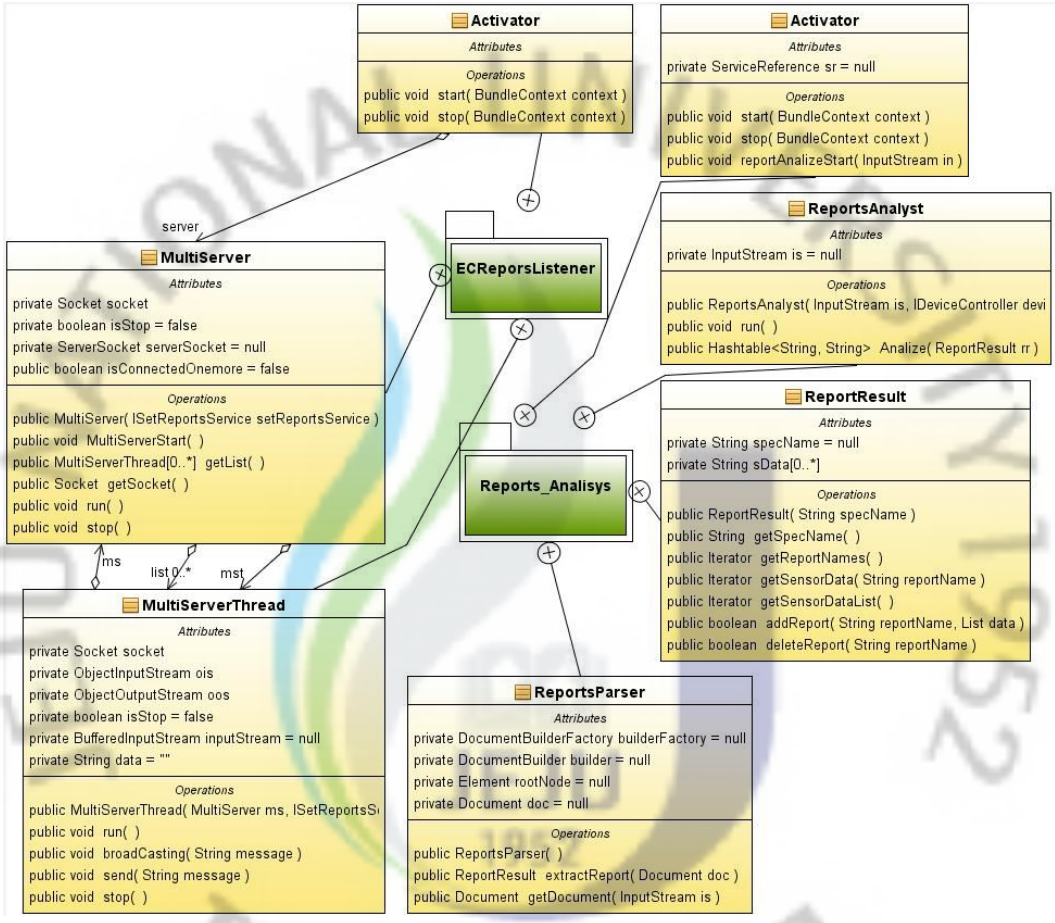

그림 13. 데이터 결과 분석 및 프로세스 클래스 다이어그램

표 7. 데이터 결과 분석 및 프로세스 클래스

패키지	설명
MultiServerThread	레포츠를 수집 받기 위한 Listener 클래스
ReportsParser	XML 문서와 레포츠를 파싱하기 위한 클래스
ReportsAnalyst	레포츠를 분석하여 데이터베이스에 저장
ReportResult	분석된 레포츠를 통해 수집된 데이터를 확인함

4) 데이터 관리 및 디바이스 제어 패키지

데이터 관리 번들 패키지에서는 획득된 데이터를 데이터베이스에 저장하고 디바이스 제어 번들 패키지는 미리 정의된 상황에 맞게 홈 네트워크 기기를 제어한다. 그림 14는 수집된 데이터를 바탕으로 디바이스를 제어하는 클래스 다이어그램이다.

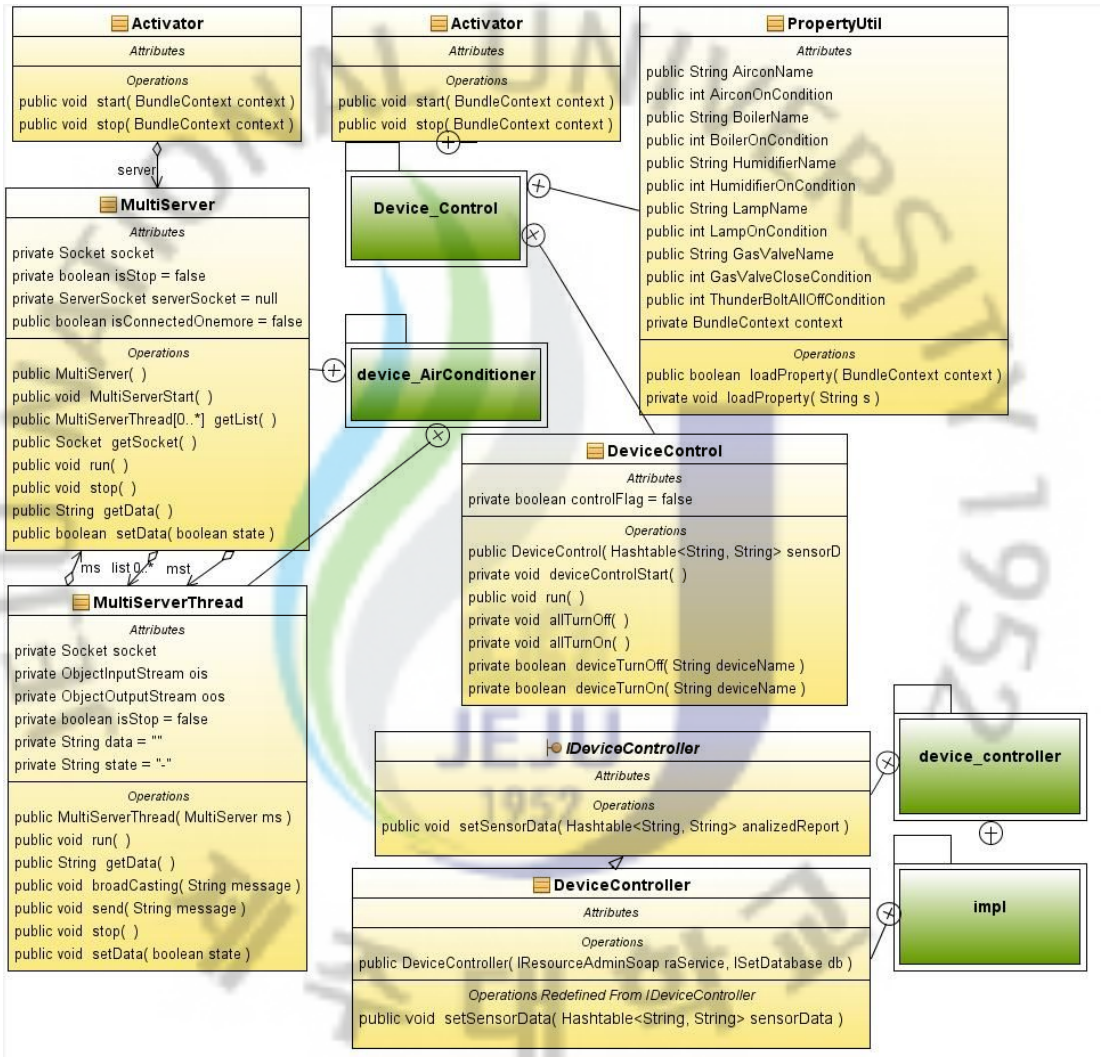

표 14. 데이터 관리 및 디바이스 제어 클래스 다이어그램

표 8. 데이터 관리 및 디바이스 제어 클래스

패키지	설명
MultiServerThread	홈 네트워크 디바이스들은 기본적으로 에뮬레이터 형태로 제작되어 OSGi응용과 소켓통신을 한다 따라서 디바이스의 정보 수집을 위한 쓰레드이다.
device_controler	사용자가 설정한 데이터를 해쉬 테이블에 저장한다.
Device_Control	사용자가 설정한 데이터를 해쉬테이블에서 불러와 조건이 일치하면 해당 private void deviceControlStart( )에뮬레이터 번들을(device_Aircondition) 실행시킨다.
PropertyUtil	OSGi 응용에서 사용하는 디바이스들의 정보를 담고 있다.

5. OSGi 응용에서 실시간 센서 데이터 스트림 처리

본 논문에서는 OSGi 응용에서 효율적인 실시간 센서 데이터 스트림 처리를 위해 사실상 국제 표준인 ALE 스펙을 기반으로 RFID/USN 등의 다양한 실시간 센서 데이터 스트림 처리가 가능한 미들웨어와 연동한다. 미들웨어는 기본적으로 웹 서비스를 제공하여 미들웨어에 접근 및 명령을 위한 다양한 ALE 인터페이스를 제공한다. OSGi 응용 개발자은 웹 서비스로 제공되는 WSDL 문서를 참고하여 미들웨어에 접근할 수 있다. 본 논문에서는 실시간 센서 데이터 스트림 처리 미들웨어와 연동하기 위한 OSGi 응용 프레임워크를 제안 하는 것이다. 그림 15는 OSGi 응용 프레임워크를 이용한 간단한 홈 네트워크 기기를 제어하는 OSGi 응용 이다.

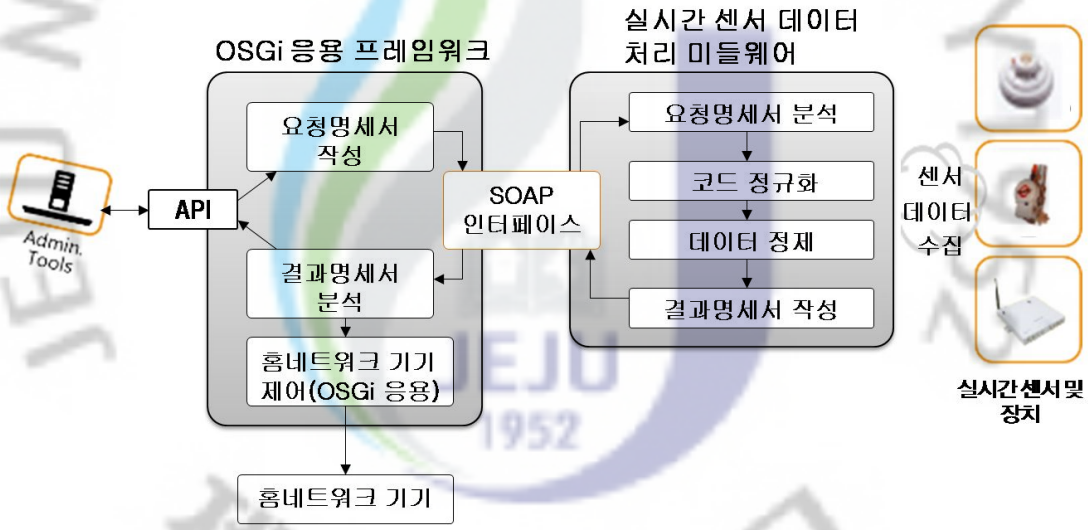

그림 15. OSGi 응용에서 실시간 센서 데이터 처리

관리자는 관리자 툴을 이용하여 스펙을 작성하지만 실제로는 OSGi 응용 프레임워크에서 제공하는 ECSpec 번들을 이용한다. ECSpec 번들을 통하여 미들웨어에게 데이터를 요청하고 요청된 데이터에 따라 미들웨어는 Reports를 생성하여 돌려준다. OSGi 응용은 받은 Reports를 분석하여 센서들의 값을 뽑아내고 관리자가 설정한 제어 상황에 맞게 홈 네트워크 기기들을 제어한다. 제어가 완료되면

데이터를 저장하고 해당 결과를 관리자에게 전송한다. 아래의 그림 16은 이러한 데이터의 흐름을 보다 자세히 보여준다.

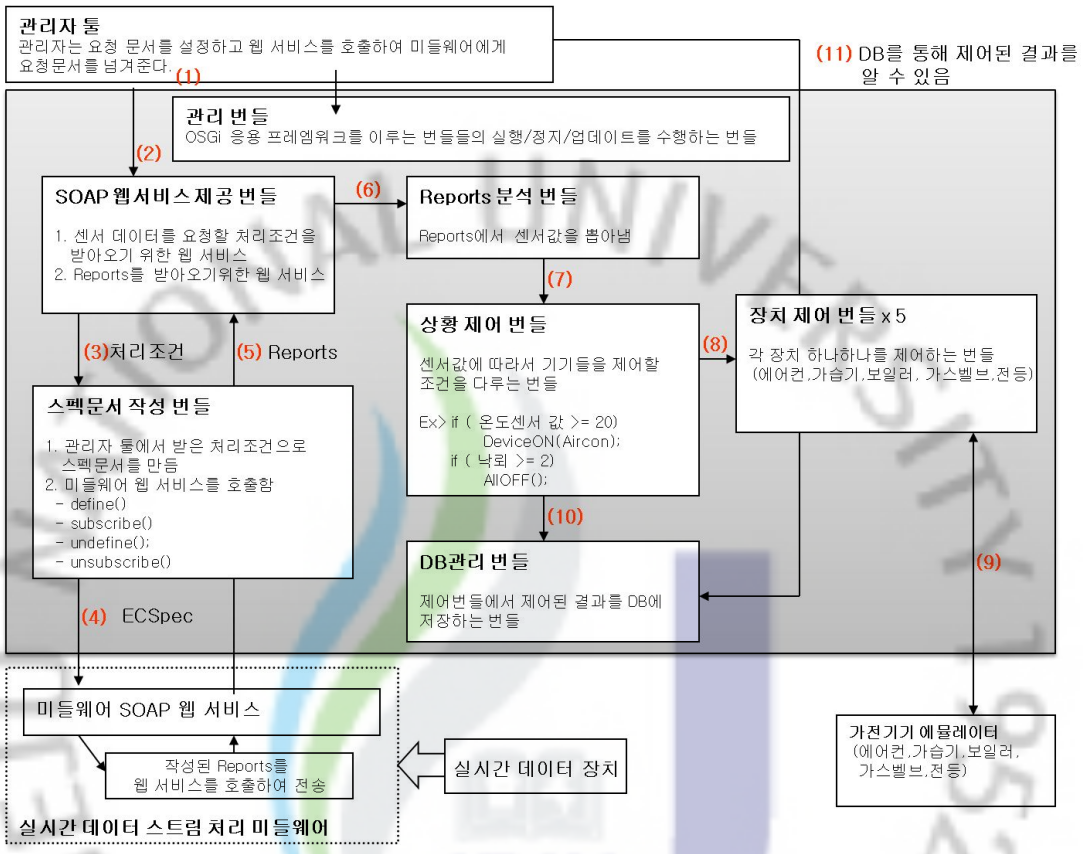

그림 16. OSGi 응용 프레임워크의 번들과 데이터 흐름

관리자 툴은 관리 번들을 통해 OSGi 응용 프레임워크에 존재하는 서비스들(번들)들의 실행/정지/삭제/ 등을 할 수 있다. 또한 번들들의 상태정보도 모니터링 할 수 있다. SOAP 웹 서비스 번들을 통해 센서 데이터를 요청하기에 앞서 문서의 이름, 처리조건 등의 인터페이스를 제공받을 수 있고, 이에 맞게 처리조건을 설정하여 스펙문서 작성 번들에다 전달하면 스펙 문서 작성 번들은 미들웨어 내부에서 사용가능한 ECSpec을 전송하게 된다. 본 논문에서는 간단한 ECSpec 문서를 미리 관리자 툴에 저장하여 구현 하였다. ECSpec을 미들웨어로 전송하면 미들웨어는 반환 값으로 Reports를 생성한다. Reports를 전송하면 OSGi 응용 프레임워크에 있는 Reports 분석 번들을 통해 XML 문서를 파싱하여 센서 값을 얻

어낸다. 획득한 센서 값을 바탕으로 관리자 및 사용자가 미리 정의해놓은 상황제어 번들에서 장치제어 번들을 실행 시키고 실질적으로 홈 네트워크 기기를 제어한다. 이후 제어된 결과를 데이터베이스에 저장하고 해당 결과 값을 관리자에게 전송한다. 아래 그림 17은 이러한 일련의 과정을 순차도로 나타낸 모습이다.

그림 17. OSGi 응용 프레임워크의 실시간 센서 데이터 순차도

IV. 구현 및 실험

1. 개요

본 논문에서는 OSGi 기반 응용 서비스에서 효과적인 실시간 데이터 스트림 처리를 할 수 있도록 설계한 OSGi 응용 프레임워크를 구현하여 국제 표준을 따르는 실시간 데이터 스트림 처리 미들웨어와 연동한다.

OSGi 기반 응용 서비스에서 사용될 OSGi 응용 프레임워크는 번들 형태로 구현되며, 해당 번들들을 OSGi 프레임워크의 서비스로 등록하고 이 서비스(Bundle)를 사용하는 OSGi 응용을 구현한다. OSGi 응용은 사용자에 의해 정의된 상황에 맞게 간단한 홈 네트워크 기기를 제어한다. OSGi 응용 프레임워크는 홈 네트워크 기기를 제어하는 OSGi 응용뿐만 아니라 실시간 센서 데이터 스트림 처리를 필요로 하는 OSGi 응용에서 사용가능하다. 이 장에서는 Ⅲ장에서 OSGi 번들 기반으로 설계한 OSGi 응용 프레임워크의 구현과 홈 네트워크 기기를 제어하는 OSGi 응용 그리고 가전기기 및 센서 에뮬레이터를 구현하여 작성된 시나리오를 바탕으로 효율성 및 자체 테스트 수행을 위한 결과를 분석한다.

2. 구현환경

본 논문에서 제안한 OSGi 응용 프레임워크는 표 9의 환경에서 구현하였다.

표 9. 구현환경

구분	하드웨어	소프트웨어	비 고
Real-Time Sensor Data Stream Middleware	Intel(R) Core(TM)2 CPU 6300 @ 1.86GHz, 3,00GB RAM	Microsoft Windows XP Pro Sun Java SDK 1.5.0_07 Eclipse SDK 3.2	Java
OSGi Application Framework	Intel(R) Core(TM)2 CPU 6300 @ 1.86GHz, 3,00GB RAM	Microsoft Windows XP Pro Sun Java SDK 1.5.0_07 Eclipse SDK 3.2 Knoplferfish OSGi 2.1	Java
Manager Tool	Intel(R) Core(TM)2 CPU 6300 @ 1.86GHz, 3,00GB RAM	Microsoft Windows XP Pro Microsoft Visual Studio 2005	C#
Sensor Emulator	Intel(R) Core(TM)2 CPU 6300 @ 1.86GHz, 3,00GB RAM	Microsoft Windows XP Pro Sun Java SDK 1.5.0_07 Eclipse SDK 3.2	Java
Home Network Device	Intel(R) Core(TM)2 CPU 6300 @ 1.86GHz, 3,00GB RAM	Microsoft Windows XP Pro Sun Java SDK 1.5.0_07 Eclipse SDK 3.2	Java

OSGi 응용 프레임워크는 번들형태로 구현되며 번들들이 돌아 갈 수 있는 OSGi 프레임워크로 Konplferfish를 이용하였다[17]. Konplferfish는 R3와 R4스펙을 지원한다. 관리자 툴은 C#으로 구현 하였으며, 실시간 센서 와 홈 네트워크 기기들은 Java를 이용한 간단한 에뮬레이터 형태로 구현 하였다. 제안하는 OSGi 응용 프레임워크는 이클립스를 이용하여 자바로 그림 18과 같이 구현하였다.

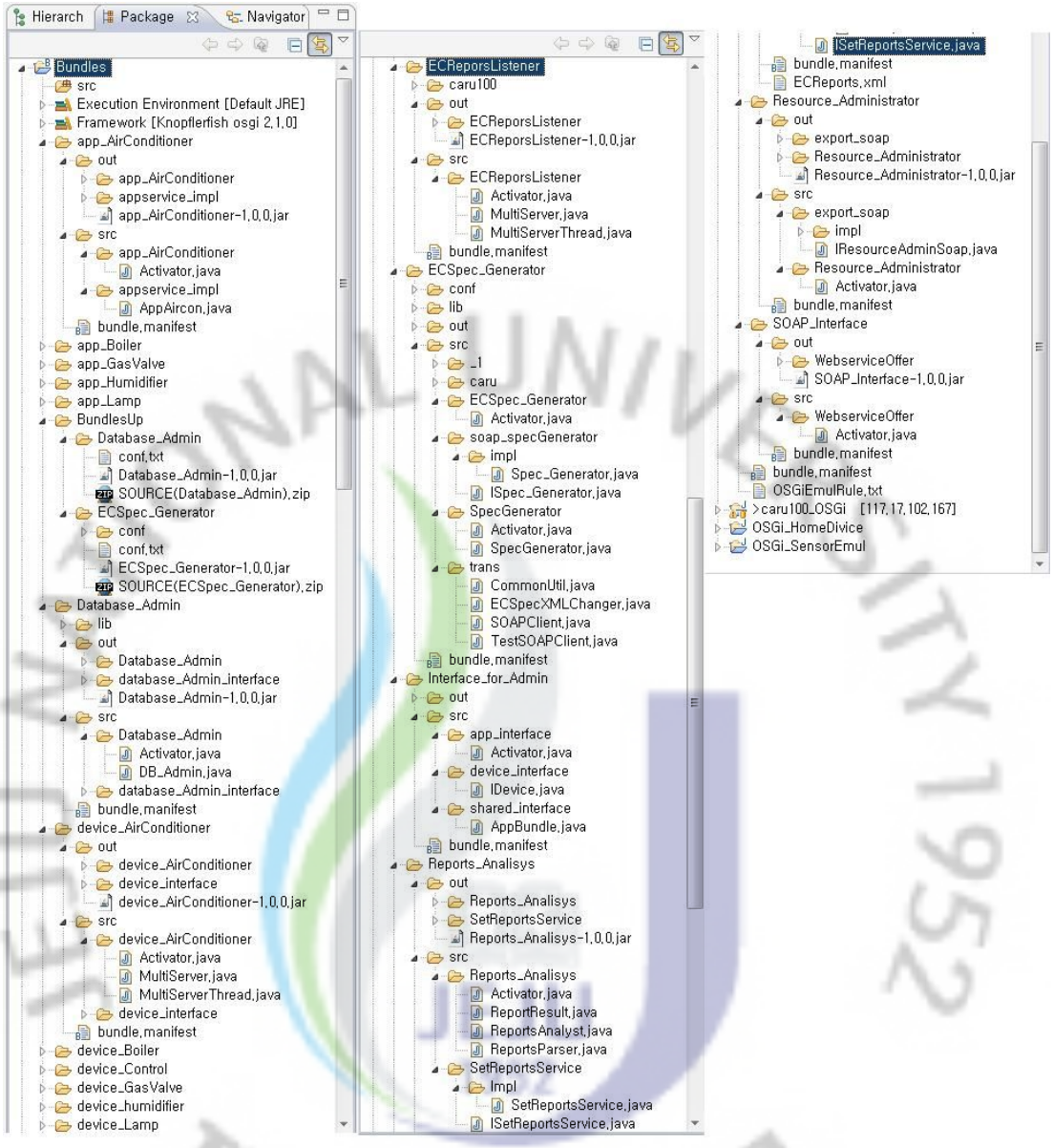

그림 18. OSGi 응용 주요 패키지 구현

OSGi 응용은 응용 번들과 OSGi 응용 프레임워크에 존재하는 번들들로 구성된다. 아래의 그림 19는 최종적으로 OSGi 응용 번들들의 주요 서비스 패키지와 번들을 보여주고 있다.

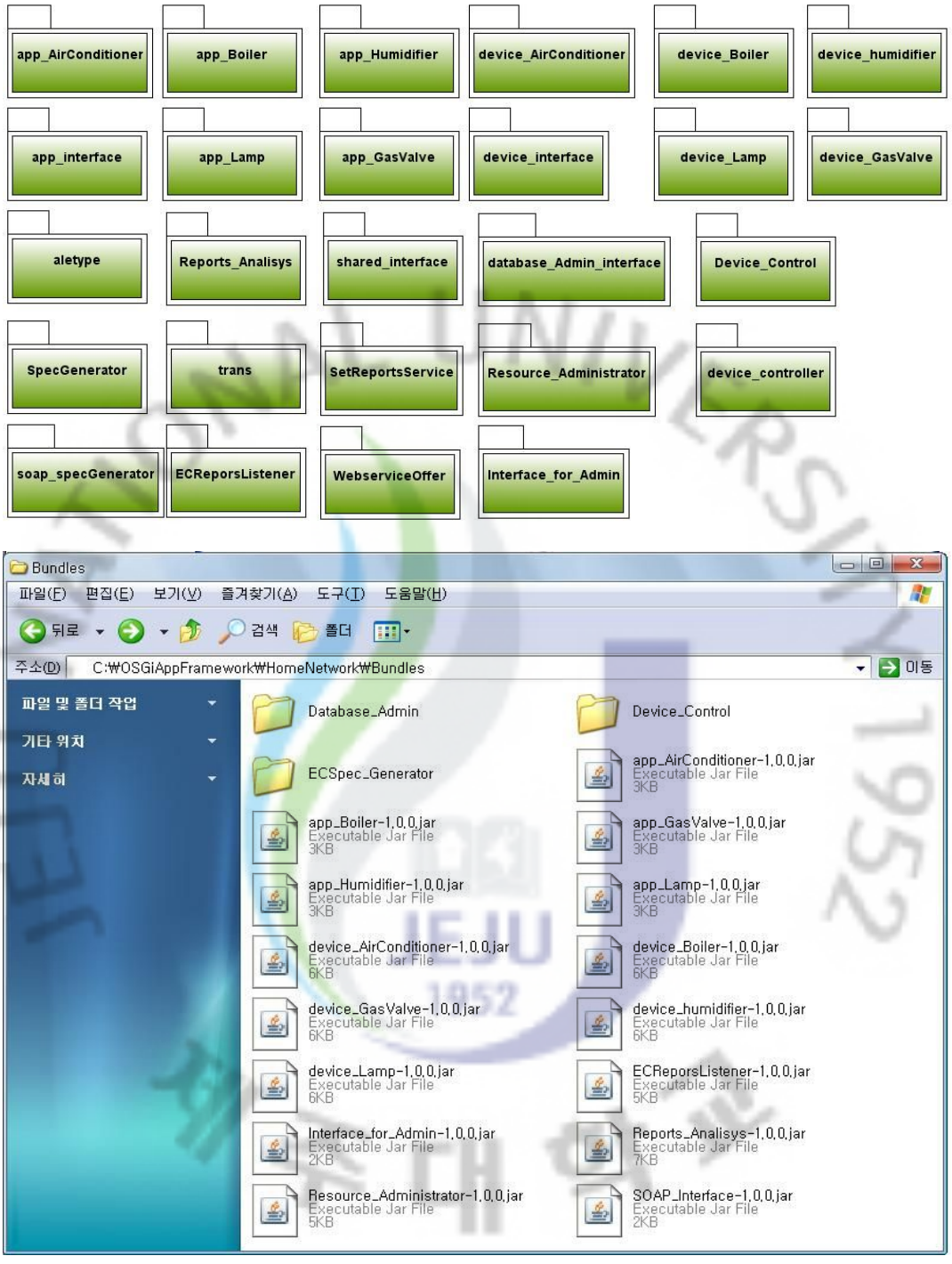

그림 19. OSGi 응용 주요 서비스 패키지 및 번들

3. 실험 내용 및 결과

본 실험에서는 OSGi 기반 홈 네트워크 기기 제어 응용 서비스가 실시간 센서 스트림 처리 미들웨어에게 데이터를 요청 했을 때, 처리된 센서 데이터를 바탕으로 홈 기기들이 제어 여부를 실험 했다. C#으로 제작된 OSGi 응용 관리자 툴을 사용하여 번들의 실행과 정지 여부를 테스트하고 수집되는 데이터의 필터링 조건의 주어 꼭 필요한 데이터만 OSGi 응용으로 수집되도록 했다. 홈 네트워크 기기 에뮬레이터는 번들 형태로 구현되어 기기의 On, Off의 제어를 할 수 있도록 하였다. 온도, 습도, 가스, 조도, 낙뢰 센서에서 총 초당 100개의 실시간 센서 데이터 스트림을 생성 하도록 하였다. 센서 에뮬레이터로 부터 초당 약 100개의 실시간 센서 데이터를 획득하여 실시간 센서 스트림 처리 미들웨어에서 필터링 등의 가공 처리를 하여 OSGi 응용으로 전송한다.

1) OSGi 응용을 위한 관리자 툴

그림 20은 OSGi 응용에서 제공하는 관리자 툴이다.

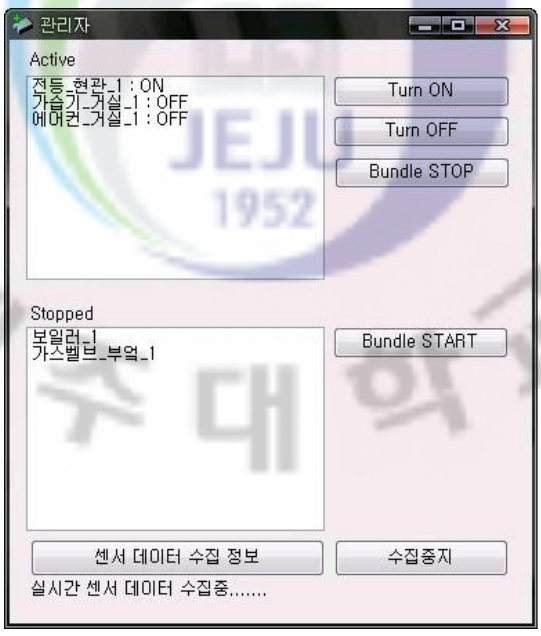

그림 20. 관라자 툴

관리자 도구를 이용하면 제어할 수 있는 응용 번들의 동작 상태를 확인할 수 있고, 응용 번들(App_Airconditioner)이 동작중이라면 그 번들을 통하여 가전기기를 제어 할 수 있다. 관리자 툴에서 센서 데이터 수집정보 버튼을 클릭하여 센서 주집 정보를 설정할 수 있으며 그림 21과 같다.

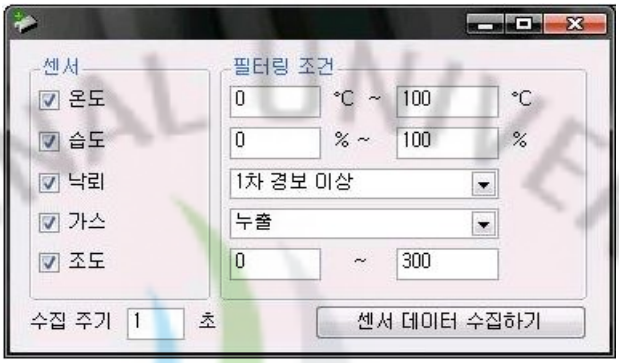

그림 21. 관리자 툴에서 센서 정보수집 환경설정

관리자 번들에서 보이는 홈 네트워크 기기들은 SOAP 번들을 통해 관리자 툴로 전송되며 번들을 관리하는 관리번들에게 gerDeviceBundleList() 호출하면 그림 22와 같은 XML 문서를 전송한다. 관리자 툴은 해당 문서를 파싱하여 현재 홈 네트워크에 존재하는 전등, 가습기, 에어컨, 보일러, 가스밸브의 환경 정보를 관리자에게 제공한다.

```
<?xml version="1.0" encoding="utf-8" ?>
- <ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://localhost/literalTypes">
  <string>전등_현관_1:Active:ON</string>
  <string>가습기_거실_1:Active:OFF</string>
  <string>에어컨_거실_1:Active:OFF</string>
  <string>보일러_1:Stopped:</string>
  <string>가스벨브_부엌_1:Stopped:</string>
</ArrayOfString>
```

그림 22. 관리자 툴에서 기기들의 정보 회득

2) 미들웨어 실행과 센서 데이터 획득

그림 23은 실시간 센서 데이터 스트림 처리 미들웨어를 실행한 모습이다.

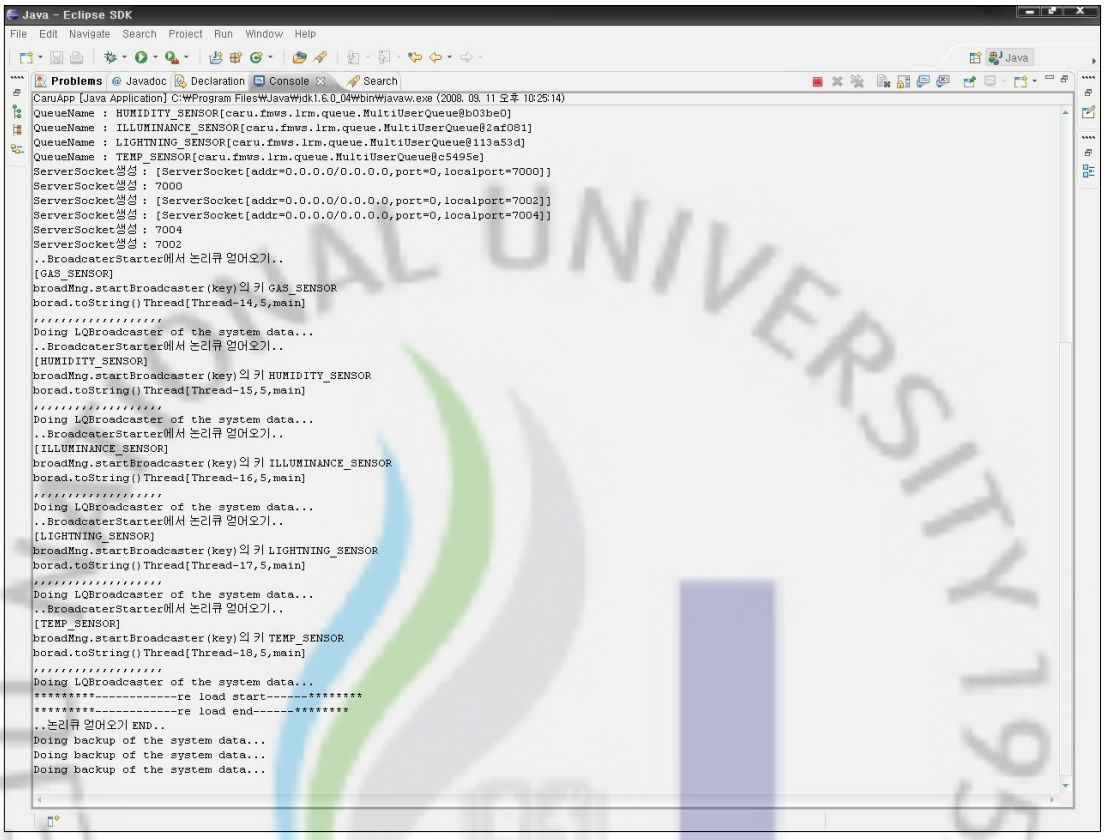

그림 23. 실시간 센서 데이터 스트림 처리 미들웨어

실시간 센서 데이터 스트림 처리 미들웨어를 실행하고 미들웨어에 연결된 5개의 센서에서 데이터를 생성시키면 미들웨어에서는 데이터를 수집하여 처리한다. 그림 24는 미들웨어의 센서 데이터 수집 상황을 보여준다.

그림 24. 미들웨어의 센서 데이터 수집

3) OSGi 프레임워크 실행 및 번들 실행

실험에서는 OSGi 프레임워크로 knopflerfish 2.1 버전을 사용한다. 아래 그림 25는 OSGi 프레임워크를 실행한 모습을 보여주며 이미 홈 네트워크 기기제어 응 프레임워크에서 구현한 번들을 인스톨한 상태이다. 이 상태에서는 OSGi 응용에서 실제로 실시간 센서 데이터를 요청 하지 않았음으로 수집되는 방대한 양의 실시간 센서 데이터에 대한 부하가 없다.

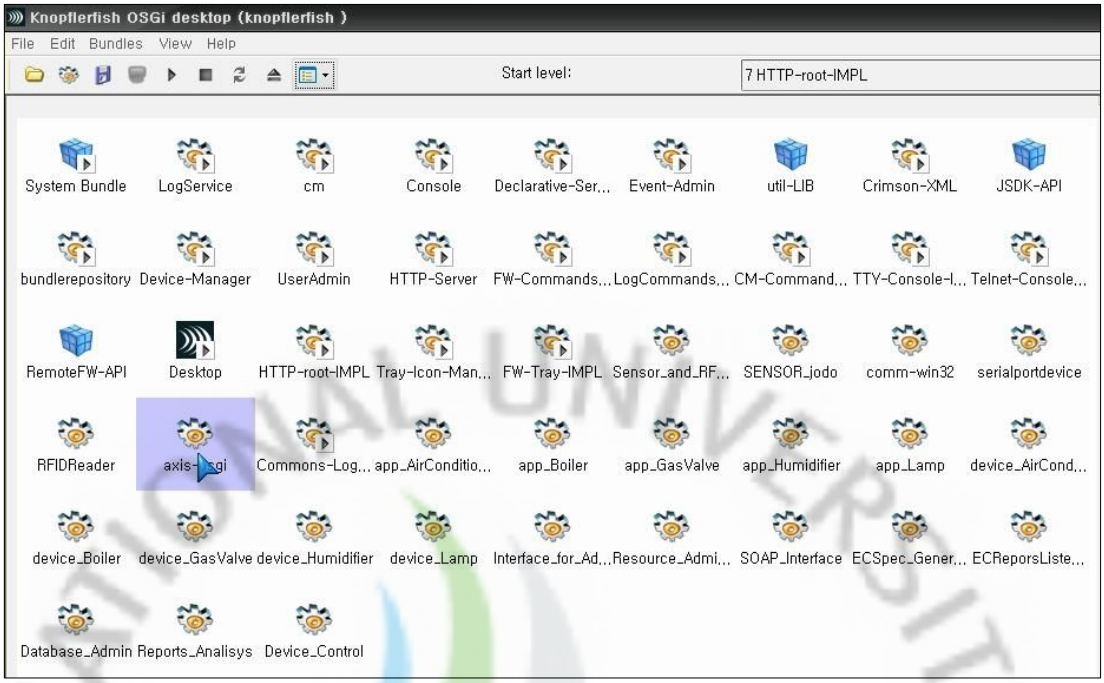

그림 25. OSGi 프레임워크 Knopflerfish

해당 번들들을 실행하면 OSGi 응용 서비스와 홈 네트워크기기 에뮬레이터, 관리자 툴, 미들웨어가 모두 연동이 된다. 홈 네트워크 기기 제어 응용 이외에 OSGi 기반 응용 서비스에서 위의 번들을 인스톨 하면 실시간 센서 데이터 스트림 처리 미들에어를 호출하여 데이터를 획득할 수 있다.

4) 실험 결과

OSGi 응용 서비스는 실시간 센서 데이터 스트림 처리를 위해 OSGi 응용 관리자 툴에서 미들웨어로 데이터를 요청하고 요청된 결과를 바탕으로 홈 네트워크 기기를 제어한다. 그림 26은 실시간 센서 데이터를 바탕으로 가전기기를 제어하고 가전기기가 제어 될 당시의 센서 데이터 값을 저장한 모습이다.

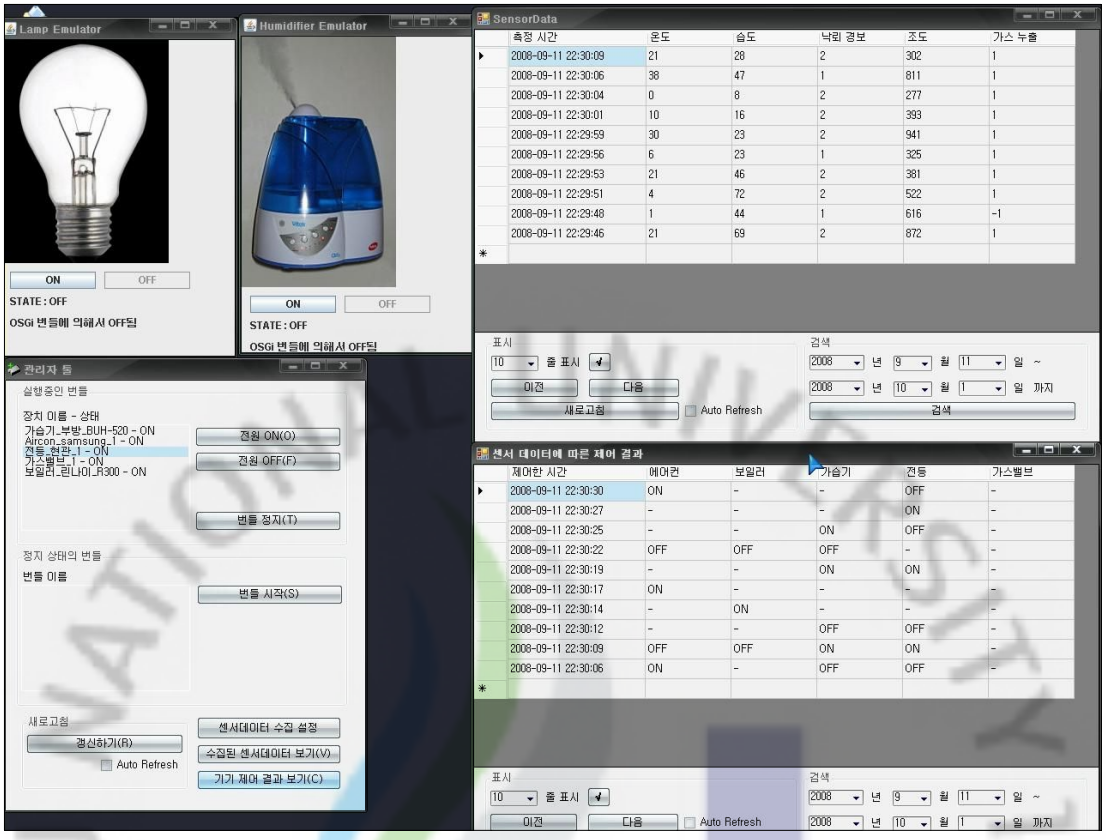

그림 26. OSGi 응용에서 실시간 센서 데이터 스트림 처리 결과

초기 센서 에뮬레이터는 초당 약 100개의 실시간 센서 데이터를 발생 시키도록 하여 관리자나 사용자가 설정한 필터링 조건에 맞게 필터링 하여 OSGi 응용으로 전달되는 실시간 데이터는 초당 약 10개 정도 이다.

V. 결론 및 토의

OSGi 플랫폼 환경에서는 대량의 실시간 스트림 데이터를 효과적으로 처리하기 위한 방법에 대한 고려가 충분치 않다. 따라서 본 논문에서는 OSGi 프레임워크 기반 응용 시스템이 실시간 데이터 스트림을 효과적으로 처리할 수 있는 방법을 제안하였다. OSGi 응용 서비스들은 기본적으로 임베디드 환경을 고려해야 함으로 시스템 자원의 한계가 있다. 따라서 본 논문에서는 OSGi 응용과 실시간 센서 데이터 처리 미들웨어를 독립 적으로 구성 하였다. 실험을 통하여 실제로 초당 100개의 센서 데이터가 생성되지만 OSGi 프레임워크로 가는 데이터의 양은 극히 소수로 OSGi 프레임워크 시스템 자원의 효율을 높일 수 있다.

제안하는 방법을 사용하면 OSGi 기반 응용에서 양질의 실시간 데이터를 사용하여 다양한 응용을 서비스 할 수 있으며 실시간 데이터 처리 미들웨어에 대해 독립적인 OSGi 응용 서비스의 구현이 가능하다. 또한, 사실상의 국제 표준인 ALE 기반의 미들웨어에 대해서는 상당 부분 연구가 진행되어 OSGi 응용 개발자는 실시간 센서 데이터 처리에 대하여 비교적 적은 노력으로 원하는 스트림 데이터를 효과적으로 획득할 수 있어 양질의 응용 서비스를 쉽게 구현할 수 있다.

본 방법은 OSGi 프레임워크 응용 서비스를 구현하여 기존의 실시간 데이터스트림 처리 미들웨어와 연동하고 미들웨어에서 처리된 데이터를 이용한 홈 네트워크 기기를 제어함으로써 다양한 OSGi 응용을 위한 OSGi 응용 프레임워크의 유용성을 입증하였다. 홈 네트워크에 다양한 실시간 센서가 도입되거나 확장될 때 별도의 시스템 도입 없이 기존의 국제 표준 미들웨어를 사용하여 현재 홈 네트워크의 단점을 보안하고 경제적인 효과를 가질 수 있다. 나아가 OSGi 응용을 임베디드 시스템 환경에서 테스트를 수행하고 실시간 데이터 스트림 처리 미들웨어와 OSGi 프레임워크 연동 방법으로서 stand-alone 방식과 embedded 방식을 연구 해봄으로써 보다 현실성 있는 유비쿼터스 홈 네트워크 환경을 실현할 수 있을 것으로 판단된다.

참고문헌

[1] M. Weiser, J. S. Brown, “The coming of age of calm technology”, Xerox PARC, 1996

[2] B. Rose, “Home Networks: a standards perspective” Communications Magazine, IEEE, Vol.39, Issue:12, pp.78-85, 2001

[3] 이현규, “이현규의 홈네트워킹 대해부 1편-15편” iNews24 전문가 리포트, 2003 3-6

[4] 장병준, 안선일, 이윤덕, "RFID/USN 기술개발 동향", 한국정보학회, 정보과학회지, 제23권 제2호, 2005.

[5] 김석우, “유니버설 미들웨어 프레임워크” - OSGi 1회-8회 마이크로 소프트웨어, pp.204-209, (주)마소인터렉티브, 2007 7월-12월

[6] 권정혁, “실전 OSGi & SpringDM”, pp.5-15, 위키북스, 2009, 10

[7] L. Gong, “A Software Architecture for Open Service Gateways,” IEEE Internet Computing, Vol.5, Issue:1, pp.64-70, 2001

[8] OSGi Alliance, “ The OSGi Service Platform ” http://www.osgi.or

[9] K. Chen, Programming Open Service Gateway with java Embedded Server Technology. Addson-Wesley, 2001

[10] 서대영, “개방형 홈 네트워크 서비스 제어를 위한 프레임워크 기술,” 한국통신학회지, 제22권, 5호, pp.68-77, 2005. 5

[11] 홍연미, 변영철, “ALE 기반 RFID 미들웨어 설계 및 구현,” 한국해양정보통신 학회논문집, 제11권, 제4호, pp.648-655, 2007.

[12] 양문석, 변영철 “RFID 미들웨어 기반 센서 데이터 스트림 처리 방법”, 한국해양정보통신 학회논문집, 제12권, 제2호, pp.231-239, 2008

[14] 나선웅, 이상정, 김동균, 최영길, ‘무선센서 네트워크를 이용한 지능형 홈 네트워크 서비스 설계, 한국컴퓨터정보 학회논문집, 제11호, 제5호, pp.183-193, 2006

[15] 온톨로지를 이용한 효율적인 EPC URN 코드 변환에 대한 연구 ALE

[16] Version 1.0, EPCglobal Working Draft, Oct. 14, 2004.

[17] http://www.knopflerfish.org/releases/current/repository.xml

[18] 오세원, 박주상, 이용준, "RFID SW기술과 표준화 동향" 한국통신학회지. 제24권 제7호, pp.17-25 2007.

[19] 김민수, 김광수, 이용준, "USN 미들웨어 특징 및 기술개발 동향", ERRI 주간기술동향 통권 1284호, 2007.

[20] 김민수, 이용준, 박종헌, ETRI, "USN 미들웨어 기술개발 동향", 전자통신동향분석, 제22권, 제3호, 2007.6

[21] 김민수, 김광수, 이용준, "USN 미들웨어 특징 및 기술개발 동향", ERRI 주간기술동향 통권 1284호, 2007.

[22] 김대영, 성종우, 송현주, 김수현, "센서 네트워크 미들웨어 기술", 전자공학회지 , 제32권 제7호, pp.32-46 2005.