oneM2M

IoT 오픈 플랫폼 기반 제품서비스 개발을 위한

IoT와 oneM2M의 이해

2017

nipa 정보통신산업진흥원 | KETI 전자부품연구원 | TTA | 한국사물인터넷협회

「IoT 오픈플랫폼 기반 개발검증지원 인프라 구축」

이 강의 교안은 미래창조과학부의
정보통신 · 방송 연구개발 사업을 통해 개발되었습니다.

목 차

1장. 사물인터넷 개요

Chapter 1. Introduction of Internet of Things

1.1 사물인터넷 개요 및 아키텍처 소개
1.2 사물인터넷 표준화
1.3 사물인터넷 플랫폼 아키텍처
1.4 오픈 소스 H/W 플랫폼
1.5 비즈니스 사례

- Assignment
- Reference

강의 개요

▪ 강의 목표

사물인터넷(Internet of Things, IoT)의 개념과 아키텍처를 학습한다.

▪ 강의 내용

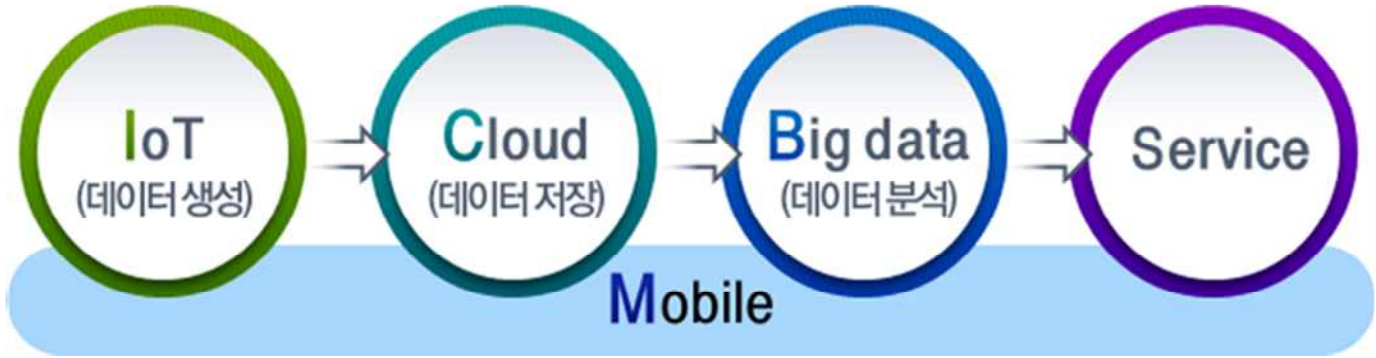
- 사물인터넷 개념 및 등장배경
- 사물인터넷 표준화 기구
- 사물인터넷 아키텍처 레퍼런스 모델
- 사물인터넷 플랫폼 개요 및 구조
- 사물인터넷 플랫폼 기술

1.1 사물인터넷 개요 및 아키텍처 소개

▪ 사물인터넷 개념(1/2)

사물인터넷(Internet of Things, IoT)은 사람, 사물, 공간, 데이터 등 모든 것이
인터넷으로 서로 연결되어, 정보가 생성 · 수집 · 공유 · 활용되는 초연결 인터넷

모바일 기반에서 IoT, 클라우드(Cloud), 빅데이터(Big Data)가
연계된 새로운 가치와 서비스 창출

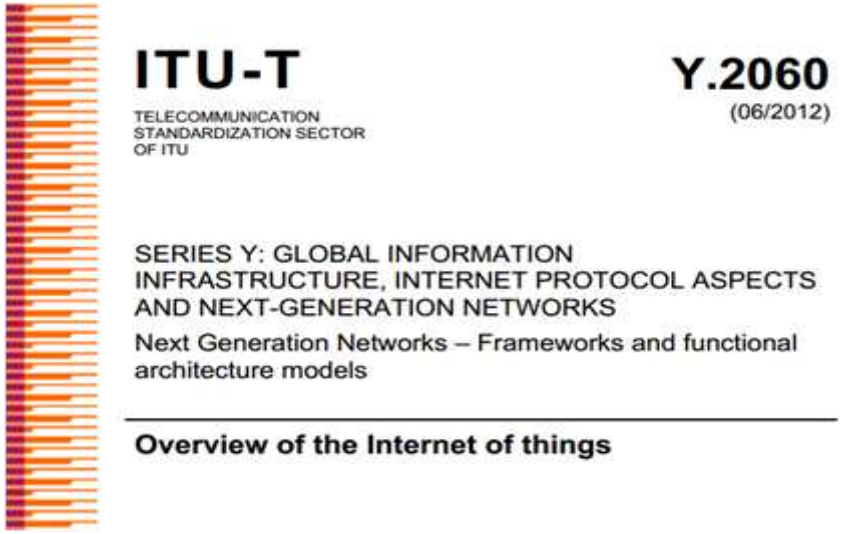


1.1 사물인터넷 개요 및 아키텍처 소개

▪ 사물인터넷 개념(2/2)

사물인터넷은 이미 존재하거나 향후 등장할 상호 운용 가능한 정보 기술과 통신 기술을 활용하여 다양한 실재 및 가상 사물 간의 상호 연결을 통해서, 진보된 서비스를 제공 수 있게 하는 글로벌 인프라스트럭쳐임(ITU-T의 Y.2060)

센서. 상황 인지 기술, 통신. 네트워크 기술, 칩 디바이스 기술, 경량 임베디드 네트워크 기술, 자율적. 지능형 플랫폼 기술, 대량의 데이터를 처리하는 빅데이터 기술, 데이터 마이닝 기술, 사용자 중심의 응용 서비스 기술, 웹 서비스 기술, 보안. 프라이버시 보호 기술 등


ITU-T

TELECOMMUNICATION STANDARDIZATION SECTOR OF ITU

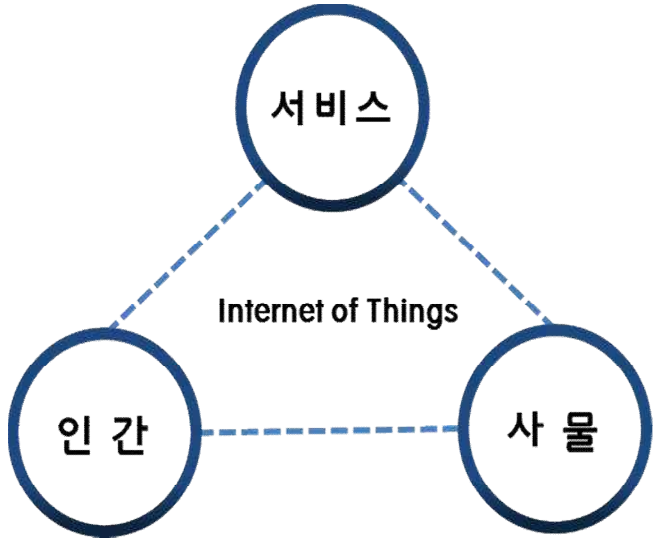
Y.2060

(06/2012)

SERIES Y: GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS

Next Generation Networks – Frameworks and functional architecture models

Overview of the Internet of things



1.1 사물인터넷 개요 및 아키텍처 소개

▪ M2M

- 사람이 직접 제어하지 않는 상태에서 장비나 사물 또는 지능화된 기기들이 사람을 대신해 통신의 양쪽 모두를 맡고 있는 기술을 의미
- 센서 등을 통해 전달, 수집, 가공된 위치, 시각, 날씨 등의 데이터를 다른 장비나 기기 등에 전달하기 위한 통신을 의미
- 일반적으로 사람이 접근하기 힘든 지역의 원격제어나 위험 품목의 상시 검시 등의 영역에서 적용
- USN이 넓은 지역에 대한 상태 정보를 필요로 하는 반면에, M2M은 개별 장치들에 대한 연결성을 제공하는 것이 기본 목적 임

구분	RFID/USN/M2M		사물인터넷(IoT)
통신/네트워크	근거리망, 이동망 중심	→	인터넷 중심
디바이스의 형태	센서 중심	→	센서와 액츄에이터의 Physical Thing과 데이터와 프로세스 등을 포함한 Virtual Thing
디바이스의 서비스 구동 수준	단순 정보 수집/수동적	→	자율 판단하는 지능 보유/자율적
서비스 플랫폼	모니터링 정보 처리	→	의미 기반 모니터링 및 자율 제어
서비스 관리 규모	수천만 개의 사물	→	수백 억 이상의 사물
서비스 적응성	통시적 서비스 제공	→	즉시적 스마트 서비스 제공

1.1 사물인터넷 개요 및 아키텍처 소개

▪ 사물인터넷 등장배경

– 기술적 측면의 사물인터넷 활성화 요인

구분	주요내용
소형화	• MEMS나 나노기술(Nano-Technologies)과 같은 반도체 기술의 발전은 전자제품에 소용되는 소자(component)의 크기를 극적으로 작게 만들고 있음 -제품의 소형화는 물론 저전력화, 대량생산에 따른 저가격화에도 영향을 주고 있음
저전력화	• 다양한 액세서리 디바이스들이 전력소모를 최소화하기 위해 저전력 블루투스(Bluetooth Low Energy BLE) 기술을 채택하고 있음 -BLE는 단방향 통신 방식을 채택하고 있으며, 패킷 전송 주기를 늘리고 최대 송신 전력을 줄여 실제로는 와이파이 대비 수 백 분의 1정도의 전력만을 소모함
저가격화	• RFID 태그 가격은 2013년 4월 기준으로 지난 18개월간 40% 가량 하락하였으며, MEMS는 지난 5년간 80~90% 정도 하락함, 센서의 개당 평균 가격도 2004년 1.3달러에서 2014년 0.6달러 수준으로 떨어졌으며, 1Gbps 단위의 인터넷 비용은 10년 전 대비 1/40 수준으로 떨어지고 있음
표준화	• 표준화된 무선통신방식이나 개방형 표준 인터페이스를 이용해 통신 칩셋들도 표준화 되어, 다른 디바이스들과 데이터를 주고받을 수 있도록 모듈화 되어 제작됨 • 누구나 새로운 디바이스를 손쉽게 제작할 수 있게 되고, 사물인터넷 플랫폼이 제공하는 표준 API를 통해 다른 디바이스들과 연결됨

1.2 사물인터넷 표준화

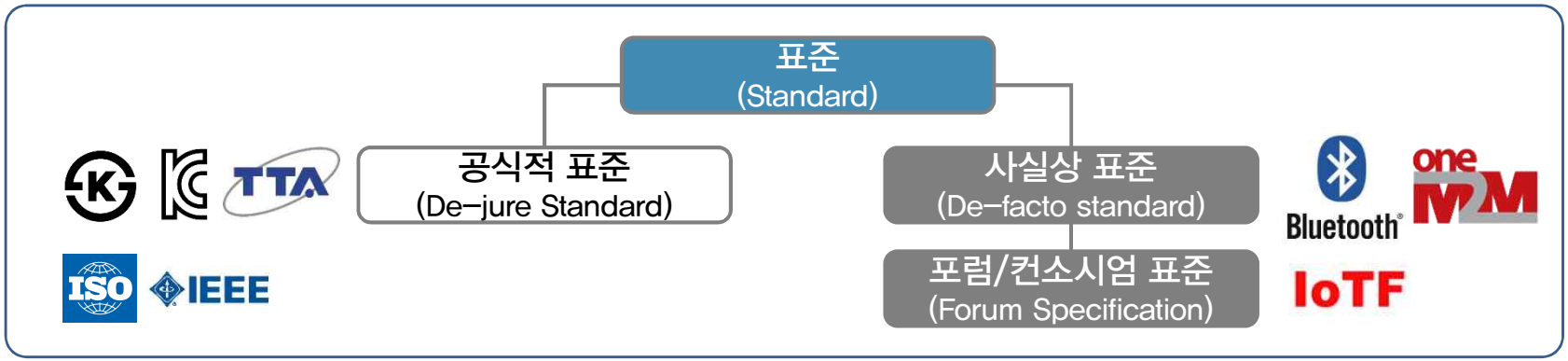
▪ 표준화 개념 및 표준화 기구 개요

– 표준의 정의

어떤 양을 재는 기준으로 쓰기 위하여 어떤 단위나 어떤 양의 한 값 이상을 정의하거나 현시하거나 보존하거나 또는 재현하기 위한 물적 척도, 측정 기기, 기준물질이나 측정 시스템을 말함
(한국표준과학연구원(www.kriss.re.kr))

– 정보통신(ICT) 분야 표준의 정의

정보통신망과 정보통신 서비스를 제공하거나 이용하는 주체끼리 합의된 규약의 집합으로 공통성, 호환성, 통일성 등을 갖춰야 함



1.2 사물인터넷 표준화

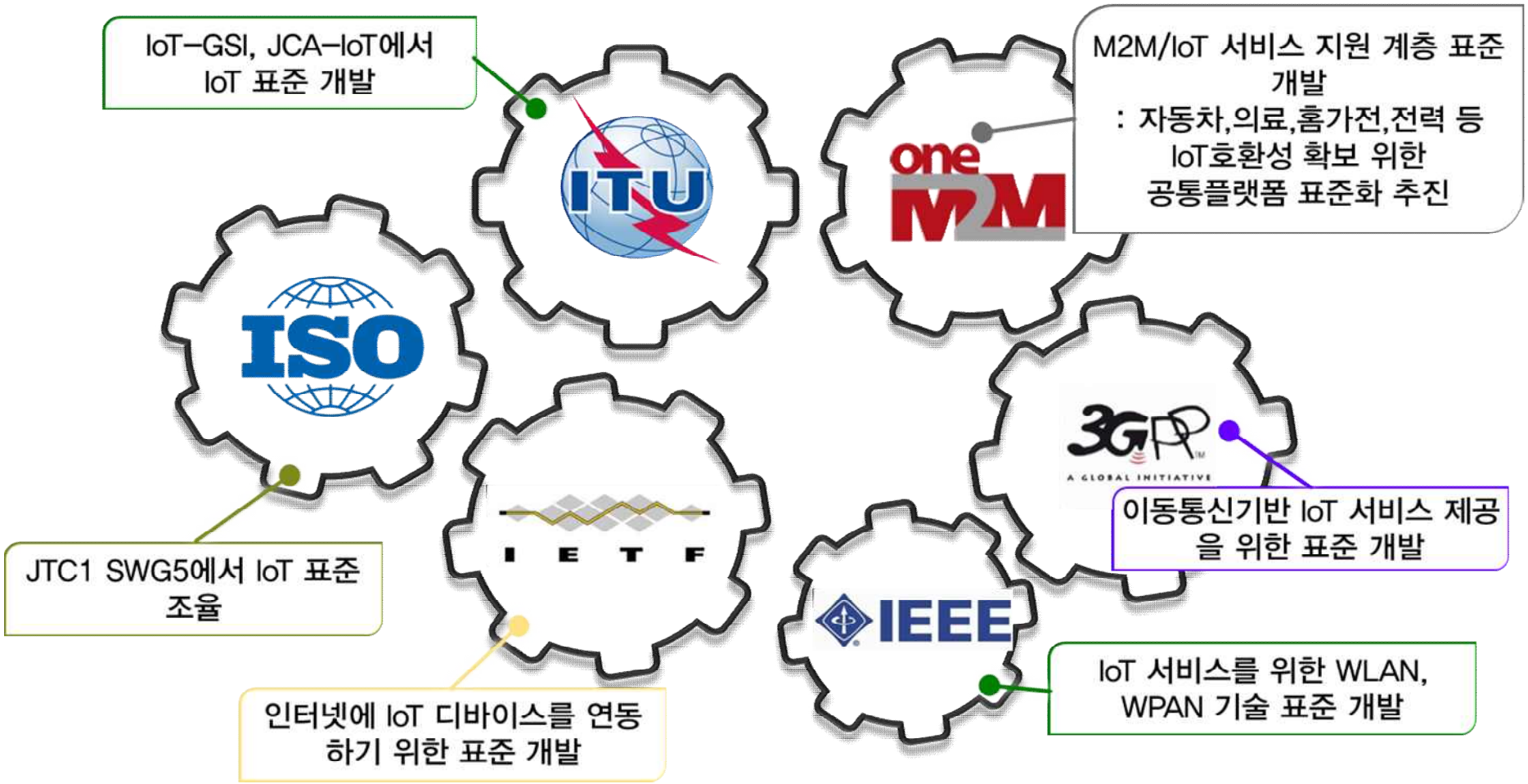
▪ 표준화 개념 및 표준화 기구 개요

– 표준의 구분

<table>
<tr><th>구분</th><th>내용</th><th>특징</th></tr>
<tr><td>공식적 표준
(De-jure Standard)</td><td>•공신력 있는 표준화 기구에서 일정한 절차와 심의를 거쳐 재정하는 표준
•표준화 기구
<table><tr><td>국제표준화기구</td><td>ISO, IEC, ITU</td></tr><tr><td>지역표준화기구</td><td>ETSI(유럽), APT(아시아태평양)</td></tr><tr><td>국가표준화기구</td><td>ANSI(미국), 기술표준원(한국)</td></tr><tr><td>단체표준화기구</td><td>TIA(미국), TTC, ARIB(일본), TTA(한국)</td></tr></table></td><td>•표준화 절차상 국제표준화 기구 간 수직관계가 형성되어 있으나, 최근 기술의 경계가 흐려짐에 따라 국제표준화 기구간 공동 표준화 활동이 활발이 이루어지고 있음
•국가나 지역의 표준화 활동 결과를 국제표준화 활동에 반영하거나(상향식),국제표준을 국가 표준화 활동과 산업체에 반영함.(하향식)
•국제표준 제정에 약 3년~6년이 소요됨.(표준화 느림)</td></tr>
<tr><td>사실상 표준
(De-facto standard)</td><td>•기업(제품)간 치열한 경쟁을 통해 시장에서 결정되는 시장표준
•1990년대 이래 약 100여 개가 생성 · 소멸 되었으며, 최근 사실상 표준의 확산을 위해 공식적 표준화 기구와 협력을 추진하고 있음</td><td>•사실상 표준은 시장원리에 따라 시장 지배기능을 가짐.
•특정기술과 이해관계가 있는 통신사업자, 방송업체, 제조 및 솔루션업체 등이 사실상 표준화에 참여함
•표준제정 속도가 빠르며, 사업화 우선의 표준화를 추진함</td></tr>
<tr><td>포럼/컨소시엄 표준
(Forum Specification)</td><td>•몇몇의 복수 기업이 자주적으로 결합해 포럼 또는 컨소시엄을 구성하여 제정한 표준
•사실상 표준과 경쟁하여 지배적 표준이 될 경우, 사실상 표준이 되거나 공식적 표준으로 제정되기도 하여 잠정적인 표준이라 할 수 있음</td><td>•사실상 표준에서 낙오된 기업들이 연합해 승자인 선두기업에 대항하는 수단으로 이용되는 경우도 있음
•아직 어느 표준이 시장을 지배하지 못한 경우, 동일 분야에서 복수의 포럼이 서로 패권을 경쟁하기도 함</td></tr>
</table>

1.2 사물인터넷 표준화

- 사물인터넷 표준화 기구 개요

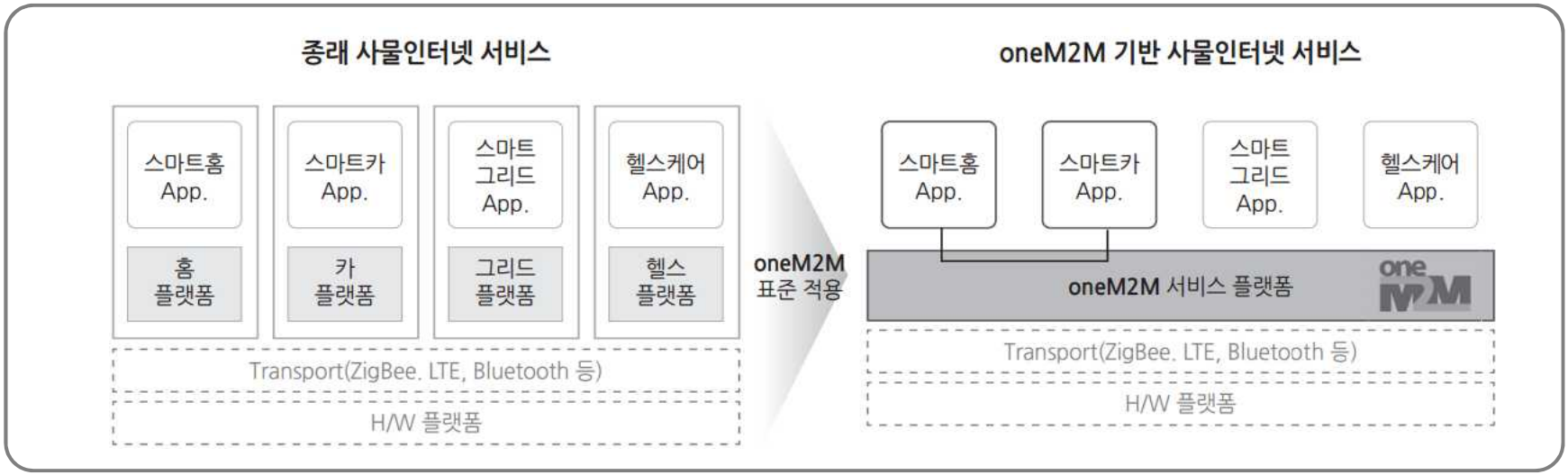


1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

① oneM2M

- oneM2M은 에너지, 교통, 국방, 공공서비스 등 산업별로 종속적이고 폐쇄적으로 운영되는, 파편화된 서비스 플랫폼 개발 구조를 벗어나 응용서비스 인프라(플랫폼) 환경을 통합하고 공유하기 위한 사물인터넷 공동서비스 플랫폼 개발을 위해 발족된 사실상 표준화 단체임
- 전세계 지역별 표준 개발기구인 TTA(한국), ETSI(유럽), ATIS/TIA(북미),CCSA(중국), ARIB/TTC(일본)등 7개의 SDO(Standard Development Organization)가 공동으로 설립함



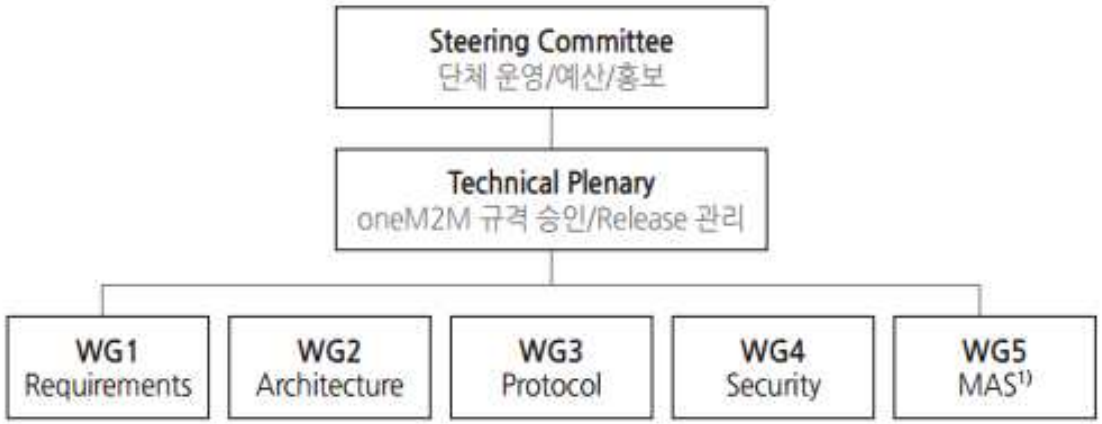
※출처 : oneM2M 사물인터넷 서비스 플랫폼 표준화 현황(김기형 LG전자, 2014)

1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

① oneM2M

- oneM2M의 기술 워킹그룹(6개)은 요구사항을 다루는 Requirement(WG1), 시스템 구조를 다루는 Architecture(WG2), 프로토콜과 관련한 Protocols(WG3), 보안관련 Security(WG4), 장치관리 및 추상화, 시멘틱과 관련된 Management, Abstraction and Semantics(WG5), 테스킹 규격을 위한 Test(WG6)로 구성되어 있음
- oneM2M은 2015년 1월 요구사항, 용어정의, 아키텍처 등 10개의 표준규격을 포함하는 1차 규격(Release 1)을 발표하고, 표준확산에 주력하고 있음



1) Management, Abstraction and Semantics

1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

① oneM2M

– oneM2M Release 1

- oneM2M 플랫폼이 제공하는 기능을 공통 서비스 기능(CSF, Common Services Function)으로 정의함

- 공통 기능은 사물인터넷 서비스 애플리케이션에서 자주 사용되는 기능을 정의한 것으로 데이터 저장/공유, 장치 관리, 그룹 관리, 구독/통지(Subscription/Notification), 위치 정보, 과금 등의 기능을 포함하며, 보안 기능은 기본적인 인증, 접근 제어 등의 기능을 제공

- 또한, oneM2M 코어 프로토콜 메시지(Primitive)는 CoAP, HTTP 및 MQTT 프로토콜 메시지를 통해 전송됨. oneM2M의 코어 프로토콜은 향후 추가 프로토콜 바인딩(Binding)을 지원할 수 있도록 특정 메시지 프로토콜에 종속성을 가지지 않도록 개발되었음

1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

① oneM2M

– oneM2M Release 2

- 다양한 인더스트리 사물인터넷 플랫폼 및 네트워크 연동이 주 목적임
- 사물인터넷 연동으로는 AllJoyn, OCF(Open Connectivity Foundation) 및 Lightweight M2M 기술과의 연동 규격을 제공함
- 네트워크 연동으로는 3GPP Rel-13 네트워크와 연동을 위한 트래픽 패턴 설정(Traffic Pattern Configuration) 기능을 정의하고 있으며 릴리즈 3에 모니터링 등의 연동 기능을 추가하기 위한 기술 보고서를 작업을 지속하고 있음
- 높은 디바이스 및 애플리케이션의 호환성을 보장하기 위해 우선적으로 가전 디바이스에 대한 데이터 모델을 정의함
  - ✓ 릴리즈 1에서는 가전 제어 및 센싱 정보를 교환하기 위해 사전에 애플리케이션 간 정의한 데이터 모델을 container 및 contentInstance 자원 타입을 이용했다. 이에 비해 릴리즈 2에서는 oneM2M 플랫폼을 이용하는 모든 애플리케이션이 표준에 정의된 가전 디바이스 데이터 모델을 사용함으로써 가전 제조사 및 애플리케이션 개발자 간에 별도의 데이터 모델을 정의하는 번거로움을 없애고 제품과 애플리케이션 간의 호환성을 보장한다.
- 프로토콜 바인딩은 동시 송수신(Full-duplex)을 지원하는 WebSocket이 추가됨

1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

② OCF

- OCF(Open Connectivity Foundation)는 사물인터넷을 구현 시 REST 구조 기반으로 경량형 CoAP 프로토콜로 사물인터넷 장치들을 연결하고 장치에 존재하는 자원들을 상호제어 할 수 있게 하는 표준 플랫폼 기술
- 2014년 7월 OCF(Open Interconnect Consortium)가 삼성, 인텔 등을 중심으로 시작해서 2015년 12월 스마트홈의 대표적 국제표준단체인 UPnP포럼을 통합 흡수하면서 회원사가 100개 이상으로 성장하였고 2016년 2월에 마이크로소프트, 퀄컴 등이 합류하여 기업 표준화 단체가 됨

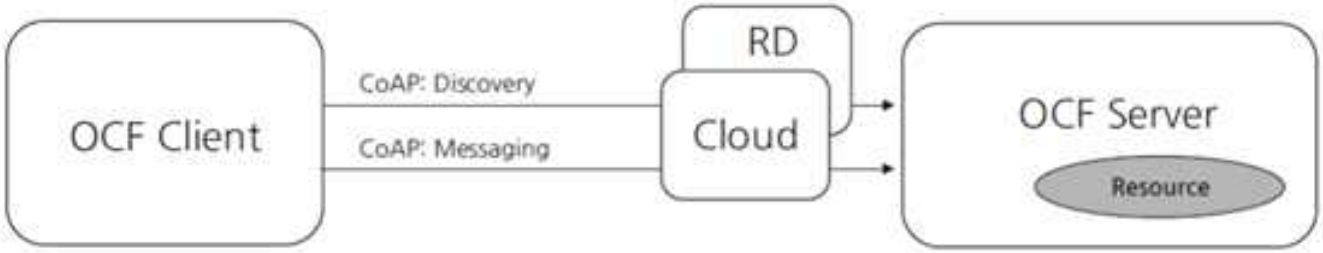
1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

② OCF

◆ 표준 기술 구성

- 다양한 사물인터넷 유무선 연결기술을 활용하여 논리적인 상호연동성을 보장하는 아키텍처를 구축하여 스마트홈, 자동차, 물류, 헬스케어 등 다양한 사물인터넷 서비스(Profiles)를 개발할 수 있도록 구성됨
- OCF 아키텍처는 클라이언트-서버의 방식으로 RESTful 아키텍처를 기반으로 리소스를 관리하는 모델
- 사물인터넷 디바이스의 제한된 성능을 고려하여 CoAP(Constrained Application Protocol)을 활용하여 경량 기기에서의 동작도 고려함



1.2 사물인터넷 표준화

※ REST(Representational State Transfer)란 ?

- World Wide Web과 같은 분산 hyper media system을 위한 software architecture의 한 형식
- Resource를 정의하고 resource에 대한 주소를 지정하는 방법 전반을 일컫는 말
- 간단한 의미로는, 웹 상의 자료를 HTTP위에서 SOAP나 cookie를 통한 session tracking같은 별도의 전송 계층 없이 전송하기 위한 아주 간단한 interface 임
- 이러한 REST원리를 따르는 system을 RESTful system이라고 지칭함

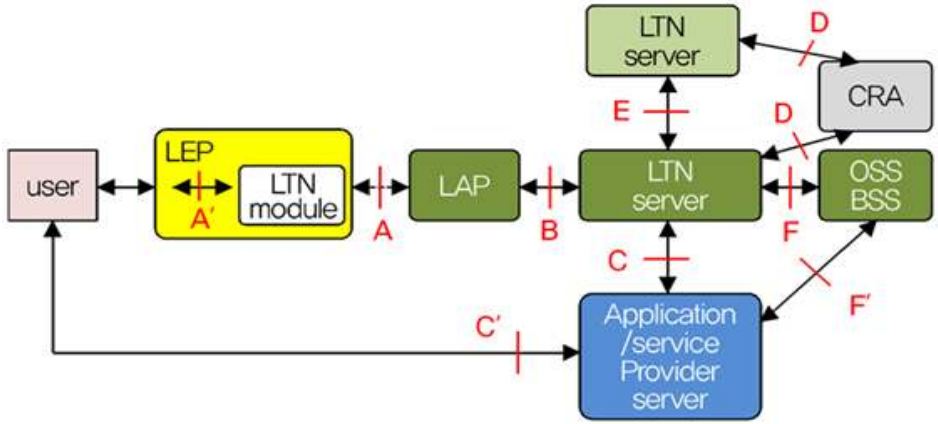
※ CoAP(Constrained Application Protocol)이란?

- Internet에서 IoT device처럼 제한된 computing 성능을 갖는 device들의 통신을 위해 IETF의 CoRE(Constrained RESTful Environment) working-group에서 표준화한 protocol
- 신뢰성 있는 동기 수송 방식의 TCP와 그 위의 HTTP는 많은 resource제약을 가진 IoT 환경에서는 적합하지 않아 비동기 수송 방식의 UDP상에서 UDP의 단점을 보완하는 개념을 포함한 통신 protocol

1.2 사물인터넷 표준화

- 사물인터넷 표준화 기구

③ 비면허 대역(LPWA) 광역 IoT 표준화



LTN END Point(LEP)
LTN Access Point(LAP)
Operation Support System(OSS)
Business Support System(BSS)
Central Registration Authority(CRA)

[그림] ETSI의 Low Throughput Network(LTN) 망 및 인터페이스

- 2014년 9월에 완료되었으며, LTN 001 유스케이스, LTN 002 기능구조, LTN 003 프로토콜 및 인터페이스 표준으로 구성됨
- 이 표준에는 크게 대역확산에 의한 협대역 펄스에 의한 Ultra Narrow Band(UNB) 표준과 Direct Sequence Spread Spectrum(DSSS) 표준이 있음
- SigFox와 LoRa의 멤버가 의장과 부의장을 맡아서 작성된 표준으로서 이들 표준에 준하여 작성된 대표적인 표준은 2014년 9월에 완성된 UNB기반의 SigFox라는 전용(propriety) 표준과 IEEE 802.15.4g기반의 DSSS방식으로 LoRa Alliance에서 2015년 6월에 완성된 개방형 LoRa 표준이 있음
- LoRa는 2015년 초에 결성된 IBM, Semtech, Actility, Microchip 등을 멤버로 구성된 LoRa 얼라이언스에 2015년 6월 16일에 발표한 LoRaWAN R1.0개방형 표준이고, IEEE 802.15.4g 기반의 표준이며 비동기식 저전력 원거리 통신망 임

1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

③ 비면허 대역(LPWA) 광역 IoT 표준화

구분	LTE-M(Cat.1)	NB-IoT(Cat.M2)	LoRa
표준화	3GPP Rel.8(표준화 완료)	3GPP Rel.13('16.6월 표준화 완료)	비표준
상용	'16년 상반기(상용화 완료)	'17년 4월(상용화 예정)	' 16년 하반기
속도	(DN) 10Mbps/(UP) 5Mbps	(DN) 200kbps/(UP) 144kbps	(DN) 5.5kbps/(UP) 300bps~5.5kbps
양방향	불가	eDRX*지원을 통한 양방향 송신	불가
주파수 대역(QoS 보장)	면허대역(LTE주파수)	면허대역(LTE주파수)	비면허대역(917~923.5MHz)
Battery Life	PSM 적용시 10년	PSM 적용시 10년	10년
Cell 커버리지	1~2km	15km	11km
커버리지	전국망 (LTE 커버리지)	전국망 (LTE커버리지 * 1.2)	특정지역 서비스 (16,000식, ' 16년말 기준)
서비스 성공률(품질)	99% 이상		26%~88%(거리 기준)
단말 생태계	칩셋/모듈 다양한 생태계 조성		모듈 SemTech 중심 소수 업체
보안	LTE 3단계 보안성 제공 가입자 인증, NAS Security (UE~MME 무결성 체크 및 암호화), AS Security (UE~eNB 무결성 체크 및 암호화)		망 인증 없는 단말간 통신으로 단말제어 불가
글로벌 로밍	글로벌 표준으로 기술진화 용이, 글로벌 로밍 수용		LoRa Alliance 대상한정 로밍 수용
이동성	저속/초고속 모두 지원		고정중심, 보행 이동자 지원
프로토콜	TCP-IP(HTTP)	UDP(Coap)	–
통화기능(핸드오버)	지원	불가	불가

1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

④ 셀룰러 기반 광역 IoT 표준화

- 셀룰러 기반의 광역 IoT 기술에 대한 표준은 3GPP의 LTE-M 표준의 진화로부터 살펴볼 수 있음
  - Rel.12 : M2M/IoT를 위한 LTE 표준의 개발은 가격과 전력 소모에 중심 개발
  - Rel.13 : LTE-M이 2015년 9월 Radio Access Network(RAN) 69차 미팅에서 WI로 선정되어(RP-151621) 통신거리와 저가 구현에 맞춘 표준화 진행 중
    - ✓ 대역폭 200KHz 기반의 협대역(narrowband) Narrow Band IoT(NB-IoT)와 1.4MHz 대역의 LTE-M 표준 개발
    - ✓ 두 표준 모두 전송속도를 유연하게 가변으로 조정할 수 있다. NB-IoT는 GSM EDGE Radio Access Network(GERAN)에서 하웨이/Nuel 및 퀄컴 주도로 진행돼 오던 표준이 상향링크는 하웨이/Neul의 UL M2M, 하향링크는 퀄컴의 DL OFDM으로 단일 통합된 NB-Cellular IoT(CIoT)표준(초기에는 Cleanstate IoT로 불림)이 GERAN에서 진행되다가, 표준화 창구가 3GPP RAN으로 통합되어, RAN 69차 회의에서 NB-IoT라는 이름의 표준에 통합되어 WI로 진행되게 됨
- NB-IoT는 3개의 동작모드 지원 (2016년 표준화)
  - 단독 동작(stand alone operation) : 기존 GSM을 대치하는 모드
  - 보호대역 동작(guard band operation) : LTE 캐리어의 보호대역 내의 미사용 RB들을 이용하는 모드
  - 대역 내 동작(in-band operation) : 보통 LTE캐리어 내의 RB들을 이용하는 모드

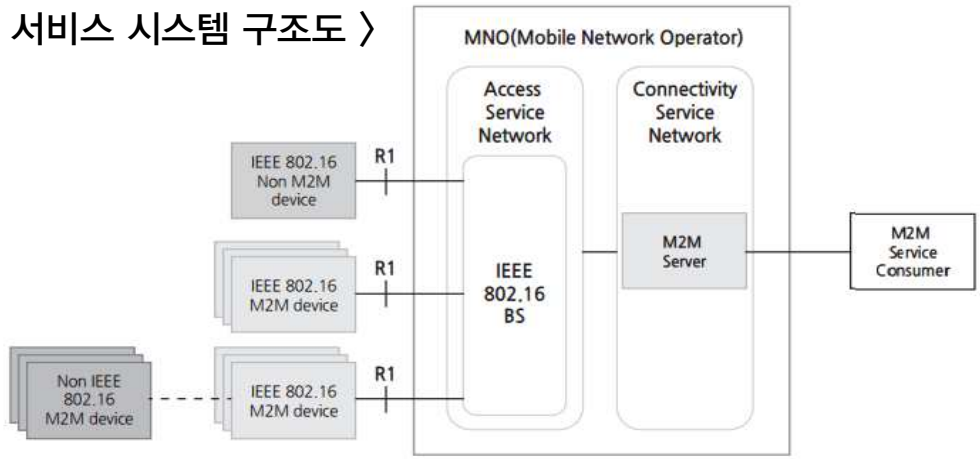
1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

⑤ IEEE(전기전자공학자협회)

- IEEE는 1980년에 대학과 기업이 함께 발족한 단체로, 데이터 통신부분에서 물리계층 및 링크계층 표준을 규정하는 표준화 기구이며, 사물인터넷 관련 표준화는 IEEE Standard Association(IEEE-SA)에서 이루어지고 있음
- IEEE는 2014년 7월 IEEE P2413프로젝트그룹을 결성하여 IoT/M2M 전반적인 프로토콜, 아키텍처구조 등에 대해 표준 개발 작업에 착수하였으며, oneM2M과 협력하고 있음

〈 IEEE 802.16p 기본 M2M 서비스 시스템 구조도 〉



※출처 : 'IEEE에서의 사물인터넷 기술 표준화 현황'(2014년 송재승 세종대학교 정보보호학과 교수

1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

⑥ 3GPP(3rd Generation Partnership Project)

- 이동통신과 관련된 사실상 표준을 제정하고 있으며, oneM2M과 마찬가지로 7개의 SDO(Standard Development Organization)들 간의 합의에 의해서 결성되고 표준을 개발해 온 표준 단체
- 3GPP에서는 사람의 개입이 꼭 필요하지 않은 하나 혹은 그 이상의 객체가 관여하는 데이터 통신 기술을 M2M 또는 MTC(Machine Type Communication)라 정의하고, 이러한 디바이스에 필요한 이동통신 네트워크 중심 기술 표준을 진행하고 있음
- MTC에서는 기존에 디바이스들이 네트워크를 통해 어플리케이션 서버에 접속하는 것을 시작으로 응용이 수행되는 것과 달리, 어플리케이션 서버가 먼저 MTC 기기를 triggering 하여 응용의 시작 및 정보의 수집 등을 요구할 수 있는 통신 모델 요구사항을 만족시키기 위해 triggering 요구의 중계를 위한 네트워크 노드 추가, 프로토콜 정의, MTC 디바이스의 주소 및 식별자 정의에 대해 표준화를 진행

1.2 사물인터넷 표준화

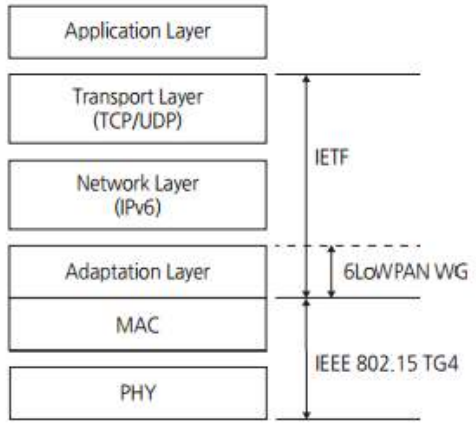
▪ 사물인터넷 표준화 기구

⑦ IETF(국제인터넷표준화기구)

- IETF(Internet Engineering Task Force)는 인터넷의 운영, 관리, 개발에 대해 협의 하고 프로토콜 표준을 개발하고 있으며, 사물인터넷의 다양한 인터넷 프로토콜들에 대한 표준을 개발하고 있음
- IEEE에서는 IPv6 기반의 저전력 무선 네트워크에 대한 표준을 6LoWPAN을 통해 추진하였으며, 6LoWPAN의 상위 어플리케이션 계층 프로토콜의 표준화를 CoAP에서 추진하였음
- 최근, 2014년 7월 IEEE P2413프로젝트그룹을 결성하여 IoT/M2M 전반적인 프로토콜, 아키텍처 구조 등에 대해 표준 개발 작업에 착수함

구분	표준화 대상 및 현황
6LoWPAN	• IEEE 802.15.4, Bluetooth, Z-wave, NFC 기반의 무선기술도 고려한 IPv6 전송에 대한 표준개발이 진행되고 있음
CoAP	• CoAP의 표준화 영역은 전송계층으로 UDP를 고려하고, 상위 어플리케이션 계층에서 디바이스간 서버/클라이언트 방식으로 리소스 이벤트에 대한 전송방법을 RESTful기반의 프로토콜을 설계함

〈 6LoWPAN 기술 범위 〉



1.2 사물인터넷 표준화

▪ 사물인터넷 표준화 기구

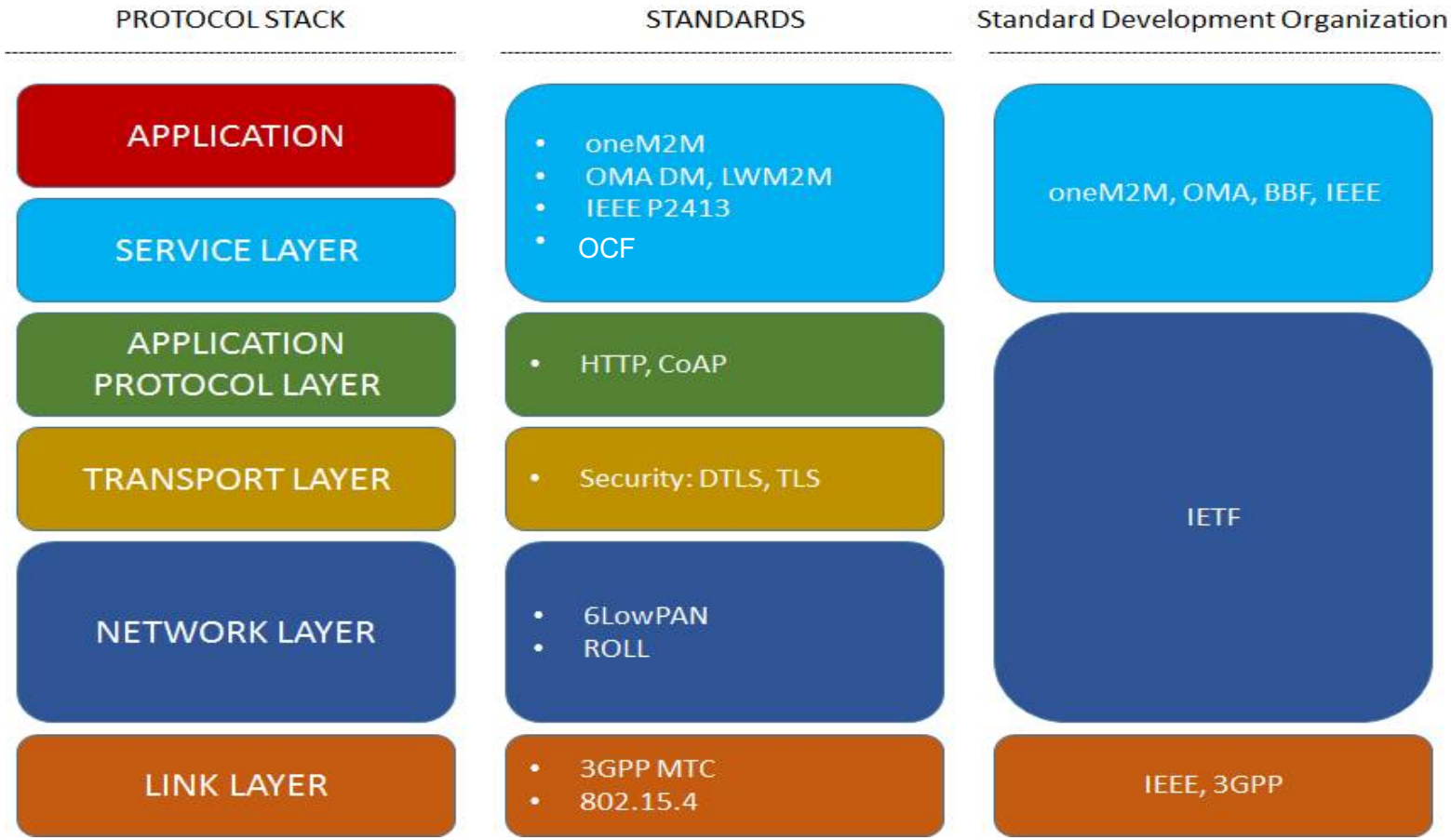
⑧ 기타 주요 표준화 단체

구분	개요	표준화 대상 및 현황
ISO/IEC JTC1	• ISO/ISE JTC1은 정보처리시스템에 대한 국제표준화 위원회(ISO/TC97)와 정보기기에 대한 국제표준화 위원회(IEC/TC83)를 통합하여 공동기술 위원회를 설립함(1987년) • ISO/IEC JTC1은 사물인터넷 특별워킹그룹5(SWG5: Special Working Group on Internet of Things)를 2012년 설립함	• SWG5는 사물인터넷을 "사물,사람, 시스템 및 정보자원이 서로 지능형 서비스로 연결되어 실 세계 및 가상계의 정보를 처리하고 그에 따라 반응이 가능한 기반구조"로 정의함 • 2014년 사물인터넷 표준화를 위한 워킹그룹10(WG on IoT)을 설립하고, SWG5에서 진행된 사물인터넷 참조구조 표준개발을 담당함
ITU-T	• ITU-T(International Telecommunication Union Telecommunication Standardization Sector)는 국제 전기 통신 연합 부문의 하나로 통신 분야의 표준을 개발하며, 1956년에 설립됨 • 2011년 부터 JCA-IoT 및 IoT-GSI를 구성하여 사물인터넷 관련 표준화 활동의 조율 및 표준화 로드맵 작성 및 표준화 계획 수립/관리를 추진함	• 2012년 사물인터넷 표준인 Y-2060 : Overview of the Internet of things 를 개발함 • 사물인터넷을 ICT를 기반으로 한 물리적 및 가상의 사물들을 연결하는 글로벌 서비스 인프라로 정의하고, 응용/서비스 및 응용지원/네트워크/디바이스의 4개 계층과 각 계층에 적용되는 관리 및 보안 기능으로 구성된 사물인터넷 참조모델을 표준화 함
Thread Group	• 구글이 주도하고 네스트랩스, 실리콘랩스, 프리스케일, ARM, 예일 시큐리티, 삼성전자가 참여하여 사물인터넷을 위한 새로운 IP기반 무선통신망 프로토콜 개발을 통해 상호호환이 가능한 사물인터넷을 구현하기 위해 설립된 컨소시엄 표준화 단체로, 2014년 1월 설립됨	• 스레드란 이름은 저 전력 기반의 IEEE 802.15.4 메시지 네트워크를 위해 설계된 IPv6 네트워킹 프로토콜을 의미하며, 저 전력 무선 프로토콜인 6LoWPAN사용을 통해 저전력으로 가정용 디바이스간 연결을 제공하고 있음

1.2 사물인터넷 표준화

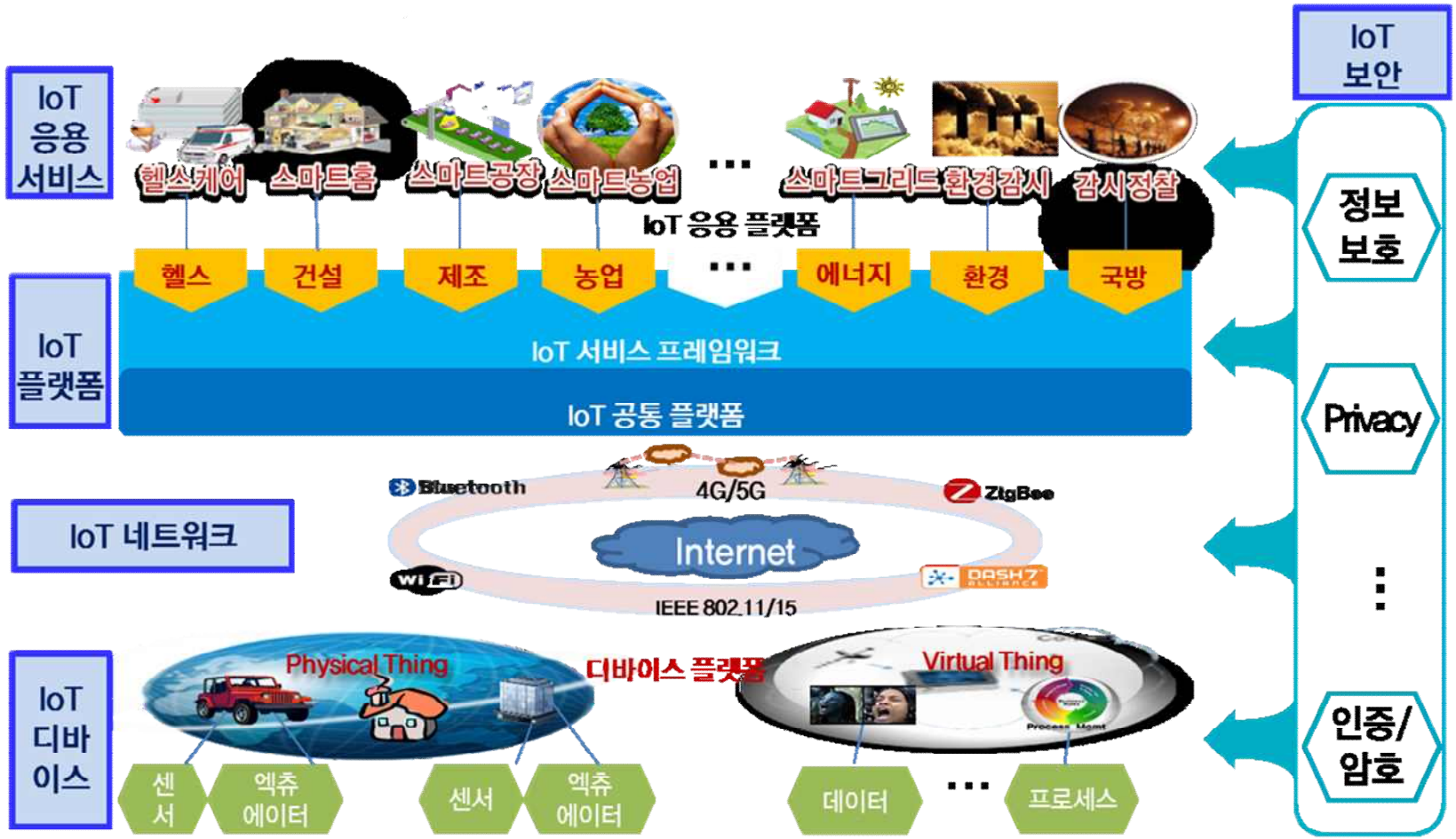
- 사물인터넷 표준화 기구

⑨ 사물인터넷 표준화 기구의 표준화 영역



1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 개요



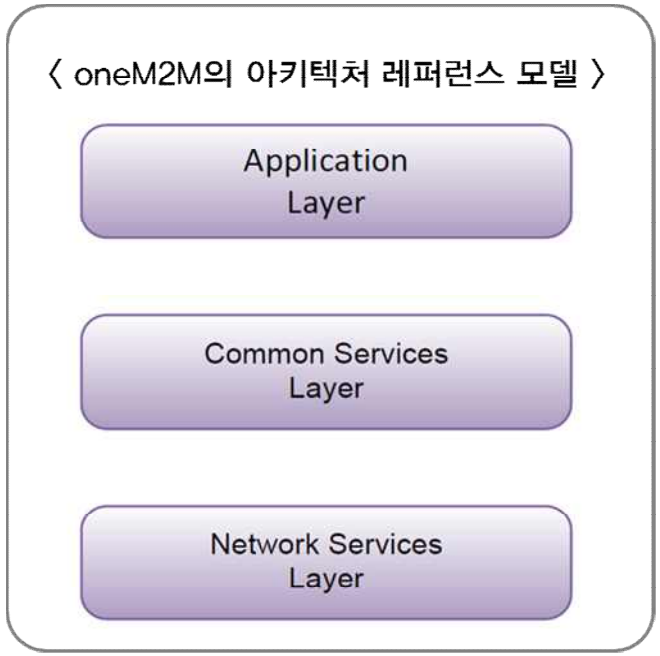
1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 레퍼런스 모델

① oneM2M Architecture Reference Model

- oneM2M 워킹그룹 2에서는 oneM2M의 네트워크 아키텍처와 아키텍처를 구성하는 엔티티(Entity) 및 공통서비스기능(CSF: Common Service Function)과 이를 제공하기 위한 공통 서비스 계층에 서의 레퍼런스 포인트(Reference Point)를 정의함
- oneM2M에서 지원하는 네트워크 아키텍처는 애플리케이션 전용 노드(ADN: Application Dedicated Node), 애플리케이션 서비스 노드(ASN: Application Service Node), 중간노드(MN: Middle Node) 및 인프라스트럭처 노드(IN: Infrastructure Node)로 구성됨(Release 1 규격 TS 0001(2015.2월))

〈 oneM2M의 아키텍처 레퍼런스 모델 〉



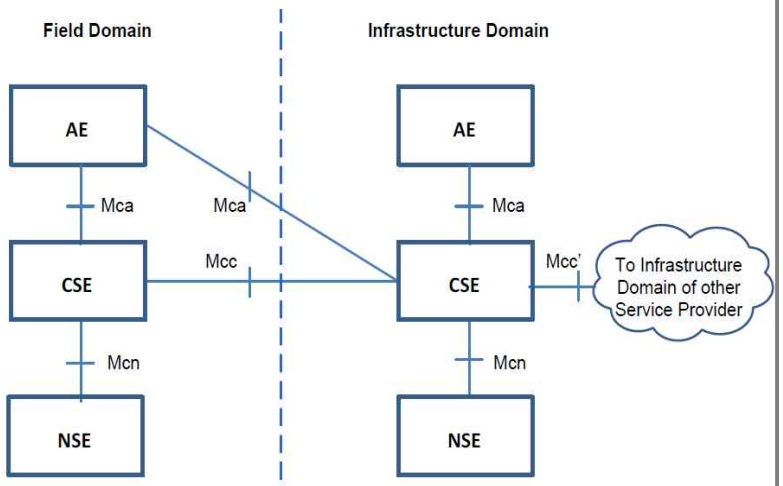
1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 레퍼런스 모델

① oneM2M Architecture Reference Model

- oneM2M 레퍼런스 아키텍처의 모든 엔티티(AE, CSE, NSE)는 세가지 계층으로 분화되며, 각 엔티티의 기능은 다음과 같이 정의함

〈 oneM2M Functional Architecture 〉



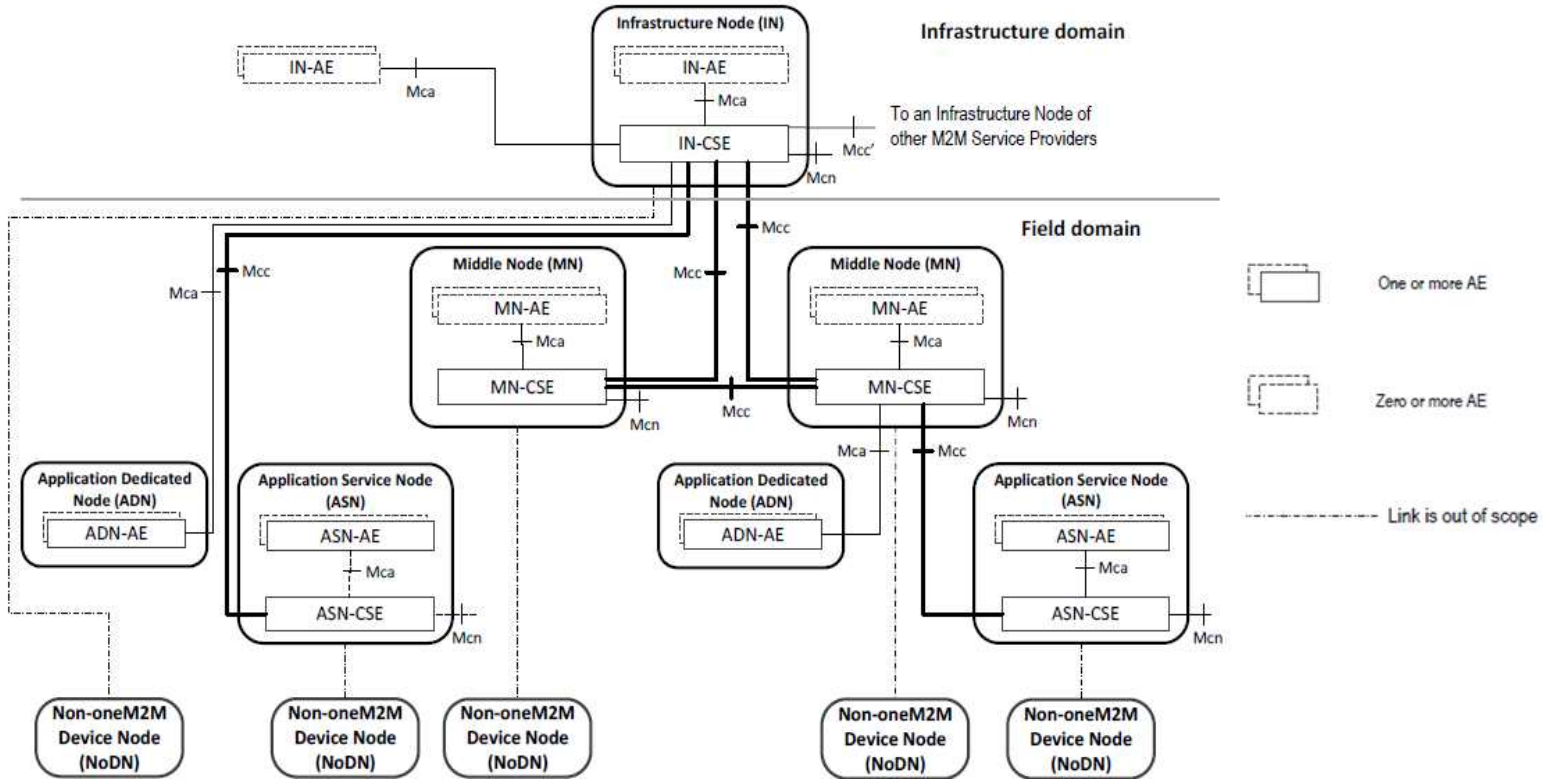
구분	내용
애플리케이션 엔티티 (Application Entity)	• 애플리케이션 엔티티는 End-to-End 사물인터넷 솔루션을 위한 애플리케이션 로직을 제공함 – 화물추적,원격모니터링, 원격검침 및 제어 등
공통 서비스 엔티티 (Common Service Entity)	• 공통 서비스 엔티티는 사물인터넷의 다양한 애플리케이션 엔티티들이 공통적으로 사용 할 수 있는 기능들로 이루어진 플랫폼임
네트워크 서비스 엔티티 (Network service Entity)	• 네트워크 서비스 엔티티는 공통 서비스 엔티티에 네트워크 서비스를 제공함. 3GGp 네트워크 연동 중심임 – 장치관리, 위치서비스, 장치 트리거링 등

1.3 사물인터넷 플랫폼 아키텍처

- 사물인터넷 아키텍처 레퍼런스 모델

① oneM2M Architecture Reference Model

〈 oneM2M 아키텍처에 의한 M2M 시스템 전체 구성 〉



1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 레퍼런스 모델

① oneM2M Architecture Reference Model

- **애플리케이션 서비스 노드(ASN, Application Service Node)**
  – M2M Application 뿐만 아니라 공통의 서비스 기능을 포함하는 일반노드
- **애플리케이션 전용 노드(AND, Application Dedicated Node)**
  – M2M Application을 포함하는 M2M 디바이스로 M2M 서비스 로직만을 포함하는 제한된 기능을 가지는 제한적 디바이스
- **중간노드(MN, Middle Node)**
  – 디바이스 노드들과 네트워크 인프라스트럭처를 연결해주는 게이트웨이 역할을 하는 노드
- **인프라스트럭처 노드(IN, Infrastructure Node)**
  – 네트워크 인프라스트럭처에 위치해 M2M 서비스를 제공해주는 노드
- **NoDN(Non-oneM2M Device Node)**
  – NoDN 은 oneM2M 엔티티를 AE도 CSE도 가지지 않는 Node이며, 관리 등을 포함하여 상호 연동할 목적을 위해 oneM2M 시스템에 붙어있는 노드를 나타냄

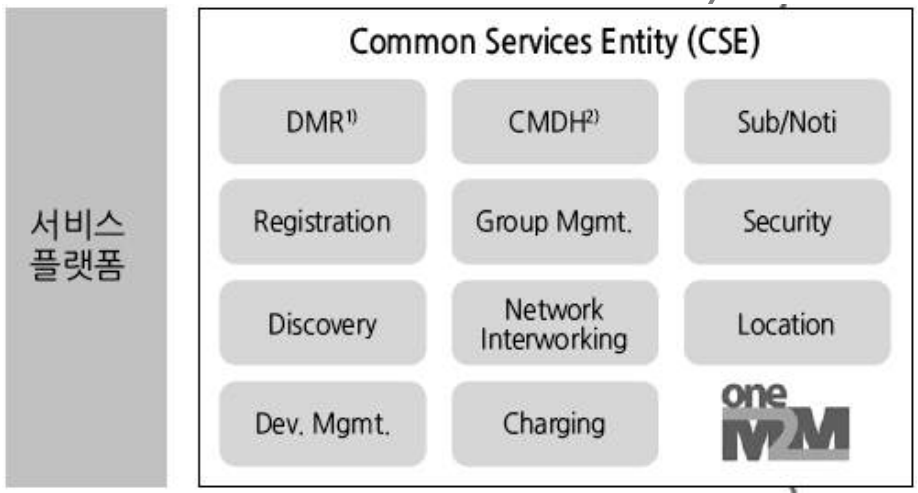
※ oneM2M 시스템에서 개별적 식별 가능한 논리적 엔티티를 Node라 함

1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 레퍼런스 모델

- 공통 서비스 엔티티에 포함되는 다양한 공통 서비스 기능들은 아래와 같으며, 이러한 기능들은 자원(Resource)을 통해 Mcc, Mca 및 Mcn 레퍼런스 포인트들을 통해 노출(Exposed)됨

〈 oneM2M 공통 서비스 기능 〉



1) Data Management and Repository,
2) Communication Management and Delivery Handling,
3) Quality of Service, 4) Device Management

Common Services Function (CSF)

DMR[1]	데이터 저장 및 관리. 데이터 분석 기능
CMDH[2]	데이터 저장 및 관리. 데이터 분석 기능
Subscription/ Notification	메시지 전달 관리 및 정책에 기반한 전송 QoS[3] 제어
Registration	정보 변경에 대한 구독/통지 기능
Group Mgmt.	플랫폼에 어플리케이션 및 장치 등록
Security	단대단 보안 연결 제공, 인증/권한 설정 기능
Discovery	특정 정보 탐색, 특정 정보에 대한 통지
Network Interworking	액세스 네트워크 (3GPP) 연동 기술
Location	장치에 위치 정보 제공 및 관리
Dev. Mgmt.	OMA DM[4], OMA Lightweight M2M, BBF TR-069 연동을 통한 장치관리 기능 제공
Charging	서비스 계층 과금

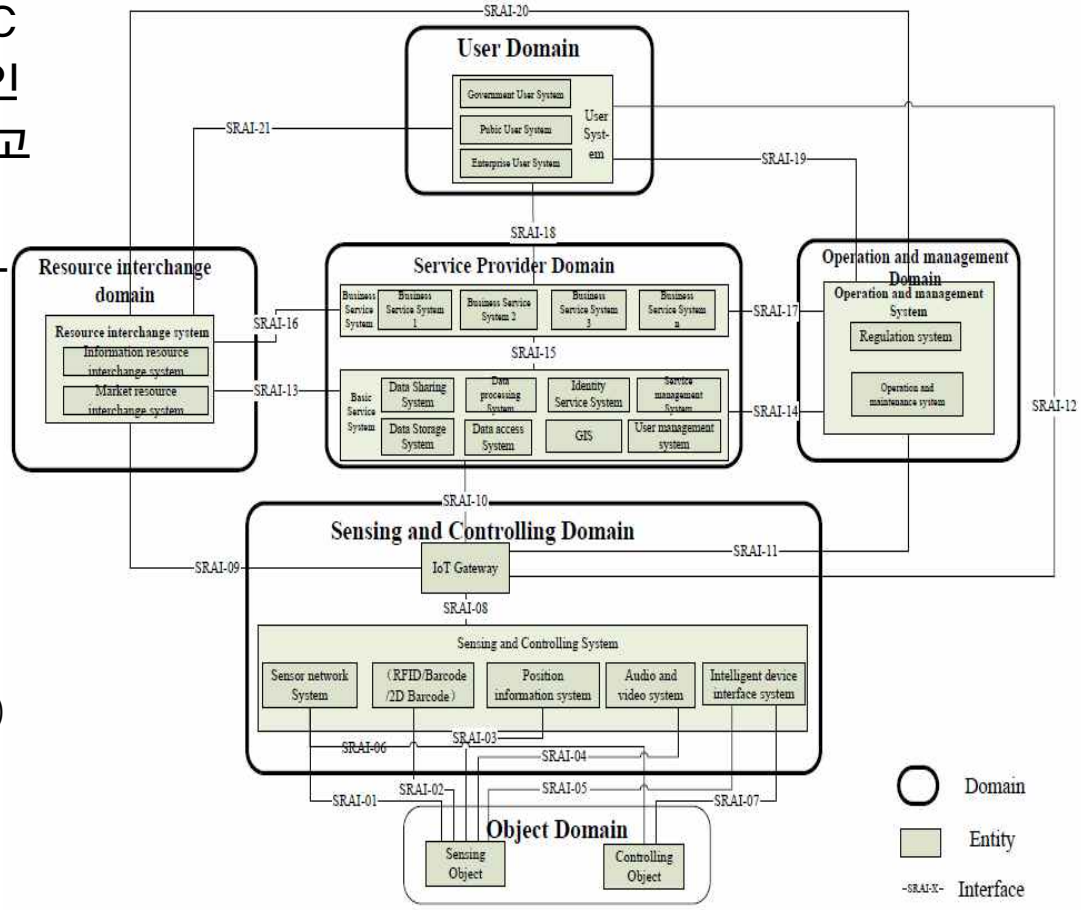
※출처 : oneM2M 사물인터넷 서비스 플랫폼 표준화 현황(김기형 LG전자, 2014)

1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 레퍼런스 모델

② ISO/IEC JTC1 IoT RA : ISO/IEC WD 30141

- IoT RA(Reference Architecture) 표준인 ISO/IEC 30141은 JTC1 총회(2014.11월)에서 사물인터넷을 위해 신설된 WG10에서 추진되고 있음
- oneM2M이 두 개의 영역으로 M2M/IoT 시스템을 정의한 것과 달리, ISO/IEC는 여섯 개의 영역(domain)으로 정의함
  - User Domain(UrD)
  - Object Domain(ObD)
  - Sensing & Actuation Domain(SAD)
  - Service Provider Domain(SPD)
  - Operation & Management Domain(OMD)
  - Resource Interchange Domain(RID)



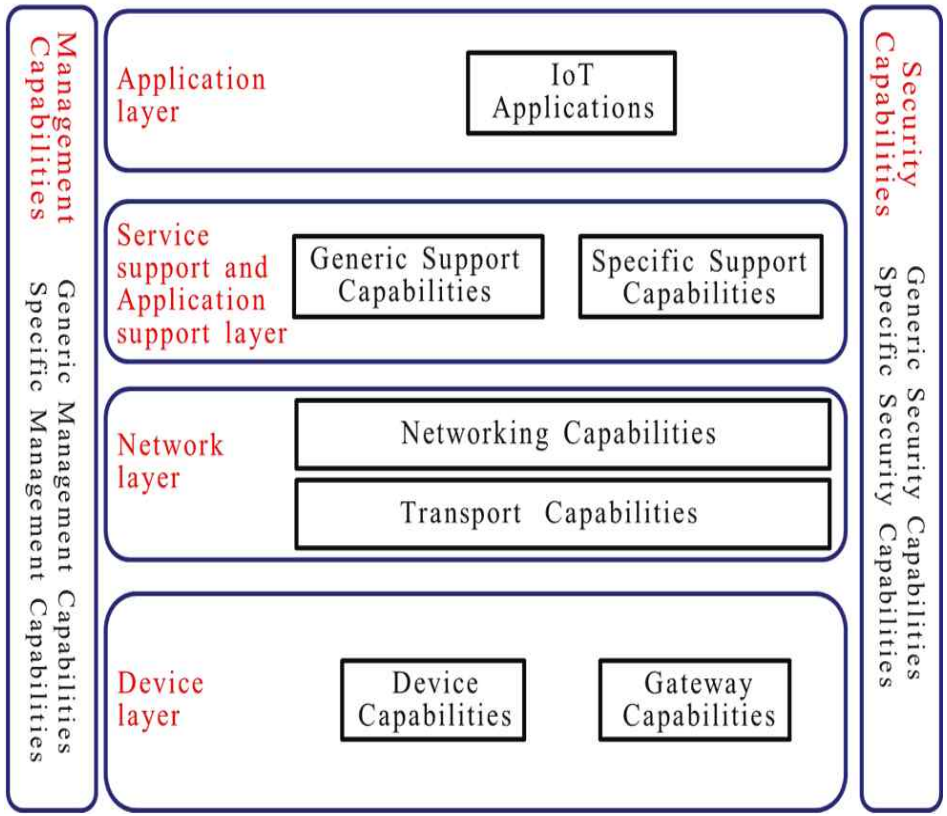
1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 레퍼런스 모델

③ ITU-T Reference Model Y.2060

- ITU-T는 2012년 공적 표준화 단체 중 처음으로 'Overview of the Internet of things'라는 제목의 아키텍처 모델 표준(Y.2060)을 제정함
- 사물인터넷을 '현존하거나 진화하는 ICT 기술을 바탕으로 물리적이거나 가상의 사물을 연결하여 진일보된 서비스가 가능한 정보사회를 위한 글로벌한 인프라'로 정의함

〈 Y.2060에서의 IoT Reference Model 〉



1.3 사물인터넷 플랫폼 아키텍처

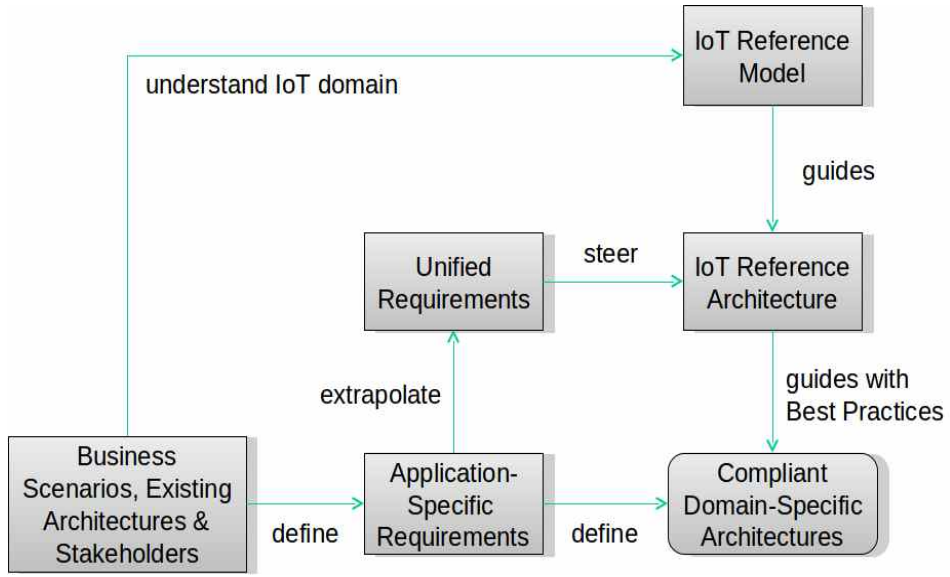
▪ 사물인터넷 아키텍처 레퍼런스 모델

④ FP7 IoT-A

- **IoT-A는 독일, 영국, 프랑스 등 8개국 17개 기관이 공동 진행한 유럽의 FP7의 단위 프로젝트임 (활동기간 : 2010.9월~2013.11월)**
  – 각각 별도로 개발되고 있는 이질적인 IoT 기술들을 통일성 있는 아키텍처로 통합하기 위한 아키텍처를 개발함

- **IoT-A는 RM(Reference Model)을 제시하고 이를 바탕으로 RA(Reference Architecture)를 구체화함**
  – 각각의 구성요소의 고려사항이나 전개 방향 등을 세분화하여, 특정 사물인터넷 시스템을 위한 아키텍처를 완성해 나가는 단계까지 상세하게 가이드하고 있음

〈 IoT 아키텍처 도출 흐름 〉



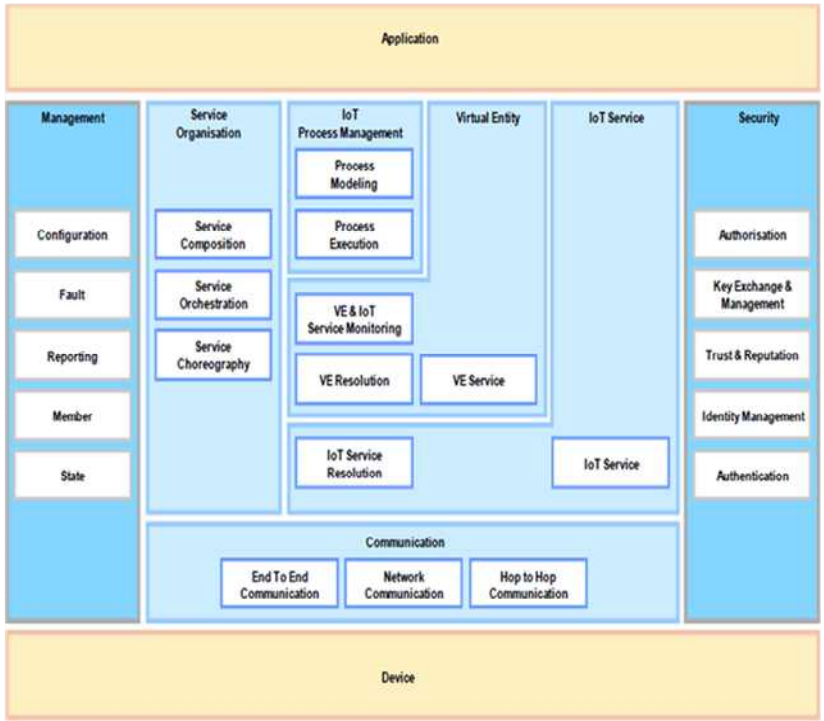
1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 레퍼런스 모델

④ FP7 IoT-A

IoT-A의 RA는 세부의 기능을 구체화하는 관점에서 응용과 디바이스 부분보다는 중간층의 엔티티들이 상세하게 세분화되어 있음

〈 IoT Reference Architecture 세부기능 〉



1.3 사물인터넷 플랫폼 아키텍처

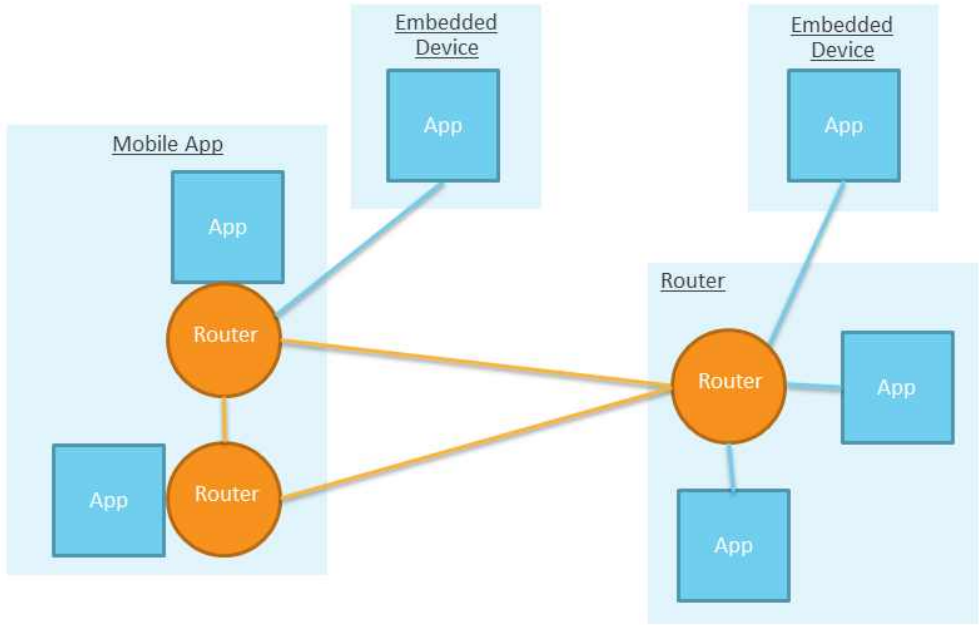
▪ 사물인터넷 아키텍처 적용

① AllSeen Alliance의 AllJoyn

- AllJoyn은 컬컴, 마이크로소프트, LG, 소니, 파나소닉, 샤프 등이 멤버로 참여하고 있으며, LG는 HDTV 등 AllJoyn을 채택한 상용제품이 출시함

- AllJoyn의 네트워크 아키텍처에서 '앱(App)'들은 '라우터(Router)'와 물리적 통신을 하며, '앱'들은 '라우터'를 통해서만 다른 '앱'들과 통신이 가능함. 이들은 한 물리적 디바이스 내부에 같이 있을 수도 있고 다른 디바이스에 있을 수도 있음

〈 AllJoyn의 Network Architecture 〉



1.3 사물인터넷 플랫폼 아키텍처

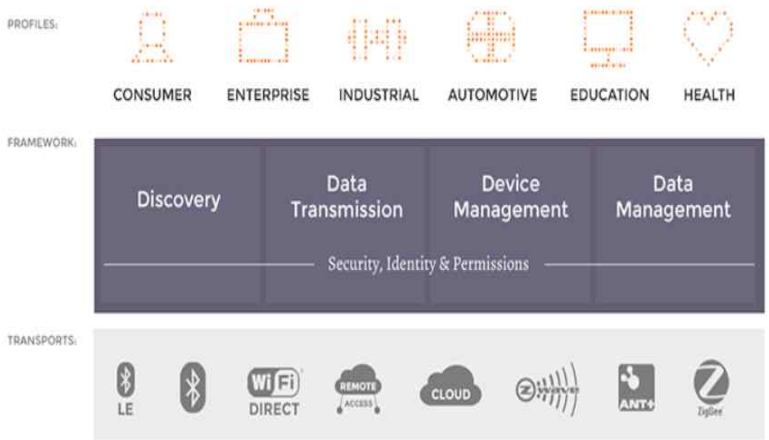
▪ 사물인터넷 아키텍처 적용

② OCF IoTivity

- **IoTvity는 오픈소스 커뮤니티의 자발적인 참여를 통해 수십 억 개의 IoT 기기를 연결하는 데 쓰이는 프레임워크 표준을 개발함**
  – IoTvity라고 불리는 RA는, 디바이스 제조자와 애플리케이션 제작자가 OCF 표준 호환 제품 및 서비스와 상호 운영되는 제품과 서비스를 제작할 수 있도록 하는 출발점으로 사용되어짐

- **IoTvity 아키텍처는 여러 기능들을 세분화하지 않고 일반화하여 AllJoyn에 비해서는 간단함**
  – 중앙에 표현된 프레임워크 빌딩 블럭은 oneM2M 기준으로 보면 CSE와 대응됨
  – 현재는 프레임워크에 4개의 기능 빌딩 블럭을 정의했는데, 이는 oneM2M의 CSE내부에 정의되어 있는 12가지 기능들 중에 4가지와 그대로 일대일 대응됨

〈 IoTivity의 프레임워크 API 〉

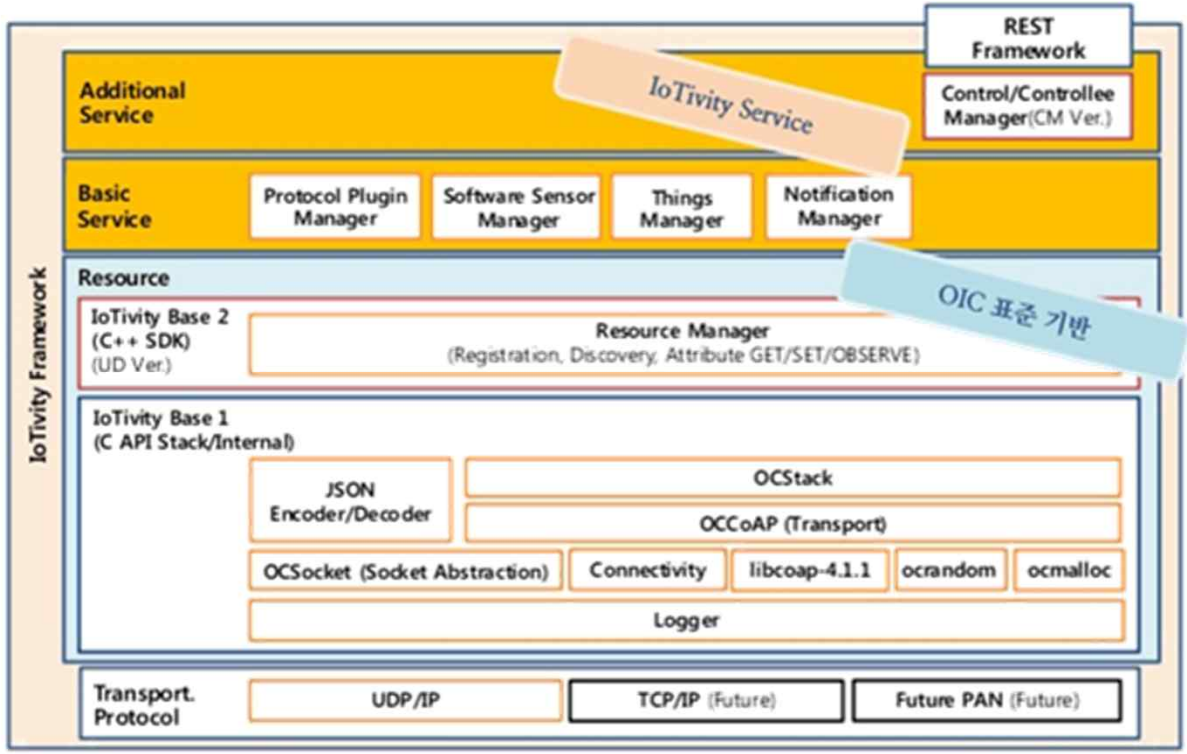


1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 적용

② OCF IoTivity

- IoTivity는 리소스 기반 RESTful 아키텍처 모델을 기반으로 하고, 따라서 모든 사물을 리소스로 표현하고 CRUDN(Create, Read, Update, Delete and Notify) 오퍼레이션을 제공한다.
- 데몬(Deamon)없이 CoAP(Constrained Application Protocol) 기반으로 설계되어 저사양, 저전력 기기 지원이 용이한 장점이 있다.
- IoTivity 프레임워크는 크게 IoTivity 서비스와 관련된 기본 서비스(Basic Service)블록과 추가 서비스 (Additional Service) 블록 그리고 OCF 표준 기반 구현 부분인 자원 (Resource) 블록으로 구성됨
- 자원 블록은 OCF 표준 기반한 부분으로 일반 리소스 모델, 리소스 발견, 메시징, 식별자 및 주소표현, CRUDN 오퍼레이션, 보안 등 IoTivity 프레임워크의 근간을 이루는 핵심적인 부분임



1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 적용

② OCF IoTivity

〈 IoTivity의 프레임워크 구성요소 〉

구분	내용
Common Solution	• 최종소비자, 회사, 산업계, 자동차 및 헬스분야 같은 여러 수직적인 상품시장들을 아우르고, OS, 플랫폼, 통신모드, 전송기술 그리고 유스케이스 등 수평적인 기술요소들을 아우르는 상호호환 솔루션을 정의함
Established Protocols	• 여러 전송기술들에 걸친 탐색과 연결을 위한 새로운 공통의 통신 프로토콜들에 대해 기존의 것들을 재사용하거나 새로운 것을 확립함
Common Approaches	• 보안과 식별성을 위해 공통적인 접근을 적용함
Defined commonalities	• 공통적인 프로파일들, 객체모델들 그리고 개발자 API들을 정의함
Interoperability	• 여러 시장들과 유스케이스들을 아우르는 디바이스와 응용의 상호호환성을 정의함
Innovation opportunity	• 혁신을 위한 기회를 제공하고 차별화를 지원함
Necessary connectivity	• 최소의 웨어러블 기기부터 가장 큰 차까지 모든 것을 연결함

1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 아키텍처 적용

③ Thread Group

- **Google의 네스트랩스가 주도한 Thread Group이 개발한 아키텍처의 링크계층은 IEEE 802.15.4를 전제로 하여 기존의 관련 칩을 그대로 사용할 수 있도록 함**
  - 기존 지그비 해당계층 프로토콜과 주요 차이점은 UDP와 IPv6사이의 라우팅 프로토콜을 Distance Vector 기법을 기반으로 문제 복원성의 요구사항을 반영함. 보안 표준인 DTLS를 UDP에 적용함

- **Thread Group의 IoT 실현을 위한 아키텍처 요구사항**
  1. 저 전력성 실현
  2. 문제 복원성이 있는 메쉬 네트워크 실현
     - 단일 장애 문제 배제, 자동 복구 가능
     - 간섭에 강함, 자가 확장 가능
     - 중요 인프라에 적용 가능한 신뢰성 보장
  3. IP프로토콜 기반이어야 한다.
  4. 사용하기 편하고 공개되어야 한다.
  5. 기존의 무선 칩을 활용
  6. 보안이 보장되어야 한다.

〈 Thread의 프로토콜 계층 〉

THREAD	Standard
Application Layer	
UDP + DTLS	RFC 768, RFC 6347, RFC 4279, RFC 4492v RFC 3315, 5007
Distance Vector Routing	RFC 1058, RFC 2080
6LowPAN (IPv6)	RFC 4944, RFC 4862, RFC 6775
IEEE 802.15.4 MAC (including MAC security)	IEEE 802.15.4 (2006)
IEEE 802.15.4 PHY	IEEE 802.15.4 (2006)

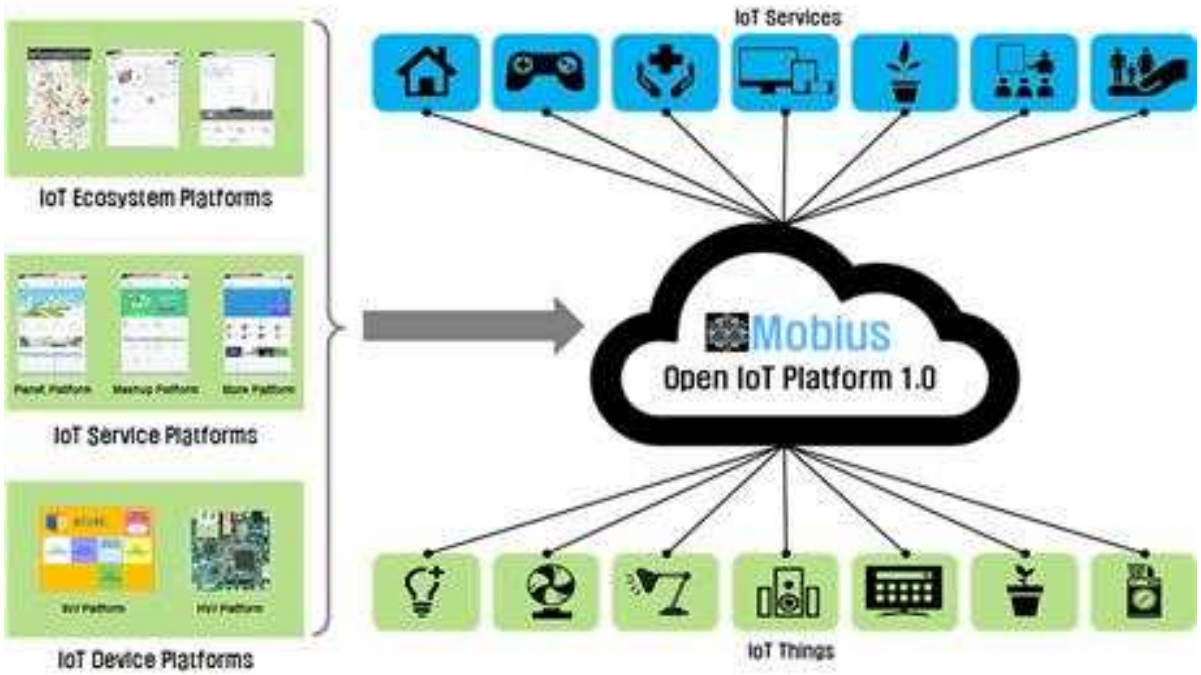
1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 플랫폼

- **더 나은 삶을 제공하기 위한 서비스 프레임워크 기술**

  – 실 세계의 사물들을 네트워크로 상호 연결하여 사람–사물, 사물–사물 간에 언제 어디서나 서로 소통할 수 있도록, 사물들로부터 데이터를 수집하거나 사물에 대한 제어 방법을 제공함.

  – 사물들이 지능적으로 서비스를 제공하기 위해 특정 서비스에 종속적이지 않으면서 데이터의 수집/제공, 사물 기기의 관리, 연결 기능 등을 제공하는 공통시스템

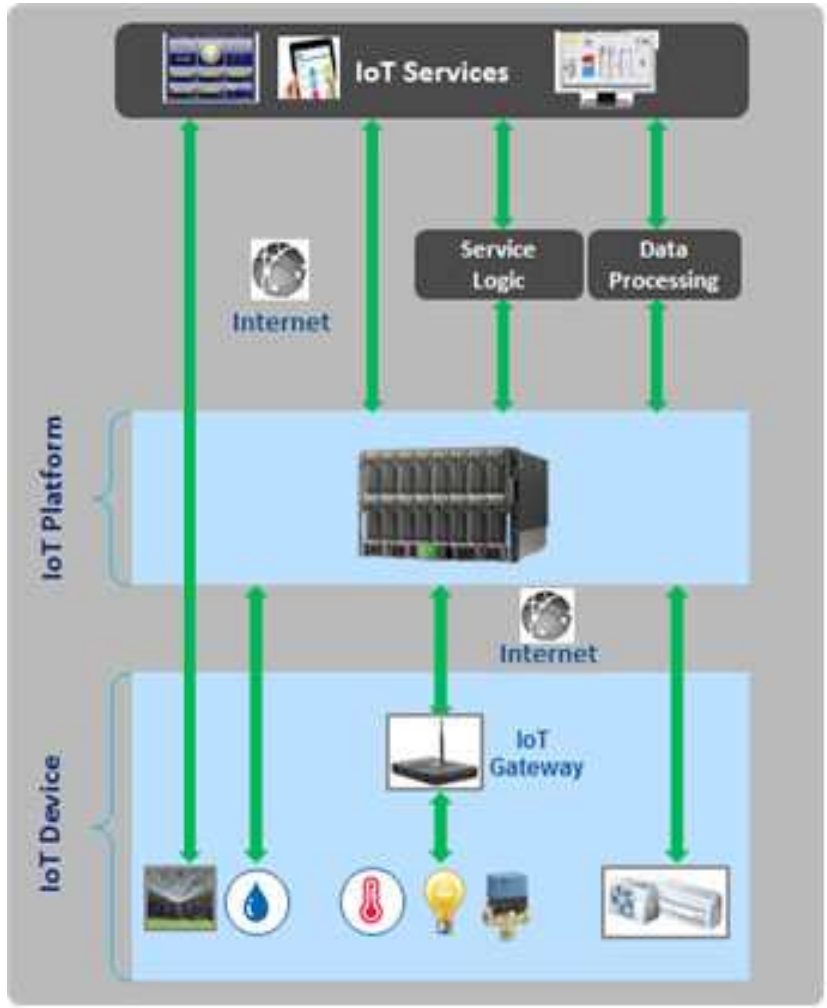


1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 플랫폼

– 사물인터넷 플랫폼의 역할

- **사물인터넷 플랫폼은 서비스를 구성하기 위해 필요한 공통 요구 기능들을 포함하고 있으며 개별 사물과 서비스에서 독립적으로 동작할 수 있어야 함**
  – 플랫폼은 서버나 클라우드 형태로 제공될 수 있으며, 디바이스에 직접 위치할 수도 있음



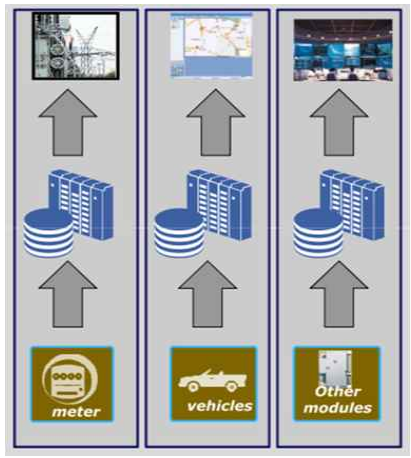
1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 플랫폼의 발전방향

– 수식적 플랫폼에서 수평적 플랫폼으로 변화

수직적 플랫폼

- **응용서비스 도메인 분야별로 별도의 플랫폼 구축, 서비스 추진 함(수직적이고 파편화 됨)**
- **수직적 플랫폼의 문제점**
  - 독립된 플랫폼 구축에 시간 및 비용, 서비스 운용을 위한 유지보수 비용 부담이 큼
  - 융합서비스 제공을 위해 플랫폼간 연동 및 통합 이슈가 발생함



수평적 플랫폼

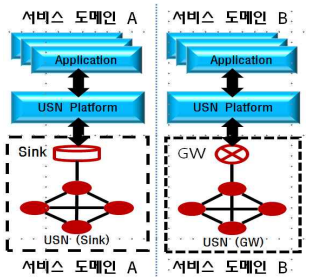
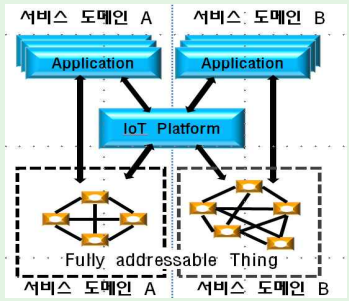
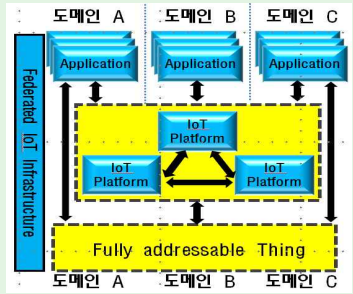
- **사물과 서비스에 독립적으로 동작할 수 있도록 개발되어, 다양한 디바이스 수용이 가능하며, 여러 서비스들의 공통 요구기능을 제공함**

- **수평적 플랫폼의 장점**
  - 특정 디바이스 및 서비스에 종속적이지 않아 유지비용이 저렴함
  - 공통 인터페이스 활용으로 서비스간 융합 및 연계가 용이함

1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 플랫폼의 발전 방향

– 사물인터넷 플랫폼 기술 특징의 현재와 미래

구분	센서네트워크	초기 IoT	미래 IoT
개념도			
서비스 방식	서비스 별 단말, 플랫폼, 어플리케이션 개발. 구축	개방형 사물인터넷 플랫폼 기반 멀티 도메인 서비스 개발. 공유	**개방형 사물인터넷 인프라** 상에서 자유로운 디바이스 및 서비스 공유. 연동
특징	• 폐쇄형(개별플랫폼) • 수직(Vertical) 구조 • 호환성 없음 • 센서 모니터링 중심 • B2B 중심	• 플랫폼기반 개방구조 • 수평적(Horizontal)통합 • 플랫폼간 호환성 없음 • 센서/액츄에이터/데이터 중심 • B2B, B2C, C2C 지원	• 개방형 인프라구조 • 수평적(Horizontal) 통합 • 플랫폼간 호환성 지원 • 데이터/프로세스/지능 중심 • B2B, B2C, C2C 지원
규모	• 근거리/이통망 서비스 규모 (수만개 미만 수준)	• 인터넷 기반 서비스 도메인 규모 (수백만 수준)	인터넷 기반 글로벌 규모 (수백억개 이상 수용)
생태계	개발/구축/운영/유지비용 과다 도메인 중심 생태계	개발 · 구축비용 적음(규모의 경제) 플랫폼 중심 생태계	개발 · 구축비용 최소화 (규모의 경제) 제품 및 서비스 중심 생태계

1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 플랫폼의 기본 구조

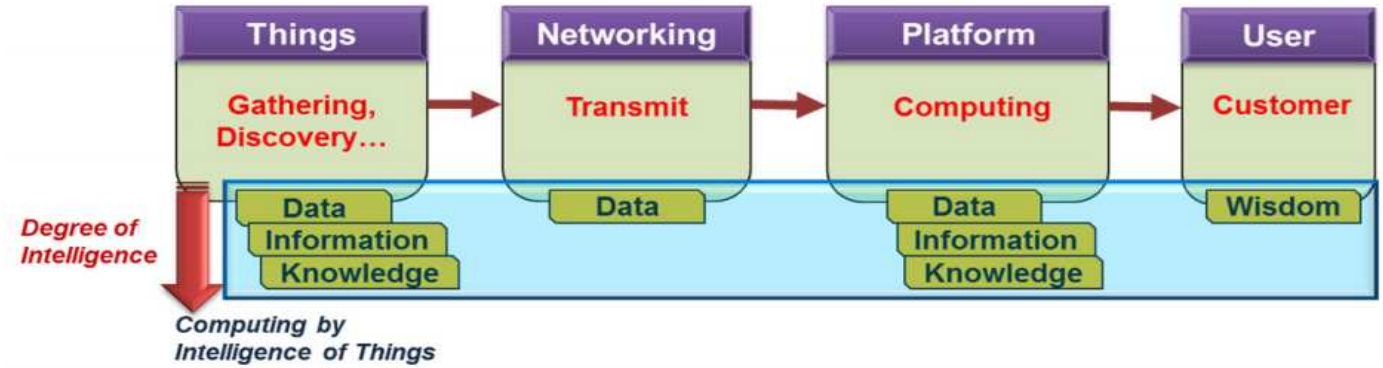
– 사물인터넷 플랫폼의 요건

- **사물인터넷 서비스 구조상에서 사물과 서비스가 요구하는 공통기능을 제공하여, 다양한 사업자들이 쉽게 서비스를 생산, 관리할 수 있고 그 서비스를 사용할 고객(개발자, 서비스 이용자)에 대한 편의가 제공되어야 함**

  – 개발자들이 필요로 하는 기능을 사용하기 쉽게 제공해야 하며, 또한 서비스 활성화를 위해 적은 비용으로 다양한 서비스를 만들 수 있도록 편의를 제공할 수 있어야 함

  – 개발된 다양한 서비스들을 고객들에게 쉽게 활용할 수 있도록 지원해야 함

〈 ISO/IEC JTC1에서의 사물인터넷 서비스 구조 〉

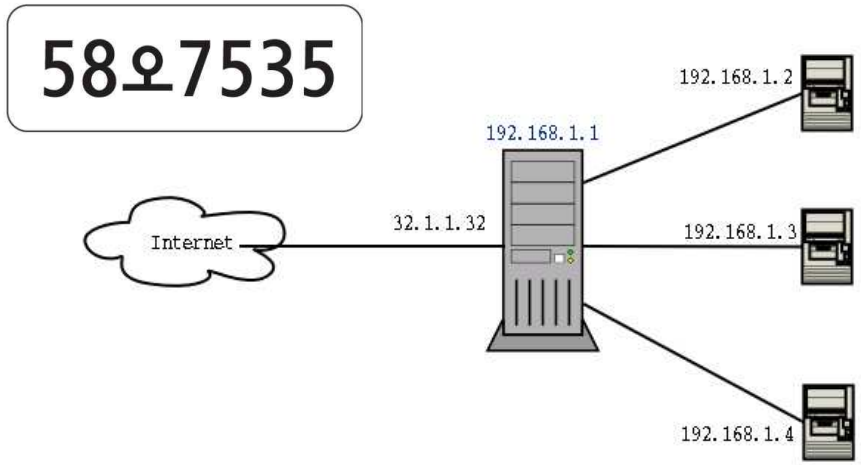


1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 플랫폼 기술

– 식별체계 기술

- **어떤 대상을 유일하게 식별할 수 있는 방법을 제공하는 기술을 식별체계 기술이라 함**
  - 학생번호, 주민등록번호, 자동차번호, 사원번호
  - 인터넷 자원 식별자(URI: Uniform Resource Identifier)
  - 국제 표준 도서 번호(ISBN: International Standard Book Number)
  - 전화번호(MSISDN: Mobile Station International Subscriber Directory Number)
  - IP주소(Internet Protocol Address)
  - 객체식별자(OID:Object Identifier) 등



1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 플랫폼 기술

– 검색 기술

- 사용자가 원하는 서비스를 제공받기 위하여 정보나 리소스 등을 찾고 찾아진 결과를 쉽게 활용할 수 있도록 제공하는 기술

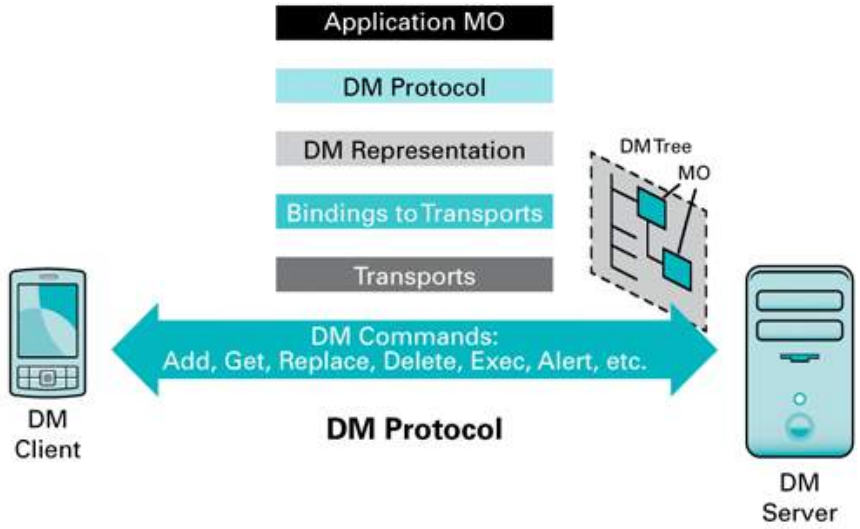
검색기술 구분	내용
클라이언트–서버	• 클라이언트의 검색 요청에 대해 서버는 자신의 저장소에 존재하는 디렉토리로부터 검색 결과를 알려줌 • 일반적으로 글로벌 환경에서의 검색 서비스를 제공할 수 있음 • 대표적으로 oneM2M과 같은 표준이 이러한 방식을 채택함
P2P (peer–to–peer)	• 리소스를 찾고자 하는 장치가 네트워크에 검색요청을 보내고, 검색요청을 받은 장치가 자신이 해당 리소스를 가지고 있을 경우 응답하거나, 모든 디바이스가 자신이 가지고 있는 리소스 정보를 주기적으로 광고함 • 네트워크상에서 리소스를 찾고자 하는 장치는 원하는 리소스에 대한 광고를 수신하여 리소스를 검색함 • 대표적인 P2P 검색 방식은 Alljoyn 의 검색 서비스를 들 수 있음

1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 플랫폼 기술

– 장치관리 기술

- 사물인터넷 장치관리 기술은, 사물인터넷 디바이스의 초기설정, 소프트웨어/펌웨어 다운로드, 디바이스의 고장 진단 및 배터리/메모리 등 하드웨어 모니터링, 디바이스 주변장치(USB, 카메라 등) 컨트롤, 시스템 리부팅, 시스템 로깅 등을 위한 기술임
- 대표적 기술로 OMA(Open Mobile Alliance) DM(Device Management), OMA LWM2M(Lightweight M2M), BBF(Broadband Forum) TR-069 기술을 활용하거나 별도의 장치관련 프로토콜을 개발하여 사용하고 있음



1.3 사물인터넷 플랫폼 아키텍처

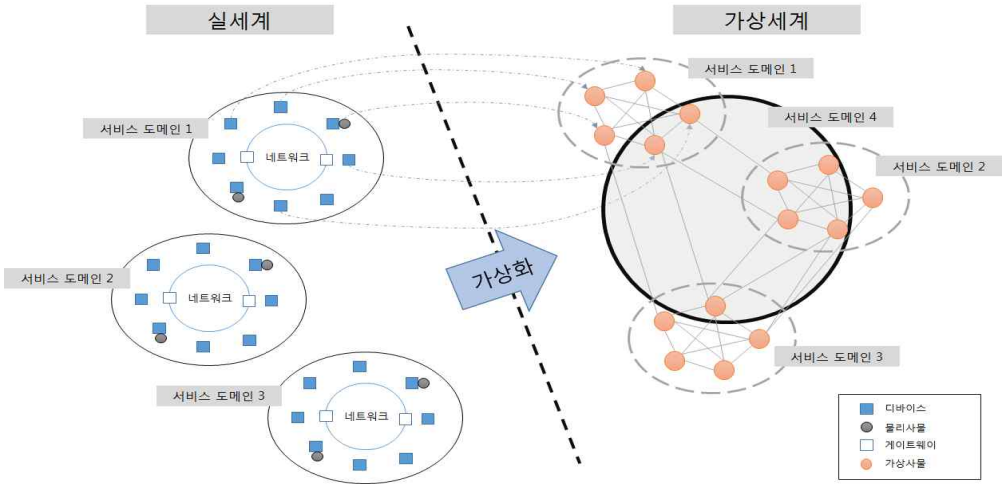
▪ 사물인터넷 플랫폼 기술

– 사물가상화 기술

- **사물 가상화 기술은, 물리적 환경에 존재하는 다양한 사물의 정보를 플랫폼 또는 디바이스에 표현하기 위해 추상화된 형태로 리소스를 생성하는 기술임**

  – 추상화로 리소스는 실제 물리적 환경에 존재하는 사물을 대신하는 형태로 존재하며, 실제 물리적 환경에 존재하는 사물을 모니터링하거나 제어할 수 있음.

- **사물 가상화를 통해 실세계에 존재하는 사물이 지원하는 네트워크, 정보체계 등에 관계없이 가상화된 리소스를 손쉽게 서비스와 연결하거나 매쉬업 서비스를 구성할 수 있음**



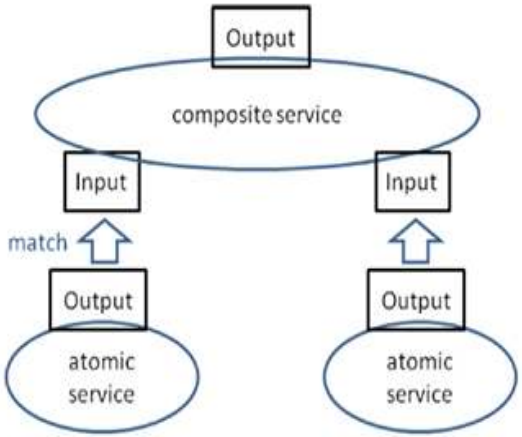
1.3 사물인터넷 플랫폼 아키텍처

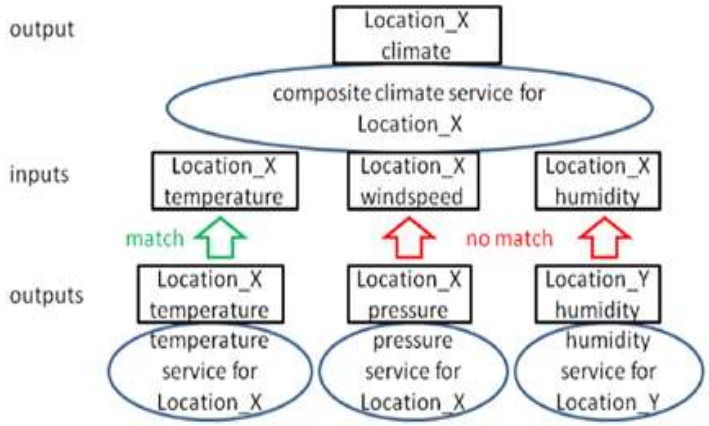
▪ 사물인터넷 플랫폼 기술

– 서비스 컴포지션 기술

- 서비스 컴포지션 기술은, 서비스 지향 구조(SOA, Service-Oriented Architecture)에서 다양한 서비스를 연동하기 위한 개념에서 출발했으며, Service Oriented 또는 Service Choreography 기술의 하부 기술로 사용됨

구분	내용
Service Orchestration	• 사용자 또는 어플리케이션으로부터 특정 서비스를 요청 받았을 때 사물인터넷 플랫폼의 오케스트레이터가 해당 서비스를 검색하고 이와 관련된 서비스를 찾아 제공해주는 기술
Service Choreography	• Service Orchestration으로부터 특정서비스를 요청 받았을 때 정의한 순서 및 명시된 서비스에 따라 서비스를 검색하고 이를 기반으로 서비스를 제공해주는 기술





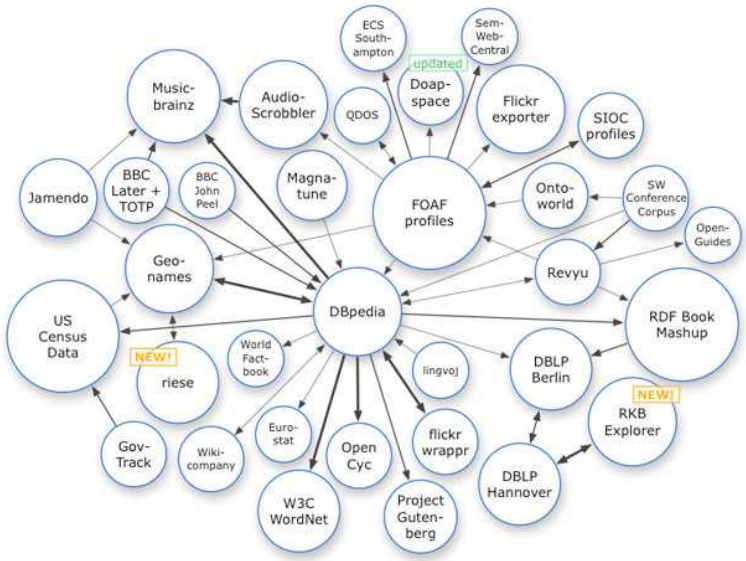
1.3 사물인터넷 플랫폼 아키텍처

▪ 사물인터넷 플랫폼 기술

– 시맨틱기술

- **시맨틱 기술은, 현재의 인터넷과 같은 분산환경에서 리소스(웹 문서, 파일, 서비스 등)에 대한 정보와 리소스들의 관계–의미 정보를 기계가 처리할 수 있도록, 온톨로지(Ontology) 형태로 표현하고 이를 자동화된 기계가 처리하도록 하는 프레임워크 기술임**

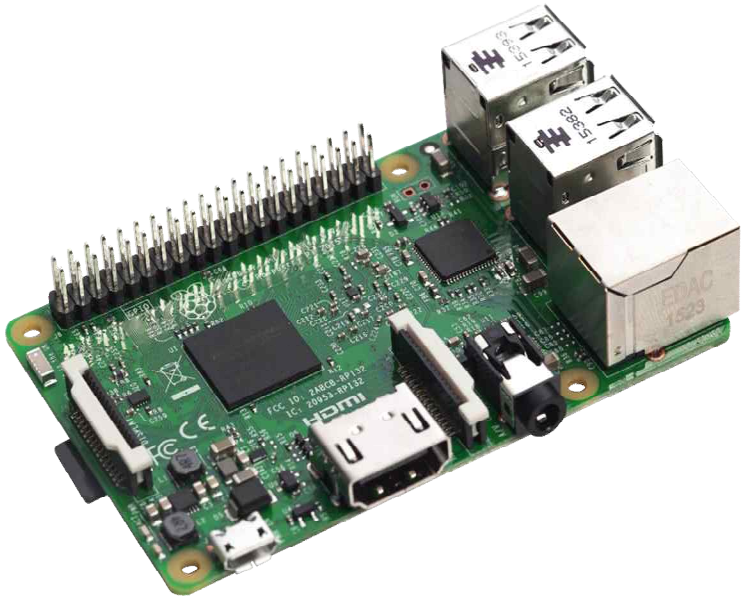
  – 시맨틱 기술은 주로 웹 기반의 어플리케이션 또는 서비스에서 의미적 상호운용을 위하여 사용되어왔으며, 이후 웹 뿐만 아니라 사물인터넷, 빅데이터 등 다양한 시스템까지 확장 사용됨



1.4 오픈소스 H/W 플랫폼

▶ Raspberry Pi

- 영국의 라즈베리파이 재단이 개발한 오픈소스 하드웨어
- 컴퓨터 과학의 교육을 증진시키기 위해 만든 싱글 보드 컴퓨터
- Raspbian (Debian 계열 Linux) 운영체제 사용
- 프로세싱 성능에 중점을 둔 오픈소스 하드웨어

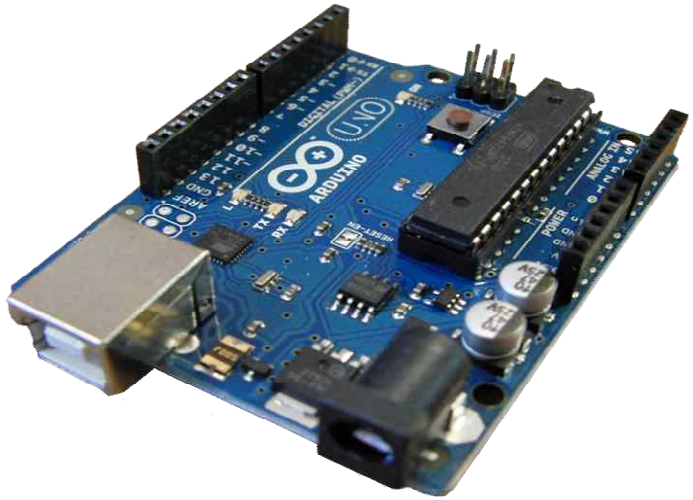


항 목	Raspberry Pi 3 model B
프로세서	*BCM2837 64Bit QUAD Core*
그래픽	*Videocore IV*
속도	*1.2 GHz*
메모리	*1GB LPDDR2 900MHz*
저장장치방식	*Micro SD*
전원공급	*5V/2.5A*
GPIO	*40 Pins*
이더넷	*10/100 Ethernet*
지원포트	*HDMI, audio-video jack, 4xUSB 2.0, Ethernet, Camera Serial Interface, Display Serial Interface*
WiFi	*2.4GHz 802.11n Wireless LAN*
Bluetooth	*Bluetooth 4.1 Classic, BLE*

1.4 오픈소스 H/W 플랫폼

▶ Arduino

- 이탈리아의 IDII(Interaction Design Institutelvera)이 개발한 오픈소스 하드웨어
- AVR 기반의 싱글 보드 마이크로컨트롤러
- 자체적인 IDE를 통해 컴파일된 펌웨어는 USB로 쉽게 업로드 가능
- 센서 및 액추에이터 제어에 중점을 둔 오픈소스 하드웨어

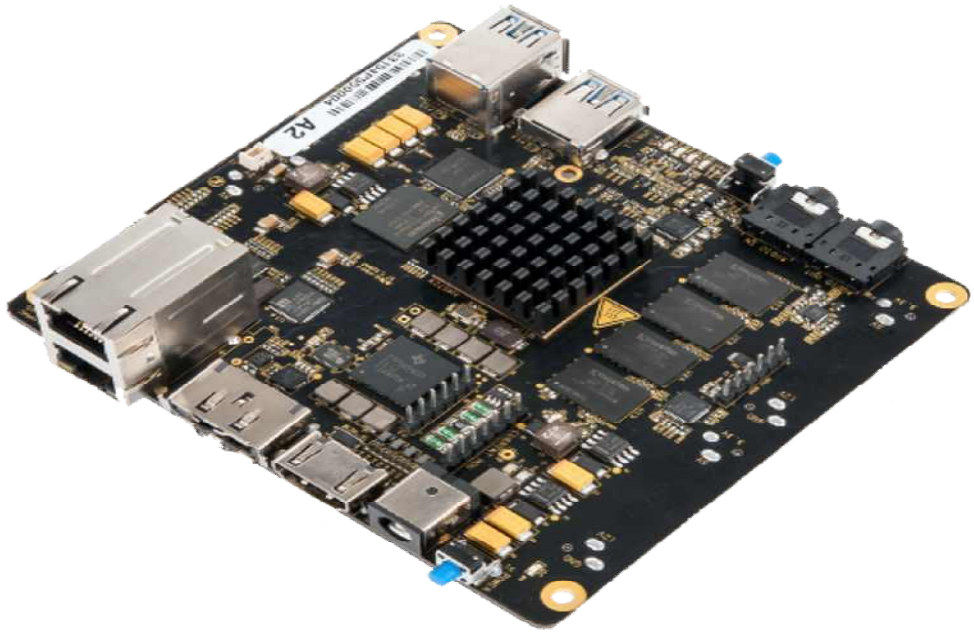


항 목	Arduino UNO
프로세서	*AVR Atmega 328p*
속도	*16 MHz*
메모리	*SRAM 2 KB (Atmega 328p)*
저장장치방식	*Flash memory 32 KB (Atmega 328p)*
전원공급	*7~12V*
동작전압	*5V*
GPIO	*20 Pins*
통신모듈	*None*
운영체제	*None*

1.4 오픈소스 H/W 플랫폼

▶ Beagle Board

- 텍사스 인트루먼트(Texas Intrument)에서 개발한 오픈소스 하드웨어
- 교육용으로 소프트웨어 역량을 키우기 위해 개발된 싱글 보드 컴퓨터
- 리눅스, 안드로이드 등 다양한 운영체제 지원
- ARM 프로세서를 비롯해 고속 비디오, 오디오, 그래픽처리장치를 탑재해 강력한 성능을 구현

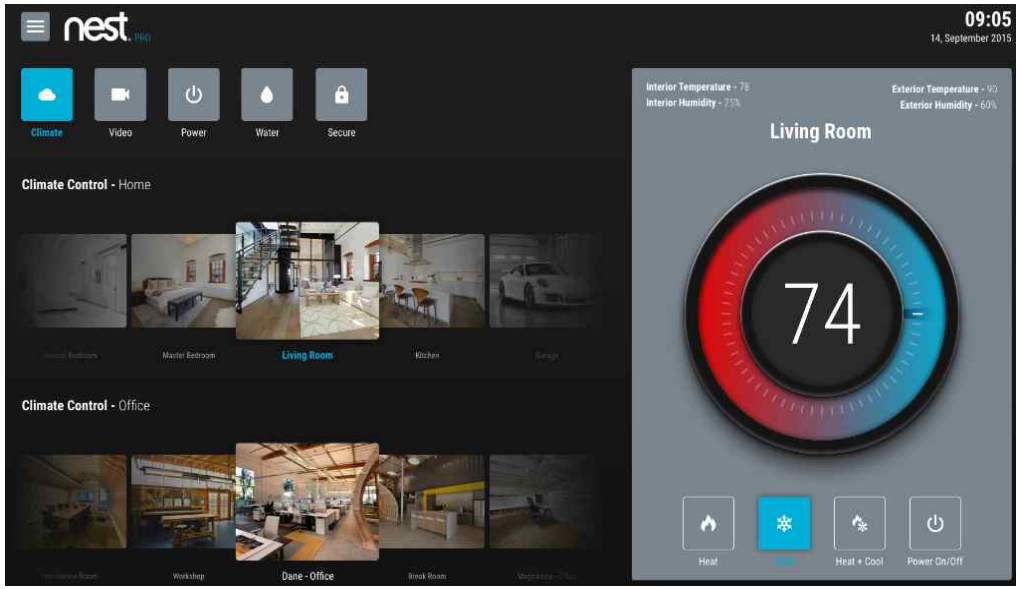


항 목	Beagle Board
프로세서	*OMAP3630 ARM Cortex-A8*
속도	*720 MHz*
메모리	*256MB LPDDR*
저장장치방식	*SD Card*
전원공급	*5 V*
GPIO	*24 Pins*
통신모듈	*None*
운영체제	*리눅스*

1.5 비즈니스 사례

▸ 스마트홈 분야

- 플랫폼 : 구글 네스트
  - 사용자의 온도조절 패턴을 학습한 후 기상정보 등을 종합하여 학습하고 자동으로 온도를 조절
  - 필립스(Philips)의 LED램프(휴(Hue)와 연동하여 거주자의 귀가를 확인하고 조명을 킴
  - CCTV(Drop cam)은 실내외에서 발생하는 움직임을 감지하여 외출 중인 거주자에게 알림
  - 실내 연기나 일산화탄소를 확인하고 화재를 감지하여 거주자에게 알림



1.5 비즈니스 사례

▶ 스마트홈 분야

- 플랫폼 : 애플 홈킷
  - 홈킷은 스마트 잠금장치, 조명, 카메라, 온도조절, 플러그, 스위치 등과 사용자의 아이폰을 안전하게 연결하여 각각의 디바이스를 제어할 수 있는 플랫폼
  - 홈킷은 아이비콘과 음성 비서 ‘시리(Siri)’에 접근할 수 있는 API를 제공하여, 사용자의 동선을 예측할 수 있고 시리를 통해 가전을 조작하는 서비스 가능
  - 최근 애플워치와 홈킷을 연동해 사용자의 사용성을 높여, 운동할 때나 외출 할 때, 집 안에서 스마트폰 없이 애플워치만으로 제어와 상태 전송이 가능

1.5 비즈니스 사례

▶ 스마트시티 분야

- 스마트 주차장
  - 주차장 정보 공유 서비스는 주차장 바닥에 주차된 차량의 존재 여부를 확인할 수 있는 센서를 부착하여 주차장 정보를 자동으로 확인 할 수 있어, 주차장을 찾아 돌아다니는 수고를 줄여줄 뿐만 아니라, 주차 수요를 바탕으로 탄력적인 요금 정책을 적용할 수 있음

1.5 비즈니스 사례

▶ 스마트시티 분야

- 스마트 표지판
  - 스마트 방향표지판은 기존의 아날로그 방향 표지판을 디지털화 한 것으로, 평상시에는 기존의 아날로그 방향 표지판처럼 도시의 주요 시설물의 방향을 안내해 주지만, 표지판이 움직일 수 있도록 하여 추가적인 정보를 제공할 수 있도록 함

1.5 비즈니스 사례

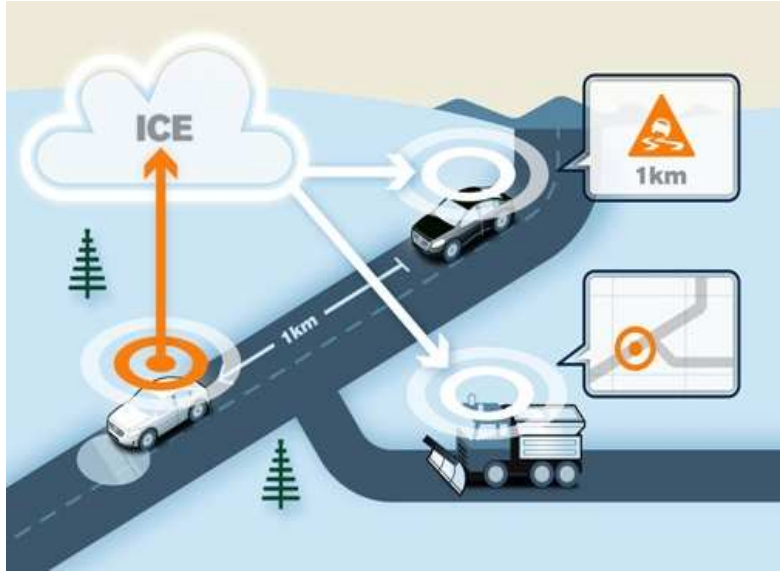
▸ 스마트 물류 분야

- 아마존 물류 창고 로봇 '키바(Kiva)'
  - 직원이 근무하기 힘든 물류창고에서 근무환경을 개선하기위해 아마존은 키바 로봇을 도입하여 20%정도의 물류비용을 절감하고 있으며, 99.6% 정도의 정확도를 제공
  - 기존까지 직원이 선반 사이를 이동하는 통로를 만들어야 했으나, 키바는 선반 아래로 이동하므로 공간활용도가 높으며 작업 효율도 2~3배 증가
  - 제어센터는 키바를 제어하여 320kg의 선반을 담당자가 있는 곳 까지 운반
- 스마트 사물함
  - 최근 일부 대학교에 택배수령을 위한 스마트 사물함을 설치
  - 온라인으로 물품을 주문한 후 사물함 중앙의 단말기에 스마트폰을 가져다 대면 물품이 들어있는 사물함이 열리는 방식

1.5 비즈니스 사례

▶ 스마트 카 분야

- 볼보의 커넥티드카
  - 볼보의 스웨덴 연구소는 전방의 차량이 도로 주행 중에 장애물이나 지형의 위험을 감지
  - 감지된 위험 정보를 서버로 전송하여 인근의 볼보 차량들에게 알려 사고예방(V2V 통신)
  - 스마트폰 앱을 통해 차량의 상태, 위치, 연료량 등을 확인 할 수 있는 기능을 제공 (자체 모니터링)
  - 원하는 시간에 히터와 에어컨 등이 작동할 수 있도록 하는 편의성 기능 제공



1.5 비즈니스 사례

▶ 스마트 카 분야

- 폭스바겐
  - CES 2016에서 폭스바겐은 LG와 제휴를 통해 차 안에서 집안의 조명, 난방, 가전기기를 제어하고 집안에서 차량의 상태를 확인할 수 있는 '커넥티드 카 + 스마트 홈 서비스' 구현
  - 상황인식 센터를 통해 운전자에게 실시간으로 안전하고 직관적으로 최적의 길을 추천
  - V차지(V-charge)는 무인 자동 주차 시스템으로 운전자는 주차장 입구에 내려 스마트폰을 통해 자동으로 주차가 이루어짐

Assignment

- Assignment 1

사물인터넷 플랫폼의 성공적인 실제 사례들을 조사하고

적용된 표준 및 레퍼런스 모델을 분석 하시오.

Reference

[1.1] 사물인터넷소개

사물인터넷 개념

그림 출처 : 한국사물인터넷협회, IoT 지식능력검정 교육 자료

[1.2] 사물인터넷 표준화 정의

정보통신(ICT) 분야 표준의 정의

그림 출처 : 한국사물인터넷협회, IoT 지식능력검정 교육 자료

사물인터넷 표준화 기구

그림 출처 : 한국사물인터넷협회, IoT 지식능력검정 교육 자료

oneM2M

그림 출처 : oneM2M 사물인터넷 서비스 플랫폼 표준화 현황(김기형 LG전자,2014)

oneM2M

그림 출처 : http://www.oneM2M.org/technical/published-documents

IEEE(전기전자공학자협회)

그림 출처 : IEEE에서의 사물인터넷 기술 표준화 현황 (2014 송재승 세종대학교 정보보호학과 교수)

IETF(전기전자공학자협회)

그림 출처 : 한국사물인터넷협회, IoT 지식능력검정 교육 자료

Reference

[1.2] 사물인터넷 표준화 정의

사물인터넷 표준화 기구의 표준화 영역

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

[1.3] 사물인터넷플랫폼 아키텍처

사물인터넷 아키텍처 개요

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

사물인터넷 아키텍처 레퍼런스 모델

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

사물인터넷 아키텍처 레퍼런스 모델

그림 출처1 : http://www.oneM2M.org/technical/published-documents

그림 출처2 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

사물인터넷 플랫폼의 정의

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

사물인터넷 플랫폼의 기술

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

Reference

[1.5] 비즈니스 사례

구글 네스트

왼쪽 그림 : https://fiber.google.com/nest/
오른쪽 그림 : https://dribbble.com/shots/2244586-Google-Nest-Pro-For-Android-TV

애플 홈킷

왼쪽 그림 : http://enjoiyourlife.com/1351
오른쪽 그림 : http://www.ditoday.com/articles/articles_view.html?idno=20237

볼보

내용 :
https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0ahUKEwjW4srEh4_SAhWByrwKHRpjC5sQFggtMAI&url=https%3A%2F%2Fwww.kbfg.com%2Fkbresearch%2FprocessFileDownloadManager.do%3Ffile_name%3D20160302090005_1.pdf&usg=AFQjCNGL8TVPl_HrW-ChwX7hSRxeyK8WLA
왼쪽 그림 : http://www.intellimec.com/
오른쪽 그림 : https://www.engadget.com/2014/03/20/volvo-ice-detection-network/

폭스바겐

왼쪽 그림 : http://www.automotiveit.com/
오른쪽 그림 : http://blog.naver.com/PostView.nhn?blogId=kangjungi78&logNo=220647137131

2장. 사물인터넷과 빅데이터 및 클라우드

Chapter 2. IoT centric Big Data and Cloud

2.1 빅데이터 개요 및 중요성
2.2 빅데이터의 속성
2.3 빅데이터의 분석기술
2.4 클라우드 서비스 개요 및 특징
2.5 클라우드 컴퓨팅 기술
2.6 클라우드 서비스 모델
2.7 사물인터넷 클라우드 구성

- Assignment
- Reference

강의 개요

▪ 강의 목표

사물인터넷(Internet of Things, IoT)과 빅데이터 및 클라우드의 관계를
이해하고 융합기술을 학습한다.

▪ 강의 내용

- 빅데이터의 개념 및 특성
- 빅데이터 분석 기술
- 빅데이터의 국내외 사례
- 클라우드 주요기술 및 서비스 모델
- 사물인터넷과 클라우드 구성

2.1 빅데이터 개요 및 중요성

▪ 빅데이터의 정의

- 사물인터넷은 기존에 연결되지 않은 새로운 99%가 연결되어, 사람과 사물 간의 모든 활동 및 연계 기록이 데이터로 수집 가능한 환경으로 빅데이터(Big Data)를 가치 있는 정보로 가공 후 삶의 질을 향상시키는데 있음.
- 빅데이터는 Public Cloud, Social Data, 그리고 수많은 센서들이 생성하는 스트리밍 데이터를 분석하여 아주 짧은 시간에 의사결정을 가능하게 해 각종 비즈니스 프로세스와 다채널 실시간 마케팅과 같이 시간에 민감함 프로세스에서 중요하게 활용되고 있음.

좁은 의미

- 빅데이터는 기존 방식으로는 수집,저장, 검색과 분석이 어려운 방대한 크기의 데이터 집합
  - 수십에서 수천TB의 크기
  - 다양한 정형/비정형 데이터
  - 생성-유통-소비(이용)의 빠른 속도

넓은 의미

- 빅데이터를 관리,분석하기 위한 인력,조직,기술까지 포괄함
- 빅데이터는 규모(Volume), 다양성(Variety), 증가속도(Velocity), 유효성(Validity), 진실성(Veracity), 가치(Value), 가시성(Visibility) 등을 데이터 집합
- 기술 한계로 과거에 무시했던 데이터 분석 행위

2.1 빅데이터 개요 및 중요성

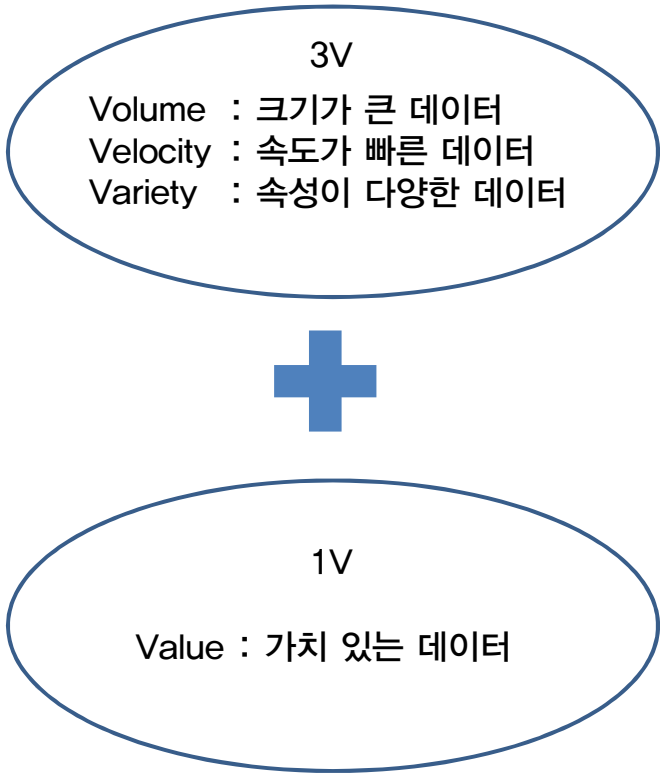
▪ 빅데이터의 중요성

- 빅데이터는 이전에 관리되지 않던 새로운 데이터를 포함하여 업무를 분석하고, 비즈니스 효율성을 향상 시킬 수 있도록 예측 능력을 높임
- 기존 분석 환경과 빅데이터 분석 환경 비교

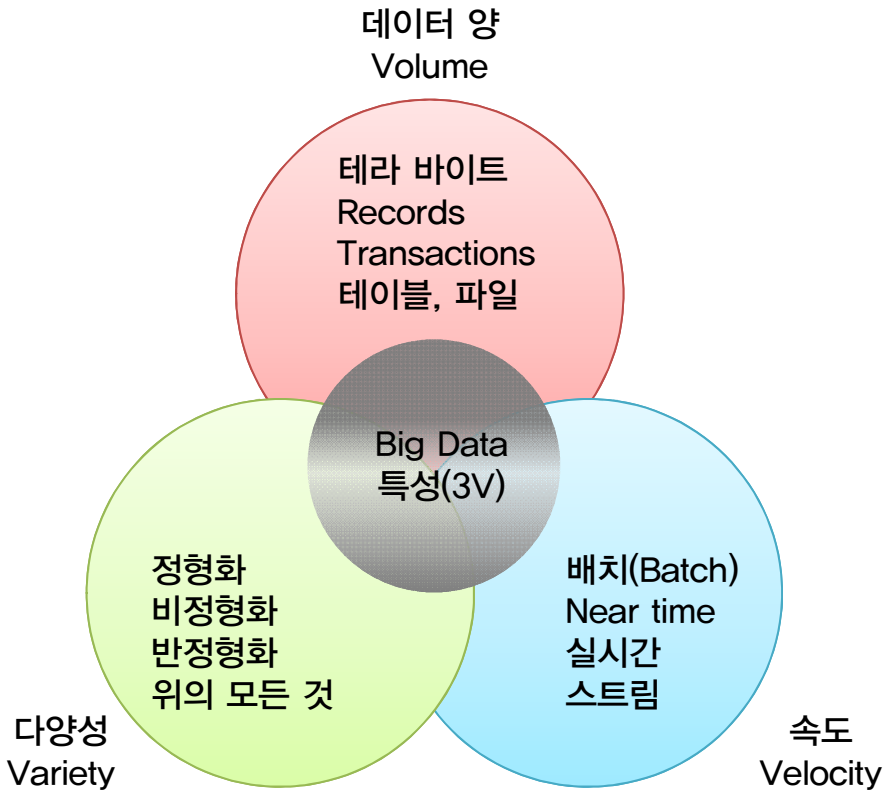
구분	기준	빅데이터 환경
데이터	• 정형화된 수치자료 중심 • 과거 비즈니스 결과 데이터기반 • Fact 중심의 다차원 분석 처리	• 비정형의 다양한 데이터 – 이상 징후 감지, 가까운 미래 예측 • 문자/영상 데이터(SMS, 검색어, CCTV 등) – 위치데이터 • 통계 중심의 상관 관계 분석
하드웨어	• 고가의 저장장치 • 데이터베이스(Data-warehouse)	• 클라우드 컴퓨팅 등 비효율적인 장비 활용 가능
소프트웨어/ 분석 방법	• 관계형 데이터베이스(RDBMS) • 통계패키지(SAS, SPSS) • 데이터 마이닝(Data mining) • Machine learning, knowledge discovery	• 오픈소스 형태의 무료 소프트웨어 • Hadoop, NoSQL, 오픈 소스 통계솔루션(R) • 텍스트 마이닝(Text mining) • 온라인 버즈 분석(opinion mining) • 감성 분석(sentiment analysis)

2.2 빅데이터 속성

- 빅데이터의 속성(3V)



[그림] 빅데이터의 3V 속성



[그림] 빅데이터의 3V 개념도

2.2 빅데이터 속성

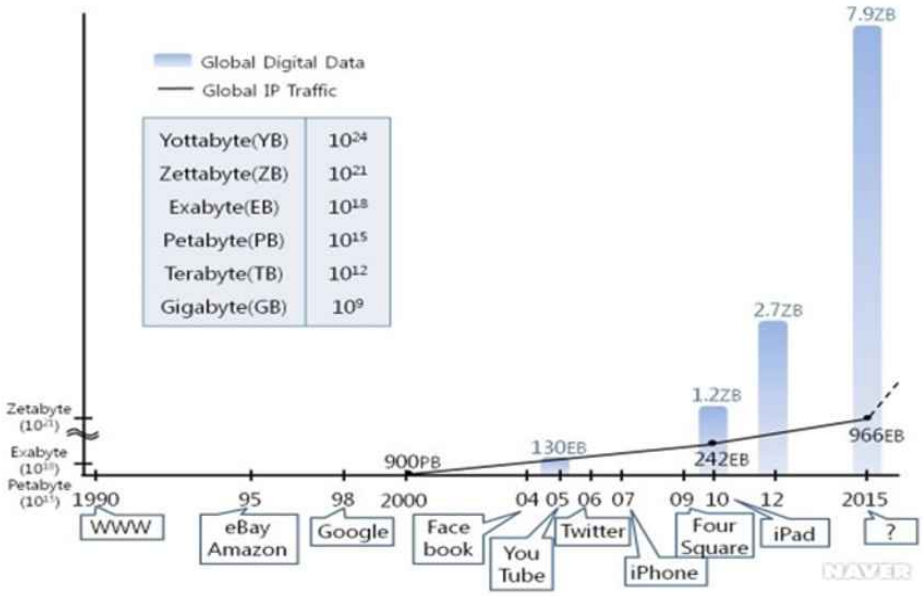
▪ 빅데이터의 속성 : 데이터 양(Volume)

▪ 데이터 양(Volume)

- 데이터의 양은 크기를 말하는 것이지만 단순히 물리적인 크기가 아닌 데이터의 '속성' 이 더 중요하고 그것을 처리하는데 있어 어려움의 유무를 의미하는 것임

▪ 빅데이터의 '빅' (Big)은 얼마 만큼일까?

- 유튜브는 100시간 분량의 비디오가 업로드됨
- 구글에선 410만건의 검색이 이루어짐
- 페이스북에선 330만건의 컨텐츠가 공유됨
- 트위터에선 34만 7222건의 메시지가 '트윗' 됨
- 앱스토어에서는 19만 4064건의 앱 다운로드됨
- 아마존에서는 13만 34364달러의 거래가 이뤄짐
- 판도라에선 3만 1773 시간 분량의 음악이 재생됨
- 이메일은 2억 개가 넘게 전송됨
- 세계적으로 전송되는 데이터량은 157만 2877GB



2.2 빅데이터 속성

▪ 빅데이터의 속성 : 데이터 양(Volume)

- 빅데이터는 확장 가능한 방식으로 데이터를 저장하고, 분석하는 분산 컴퓨팅 기법이 적용
  - 빅데이터는 기존 파일 시스템에 저장하기 어려울 뿐만 아니라, 데이터 분석을 위해서 사용하는 BI/DW 같은 솔루션에서 소화하기 어려울 정도로 데이터 양이 급격하게 증가하고 있음

- 많은 회사가 이미 대량의 로그 형태로 보관데이터를 가지고 있지만 그것을 처리할 능력은 없음
  - 부피가 큰 데이터는 Greenplum 같은 데이터 웨어하우스(data warehouse) 혹은 데이터베이스의 대량 병렬 처리 아키텍처와 아파치 하둡 기반의 솔루션을 활용함

구분	주요내용
데이터웨어하우스	• 미리 정해진 스키마를 포함하고, 규칙적이고 느리게 변하는 데이터 세트에 적합
아파치 하둡	• 하둡은 처리하는 데이터 구조에 조건이 없음 • 하둡은 다수 서버에 걸친 분산 컴퓨팅 문제를 위한 플랫폼 • 고유의 분산 파일시스템인 HDFS를 이용 (다수의 컴퓨팅 노드에서 데이터를 이용할 수 없음)

2.2 빅데이터 속성

▪ 빅데이터의 속성 : 다양성(Variety)

▪ 다양성(Variety)

- 빅데이터에서는 기존의 관계형 데이터베이스로 다루기 어려운 구조화되지 않은 데이터(비 구조화 데이터)를 분석하여 유용한 지식을 얻음
  - 최근에는 구조화된(판매 데이터, 재고 데이터 등) 데이터 외에 최근 텍스트, 위치정보, 센서 데이터, 동영상 등 다양한 형태의 데이터가 급증하고 있음
- 데이터는 정형화 여부에 따라 정형(Structured), 반정형(Semi-Structured), 비정형(Unstructured)으로 나눌 수 있음

[출처] 빅 데이터: 산업 지각변동의 지원 (삼성경제연구소, 2012년 5월)

2.2 빅데이터 속성

▪ 빅데이터의 속성 : 다양성(Variety)

▪ 정형데이터(Structured Data)

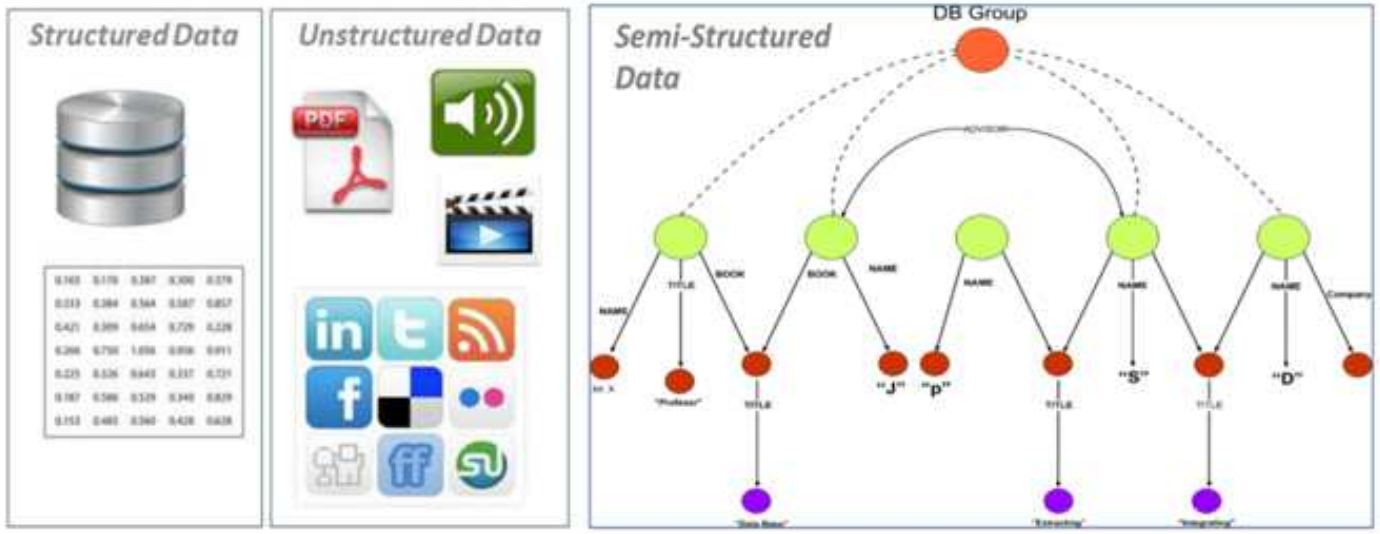
- 고정된 필드에 저장되는 데이터를 의미하며, 기존의 솔루션을 이용하여 비교적 쉽게 보관, 분석, 처리 작업 등이 가능함
  - 온라인 쇼핑몰 제품 주문 시 이름, 주소, 연락처, 배송주소, 결제정보 등을 입력한 후 주문을 하면 데이터베이스에 미리 생성되어 있는 테이블에 저장됨
  - 이때 테이블은 고정된 필드들로 구성이 되며, 일정한 형식을 갖추고 저장되는 데이터를 말함

	A	B	C	D	E	F	G	H	I	J
1	시군구명칭	내과	외과	정형외과	성형외과	산부인과	소아청소년과	안과	이비인후과	피부과
2	강남구	79	25	28	319	46	28	68	51	119
3	강동구	60	8	28	4	20	28	18	27	18
4	강서구	50	6	26	1	14	25	15	28	15
5	관악구	49	7	26	2	20	25	15	22	11
6	구로구	31	3	17	2	16	23	14	24	11
7	도봉구	26	8	8	0	8	16	10	15	6
8	동대문구	39	8	19	4	15	16	13	22	6
9	동작구	38	5	16	4	10	20	15	23	10
10	마포구	39	9	19	9	22	22	14	25	15
11	서대문구	29	10	11	3	9	14	11	18	12
12										

2.2 빅데이터 속성

▪ 빅데이터의 속성 : 다양성(Variety)

- 반정형데이터(Semi-Structured Data)
  - 고정된 필드로 저장되어 있지는 않지만, XML이나 HTML같이 메타 데이터나 스키마 등을 포함하는 데이터를 의미함
- 비정형 데이터(Unstructured Data)
  - 고정된 필드로 저장되어 있지 않은 데이터를 의미하며, 유튜브의 동영상, SNS나 블로그 사진과 오디오 데이터, 메신저 대화내용, 스마트폰 위치정보, 유무선 전화기 통화내용 등이 해당됨



2.2 빅데이터 속성

▪ 빅데이터의 속성 : 속도(Velocity)

- 빅데이터의 속도는 실시간 처리와 장기적인 접근으로 나눌 수 있는데, 지금의 디지털 데이터는 매우 빠르게 생성되기 때문에 데이터의 생산, 저장, 유통, 수집, 분석이 실시간으로 처리되어야 함
- 인터넷과 모바일 시대는 우리가 제품과 서비스를 전달하고 소비하는 방식이 점점 측정되고, 원 제공자에게로 데이터 흐름을 만들어 주고 있음
  - 온라인 소매업자는 단지 최종 판매만이 아닌 고객의 모든 인터랙션 정보를 연결할 수 있음.
  - 고객이 지리정보가 있는 이미지나 오디오 데이터의 스트리밍 소스를 가지고 있기 때문에, 스마트 폰 시대에는 다시 데이터 유입률이 증가함
  - 빅데이터의 배치 처리를 위해 큰 규모의 저장 공간에 빠르게 변하는 데이터를 스트리밍하는 것이 가능함. 입력에서 데이터를 가져와 결정하는 피드백 순환(loop)속도가 중요함

2.3 빅데이터 분석기술

▪ 빅데이터의 분석

▪ 빅데이터 분석 인프라의 요건

- 빅데이터 분석에 필요한 인프라
  - –다양한 시스템에 광범위하게 저장된 데이터에 대해 심층적 분석을 지원하고,
  - –데이터 용량의 확장할 수 있어야 하며,
  - –응답 시간의 단축하고,
  - –분석모델을 기반으로 의사결정을 자동화 할 수 있어야 하고
  - –빅데이터와 전통적 엔터프라이즈 데이터를 통합하여 분석할 수 있어야 함

▪ 빅데이터 다차원 분석의 중요성

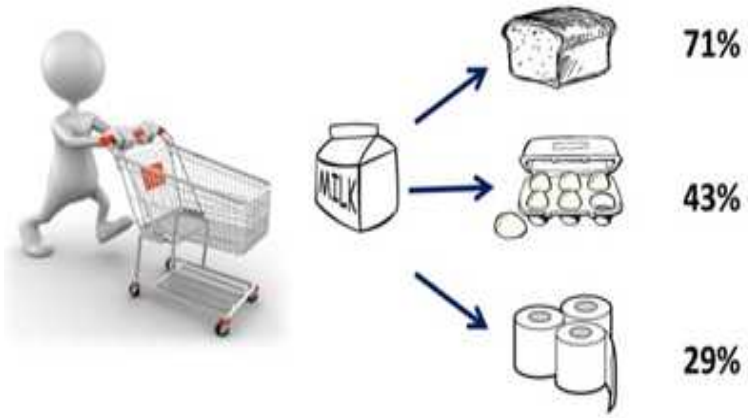
- 위치기반 데이터와 소셜미디어에 의해 생산되는 사용자 활동 데이터가 새로운 차원의 정보로 양산되기 시작
  - –위치정보와 소셜미디어 활동 데이터를 활용해 사용자(고객)의 행동패턴, 선호도와 고객 경험을 파악하는 상황인지(context awareness)가 가능해졌고,
  - –기업들은 자사의 서비스나 제품 등 다양한 제안을 고객의 상황에 맞게 즉시 추천할 수 있게 됨

2.3 빅데이터 분석기술

▪ 빅데이터의 분석

▪ 빅데이터 분석 알고리즘

- 연관규칙학습(Association Rule Learning)
  - 특정한 성격을 가진 데이터군과 일정한 규칙에 따라 연결되는 다른 특정한 성격의 데이터군을 찾아내는 방법
  - {양파,감자}={햄버거}라는 경향이 상품 판매 데이터에서 발견되는 경우, 소비자가 양파와 감자를 살 때는 햄버거와 고기 또한 같이 살 확률이 높음



[그림] 연관규칙학습 예시

- 분류(Classification)
  - 특정한 어떠한 규칙 혹은 특성을 기준으로 구분된 데이터군을 기반으로 새롭게 추가되는 데이터가 속할 만한 데이터군을 찾아내는 방법임
  - 고객들의 구매 결정, 해지, 소비율 등을 설명할 기준이 되는 명확한 가정이나 데이터가 있을 경우 이용하며, 군집화가 상반되는 개념임

2.3 빅데이터 분석기술

▪ 빅데이터의 분석

▪ 빅데이터 분석 알고리즘

• 군집화(Clustrering)

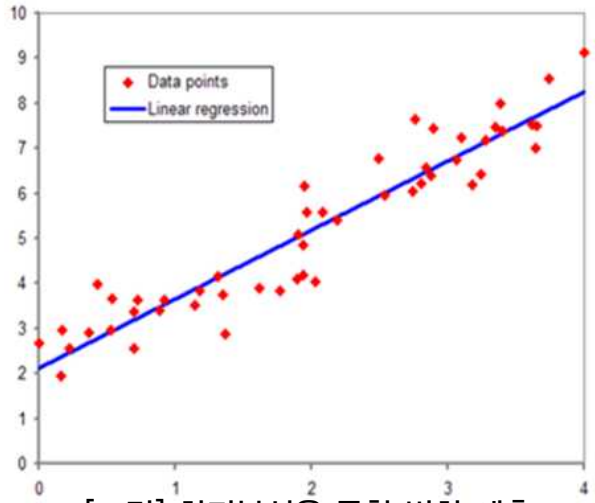
– 하나의 큰 데이터 군을 통계적 기법을 활용하여 비슷한 특성(유사성)을 지니는 여러 개의 작은 묶음으로 분류하는 방법으로 분류의 기준이 되는 유사성은 사전에 정해지지 않음

– 고객군을 비슷한 특성을 가진 소집단으로 묶어 타깃 마케팅 그룹을 만들려고 할 때 활용하며, 훈련 데이터군이 이용되지 않기 때문에 비지도학습(Unupservised Learning)이라함

• 회귀분석(Regression)

– 어떠한 현상을 구성하는 종속변수 값의 변화가 하나 이상의 독립변수 값을 변화시키는지, 어떻게 변화시키는 지의 여부를 찾아내는 방법임

– 보통 변화 예측을 할 때 사용하며, 소비자 만족도에 가장 큰 기여를 하는 변수를 찾아내거나 다양한 시장이나 경제적 변수에 따른 판매량 예측 등에 활용함



[그림] 회귀분석을 통한 변화 예측

2.3 빅데이터 분석기술

▪ 빅데이터의 분석

▪ 빅데이터 분석 알고리즘

• 감성분석(Sentiment Analysis)

– 자연어처리 기술에 기반하여 웹을 포함한 텍스트 기반의 문서에서 글쓴이의 감정을 나타내는 정보들을 찾아내 긍•부정도(긍정, 중립, 부정)를 분석하여 특정 주제에 대해 갖고 있는 성향을 파악하는 기법

– 블로그, 트위터, 페이스북 등의 소셜미디어를 분석하여 고객군을 파악하는 기법으로, 고객, 주주들이 기업에서 새로운 서비스에 대해 나타내는 긍 •부정 성향을 파악하여 서비스에 반영하려는 시도를 하고 있음

[그림] 감성분석을 통한 빅데이터 분석

2.4 클라우드 서비스 개요 및 특징

▪ 클라우드 서비스 정의

▪ 클라우드 서비스는 인터넷 기반의 자원, 소프트웨어 및 정보 인프라를 제공하는 것으로, 인터넷을 통한 요청형 제공방식(On-demand)의 서비스임

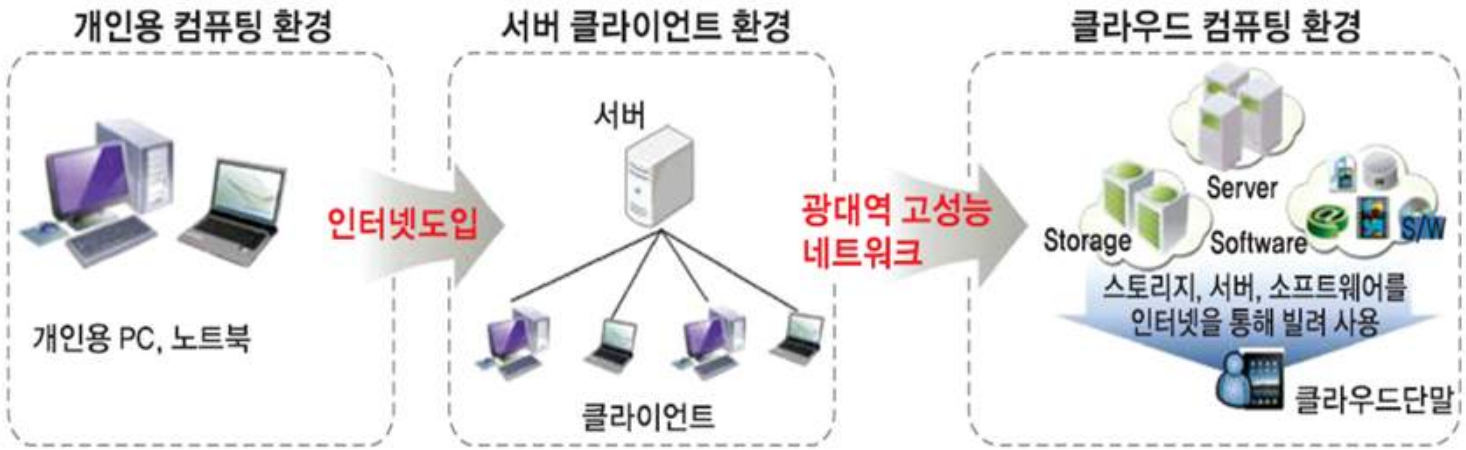
- 클라우드 서비스 제공자는 다량의 컴퓨터 자원을 분배, 가상화하여 각 이용자에게 제공하고, 서비스 이용자는 클라우드 서비스를 통해 자신의 컴퓨터에 직접적인 프로그램 설치 없이도 원하는 자원을 필요할 때 필요한 만큼, 즉각적으로 인터넷을 통해 서비스 받을 수 있음

▪ 클라우드 서비스는 최근 스마트폰과 같은 모바일 기기의 확산으로 다양한 서비스가 개발되고, 실시간 서비스가 이루지는 환경에서 트래픽 증가와 서비스의 지속성을 유지하기 위해 많은 곳으로 검토되고 도입되고 있음

2.4 클라우드 서비스 개요 및 특징

▪ 컴퓨팅 환경의 변화

- 초기의 컴퓨팅 환경은 개인 PC를 사용하여 직접 연산 및 처리를 수행하는 형태 하였으나, 인터넷이 확산되면서 데이터 연산과 처리의 일부가 인터넷 서비스로 대체됨
- 네트워크의 발전으로 이용자가 직접 컴퓨팅 자원을 구매하지 않아도 원격으로 클라우드 서비스에서 컴퓨팅 자원 및 소프트웨어를 전부 임대하여 사용할 수 있는 클라우드 컴퓨팅 환경이 구축됨
  - 클라우드 서비스는 데이터의 위치 및 소유, 컴퓨팅의 주체 및 관리, 서비스의 제공 등 기존의 컴퓨팅 환경과 구분되는 특성을 가짐



2.4 클라우드 서비스 개요 및 특징

▪ 컴퓨팅 환경의 변화

▪ 클라우드 서비스로 제공되는 스토리지 서비스, 소프트웨어 임대 서비스의 경우 웹하드, SBC(Server Based Computing)등 기존의 응용인터넷 서비스와 유사해 보일 수 있으나, 가상화 서비스를 기반으로 서비스를 제공하고 있어, 기존 구축 형태의 Fixed 된 서비스들과 구별이 됨

기존 응용 인터넷 서비스	클라우드 서비스
웹하드 • 단순 파일 저장 기능 • 파일 다운로드 후 개인 PC에서 가공	스토리지 제공 서비스 • 다양한 단말과 데이터 동기화 서비스 지원 • 서버에서의 데이터 가공 서비스 지원
ASP • 단일서버로 서비스 제공자가 특정 사용자를 대상으로 환경 구축 후 사용 • 사용자에 의한 컴퓨팅 환경 변경 불가	가상 서버/데스크탑 서비스 • 가상화된 서버 그리드로 서비스 제공자는 서비스 제공을 위한 공동 플랫폼만 구축 • 사용환경 설정이 사용자에 의해 간편하게 이루어짐
웹하드 • 사업자가 지원하는 고정적인 형태의 서비스만 사용	소프트웨어 제공 서비스 • 사용자가 원하는 S/W들로 사용환경을 동적으로 구성할 수 있는 기능 제공

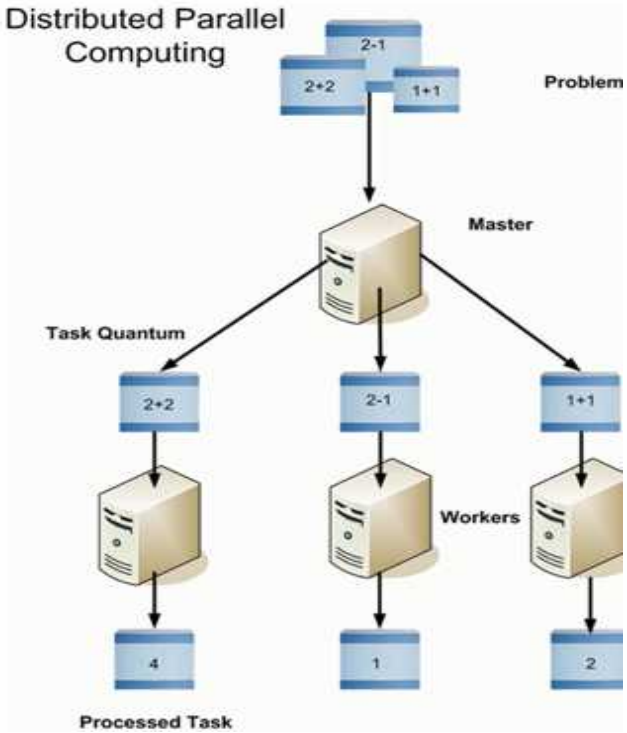
2.5 클라우드 컴퓨팅 기술

▪ 클라우드 컴퓨팅 주요기술

▪ 빅데이터 분석 알고리즘

• 분산컴퓨팅은 클라우드 컴퓨팅 하드웨어 구성 시 인트라넷 또는 인터넷으로 연결된 다수의 컴퓨팅 자원을 하나로 연결하는 기술임신뢰성을 확보한다.

- 분산컴퓨팅과 기술로는 분산파일시스템, 분산데이터베이스 등
- 분산컴퓨팅은 독립적인 파일 시스템 및 데이터 베이스를 단일 시스템으로 인지하고 접근할 수 있도록 함
- 대용량 데이터들에 대한 빠른 처리 속도를 가져올 수 있음
- 분산컴퓨팅에서는 각 하드웨어 자원이 고장 날 수 있다는 것을 전제하고, 대비책을 마련하여 자료에 대한 신뢰성을 확보한다.



2.5 클라우드 컴퓨팅 기술

▪ 클라우드 컴퓨팅 주요기술

▪ 가상화

- 가상화는 넓은 의미로는 컴퓨터 자원에 대한 추상화를 의미하며, 클라우드 컴퓨팅에서 가상화는 자원가상화(resource virtualization)를 의미함
  - 자원가상화는 스토리지 볼륨, 네임 스페이스, 네트워크 자원 등의 시스템 리소스 가상화를 말함

▪ 시스템 관리

- 클라우드 컴퓨팅에서 시스템 관리는 사용자의 가상 컴퓨팅 환경을 프로비저닝(Provisioning)이라 하고, 제공된 가상 시스템을 모니터링하며, 사용자 서비스 별 자원활용 정동에 따른 동적 자원 할당 및 동적 스케쥴링을 제공함
- 컴퓨팅을 구성하는 주요 시스템 솔루션 마스터를 관리하여 시스템 전체의 고 가용성을 보장함

2.5 클라우드 컴퓨팅 기술

▪ 클라우드 컴퓨팅 주요기술

▪ 서비스 플랫폼 서비스

- 플랫폼은 사용자들이 클라우드 컴퓨팅 인프라에 사용자 고유의 응용 또는 인터넷 서비스를 구축하기 위한 인터페이스를 제공함
  - 서비스 간 호환성을 위해 SOA를 기반으로 하며, 단순화된 SOAP이나 REST 프로토콜을 제공함.
  - 서비스 플랫폼에서는 소프트웨어 개발 환경들과 보유 서비스들의 API를 제공하며, 협업을 위한 인터페이스, 대용량 데이터 처리를 위한 분산 병렬 처리 환경, 데이터베이스 인터페이스 등을 제공함

▪ 기타 : 보안, 과금, 사용자 인증 등

- 클라우드 컴퓨팅의 사용 용량에 따른 과금 정책 및 사용자 인증 인터페이스를 제공하고, 사용자들의 데이터 접근 등에 대한 트러스티드 플랫폼 기술을 확보하여 사용자들이 클라우드 컴퓨팅 서비스를 신뢰하고 사용할 수 있어야 함.

2.5 클라우드 컴퓨팅 기술

▪ 클라우드 컴퓨팅 주요기술

– 유의사항

유의사항	주요내용
확장성(Scalability)	• 클라우드 컴퓨팅은 사용하는 부하에 따라 사용자 가상시스템을 신축성 있게 운영하기 위해 서비스 스케쥴러와 프로비저닝 기술, 대단위 확장성 보장 기술, 서비스 중단 없이 서버를 추가 및 확장할 수 있는 기술 필요
가용성(Availability)	• 가용성은 총 시간과 서비스 접근 가능한 시간에 대한 비율로 99.999%(연간 다운로드시간 5분)가 최고 목표 수준임 • 클라우드 컴퓨팅 인프라의 하드웨어적인 결함이나, 시스템 다운에 대한 대체 솔루션 필요
신뢰성(Reliability)	• 입력되는 작업(task)들이 적법한 작업인지를 판단하는 항목 등 해킹 침입에 대비한 기술 • 저장된 데이터에 대해 일부 내용이 파손되거나 유실될 경우를 대비한 자동 백업과 싱크, 복구 기능 등을 제공
활용률(Utilization)	• 활용률은 클라우드 컴퓨팅 자원의 집약적 운영을 위한 것으로, 운영비용 절감을 위한 기술 • 부하에 따라 불필요한 자원을 끄는 방법으로 적은 컴퓨팅 리소스로 운영하여 비용을 절감함
협업성, 이동성(Mobility)	• 중앙집중 형태로 저장된 데이터를 여러 협력자 간 공유하고 협업할 수 있어야 함 • 사용자 간 클라우드를 접근하는 위치나 단말에 무관하게 동일한 작업을 수행 할 수 있는 인터페이스를 제공해야 함(데이터 공유, 데이터 이동성 보장)

2.6 클라우드 서비스 모델

▪ 클라우드 서비스 모델

▪ 서비스 제공 자원에 따른 모델 구분

•클라우드 서비스는 제공하는 자원의 레벨에 따라 세가지 모델로 분류한다.

2.6 클라우드 서비스 모델

▪ 클라우드 서비스 모델

▪ 서비스 제공 자원에 따른 모델 구분

서비스모델	주요내용	서비스 예
IaaS	• 이용자에게 서버, 스토리지 등의 하드웨어 자원을 임대, 제공 –CPU 성능, 메모리 및 하드디스크 크기 등의 물리적인 자원성능을 주문 – 제공받은 PC 또는 서버를 활용하여 환경 구성	• 스토리지 제공 서비스 –애플 아이클라우드, KT 유클라우드, 드롭박스 등 • 가상 서버 / 데스크탑 제공 서비스 –아마존 EC2 등
PaaS	• 이용자에게 SW 개발 플랫폼을 임대, 제공 –이용자는 SW 개발 플랫폼 주문 –이용자는 주문한 환경에서 응용프로그램 개발 –응용프로그램을 본인이 사용하거나 타인에게 서비스하는 방식으로 활용	• 응용프로그램 개발 환경 제공 서비스 –구글 AppEngine, VM웨어 Cloud Founday 등
SaaS	• 이용자가 원하는 소프트웨어를 임대, 제공 –클라우드 서버에 설치된 SW를 온라인으로 제공 –이용자는 원하는 SW를 물리적 구매나 설치할 필요 없이 원격의 가상공간에서 운용	• 세일스포스닷컴 : 고객관리 프로그램 –픽스러 : 이미지 편집 S/W(포토샵) 제공 –구글 Docs : 문서 편집 및 공유 기능 제공 –틸론 elcloud : office, CAD 등의 S/W 제공 등

2.6 클라우드 서비스 모델

▪ 클라우드 서비스 모델

▪ 서비스 대상에 따른 모델 구분

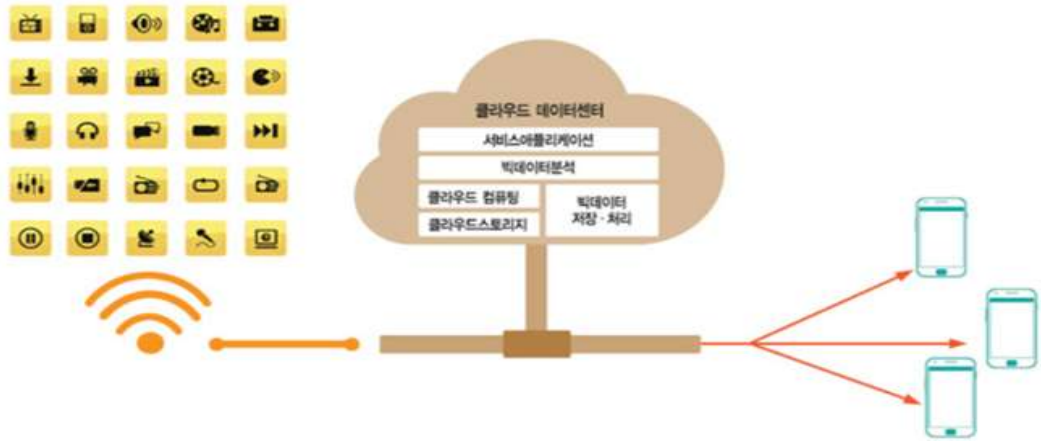
서비스모델	주요내용
공공용 클라우드 서비스 (Public)	• 불특정 다수의 사람들을 대상으로 하는 서비스 – 일반적인 이용자가 아닌 제 3의 서비스 제공자가 운영, 관리 – 특정 기업만 제한적으로 제공하지 않고 여러 서비스 사용자에게 제공
사설용 클라우드 서비스 (Private)	• 기업 및 기관 내부에 제한적으로 서비스 제공 – 특정 기관 내부 데이터센터에 구축 – 일반 이용자들의 접근을 허용치 않음
혼합형 클라우드 서비스 (Hybrid)	• 공공용, 사설용 클라우드 서비스가 합쳐진 것 – 공유를 원치 않는 일부 데이터 및 서비스에 대해 사설용 정책 설정 – 상대적으로 중요도가 낮은 어플리케이션이나 일반데이터는 공공용 – 조직의 핵심 어플리케이션이나 민감한 데이터는 사설용 클라우드에서 관리

2.7 사물인터넷 클라우드 구성

▪ 클라우드 서비스 기반의 사물인터넷

- 사물인터넷 환경에서는 디바이스 하나하나가 데이터의 생산자이자, 소비자로서의 역할을 수행함
  - 디바이스들은 실시간으로 데이터를 생산하고, 생산된 데이터는 사용자 또는 다른 디바이스가 소비함
  - 디바이스 숫자 또는 사용자 수에 비례하는 데이터가 아닌 상호관계에 의한 데이터의 폭발적 증가

- 보다 많은 디바이스에 인터넷 연결 가능한 기술들이 적용되고, 보다 많은 삶의 영역들이 자동화 될수록, 보다 많은 컴퓨팅과 스토리지 자원이 필요하게 되어, 클라우드 인프라에 대한 수요는 증가할 것임



2.7 사물인터넷 클라우드 구성

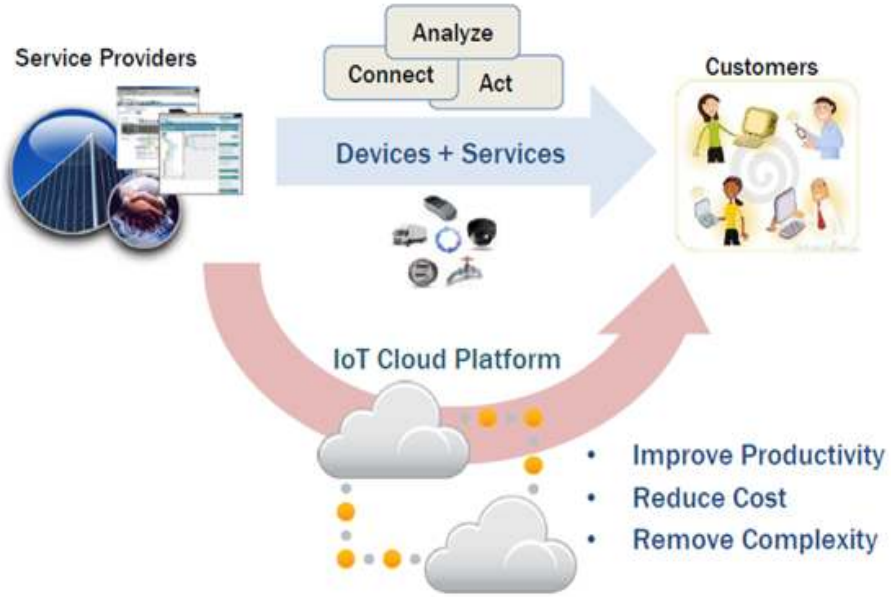
▪ 사물인터넷 클라우드 구성

▪ 혁신적인 사물인터넷 서비스를 촉진하는 인프라가 사물인터넷 클라우드임

- 사물인터넷을 위한 생태계는 센서, 디바이스를 포함하여 이를 연결하기 위한 통신, 각종 응용분야에 특화된 기술과 솔루션, 그리고 다양한 서비스를 개발할 수 있는 범용 플랫폼으로 구분됨

▪ 사물인터넷 클라우드 구성요소

- 사물, 연결, 운영, 응용계층으로 구성
- 센서를 포함하는 물리적 장치
- 연결을 위한 유무선 통신망 및 프로토콜
- 운영을 위한 논리적 프로세스와 관리 서비스
- 서비스를 제공할 응용프로그램
- 보안기능, 관리기능



Assignment

- Assignment 2

사물인터넷 클라우드의 성공적인 실제 사례들을 조사하고

적용된 클라우드 모델을 분석 하시오

Reference

[2.1] 빅데이터 개요 및 중요성

빅데이터의 정의

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

[2.2] 빅데이터 속성

빅데이터의 속성 : 데이터 양(Volume)

그림 출처 : 삼성경제연구소, 2012년 5월

빅데이터의 속성 : 다양성(Variety)

그림 출처 : 삼성경제연구소, 2012년 5월

[2.3] 빅데이터 분석기술

빅데이터의 분석

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

[2.4] 클라우드 서비스개요 및 특징

컴퓨팅 환경의 변화

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

[2.5] 클라우드 컴퓨팅 기술

클라우드 컴퓨팅 주요기술

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

[2.6] 클라우드 서비스 모델

클라우드 서비스 모델

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

Reference

[2.7] 사물인터넷 클라우드 구성

클라우드 서비스 기반의 사물인터넷

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

사물인터넷 클라우드 구성

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

3장. 사물인터넷 네트워크 I

Chapter 3. Internet of Things Network I

3.1 IEEE 802.11 WLAN
3.2 BLE(Bluetooth Low Energy)

- Assignment
- Reference

강의 개요

▪ 강의 목표

사물인터넷(Internet of Things, IoT)의 통신 프로토콜을 이해한다.

▪ 강의 내용

- IEEE 802.11 기반 WLAN 표준 기술
- BLE의 개념 및 동작 모드

3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 등장 배경

▪ WLAN

- 건물, 대학 캠퍼스와 같이 소규모 지역에서 무선 전파를 사용하여 접근 지점으로부터 각 단말까지 통신을 제공하는 기술

▪ WLAN 기술의 변화 과정

- 1990년대 초 미국의 연방통신위원회(FCC)에서 ISM(Industrial, Scientific and Medical)을 위한 비인가 주파수 대역을 사용할 수 있도록 허가
- 1997년 6월 전기전자기술협회(IEEE)에서 각종 무선 LAN 기술을 통합하여 802.11로 표준화
- 1999년 9월 IEEE에서 고속 무선 이더넷 표준 IEEE 802.11b를 승인
- 이후, 본격적으로 무선 LAN 기술의 응용이 확대되면서 낮은 전송률과 간섭의 문제를 보완하여 802.11g, 802.11n 표준을 승인

3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 기반 WLAN 표준

▪ IEEE 802.11 정의

- 무선 근거리 통신망(WLAN)에 관한 프로토콜과 전송 규격
- IEEE 802 표준 위원회의 11번째 워킹 그룹에서 개발된 표준 기술

▪ Wireless Fidelity, WiFi 정의

- IEEE 802.11을 준수한 표준
- 와이파이 연합(WiFi Alliance)의 인증을 받은 무선 근거리 통신망

3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 특징

- 저전력을 사용하여 동작
- 전 세계적으로 허가된 비인가 주파수대역(ISM 대역) 사용
- WLAN 기술의 물리계층과 데이터링크 계층에 대한 표준 정의
- 대역 확산 기술 사용
- 접근지점(AP)의 수에 의해 적용범위 확장 가능
- 무선 반경 내의 제약 없는 통신을 통해 유연성 제공
- 유선 네트워크 구축으로 발생하는 비용 절감

3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 기반 WLAN 표준

표 준 안	속도[Mbps]	거리 [m]	주파수대역 [GHz]	특 징
IEEE 802.11a	54	33	5	• 중복되지 않은 채널을 제공하여 간섭이 적은 편 • OFDM 변조를 사용하여 5 GHz 주파수대역을 사용 • 주파수 대역이 달라 상호간 호환이 없음
IEEE 802.11b	11	100	2.4	• 대부분의 무선 LAN 시스템에서 사용됨 • 최대 100m의 전송거리 제공 • 겹치지 않는 실질적인 채널은 최대 3개에 불과 • 전송속도가 느림
IEEE 802.11g	54	80	2.4	• 2003년 완성된 표준 기술 • IEEE 802.11b와 호환가능 • 2.4GHz의 기기들로부터 간섭영향
IEEE 802.11n	300	70	2.4 / 5	• 다중 안테나 기술을 통해 성능 향상 • 2.4GHz의 기기들로부터 간섭영향

3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 프로토콜 계층

▪ OSI 7계층 모델과 IEEE 802 레퍼런스 모델 비교

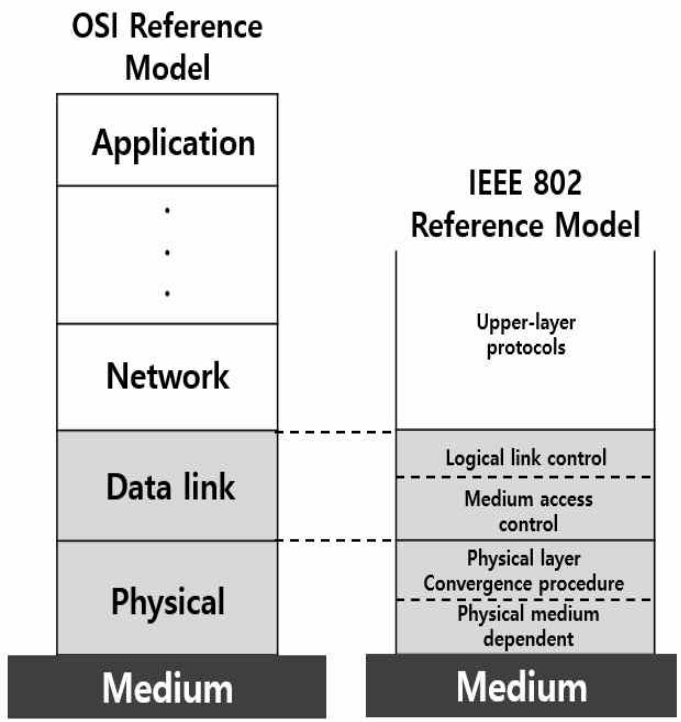
- 데이터 링크 계층(Data Link Layer)
- 물리 계층(Physical Layer)

▪ 데이터 링크 계층(Data Link Layer)

- 논리적 링크 제어 계층(Logical Link Control Layer)
- 매체 접근 제어 계층(Media Access Control Layer)

▪ 물리 계층(Physical Layer)

- PLCP계층(Physical Layer Convergence Procedure)
- PMD계층(Physical Medium Dependent)



3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 프로토콜 계층 구조

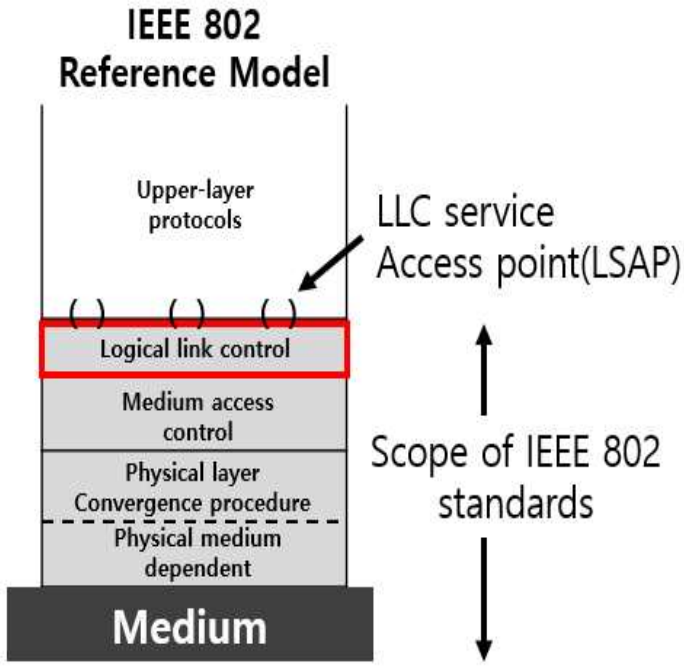
▪ 논리적 링크 제어 계층(Logical Link Control Layer)

- 다양한 프로토콜로부터 같은 네트워크를 사용할 수 있도록 함
- 노드간 흐름 제어 및 에러 제어를 수행

▪ LLC 부계층 헤더 구조

DSAP	SSAP	Control	Data

- DSAP(Destination Service Access Point)
  - 목적지의 주소
- SSAP(Source Service Access Point)
  - 발신지의 주소
- Control
  - 프레임 또는 ACK의 번호



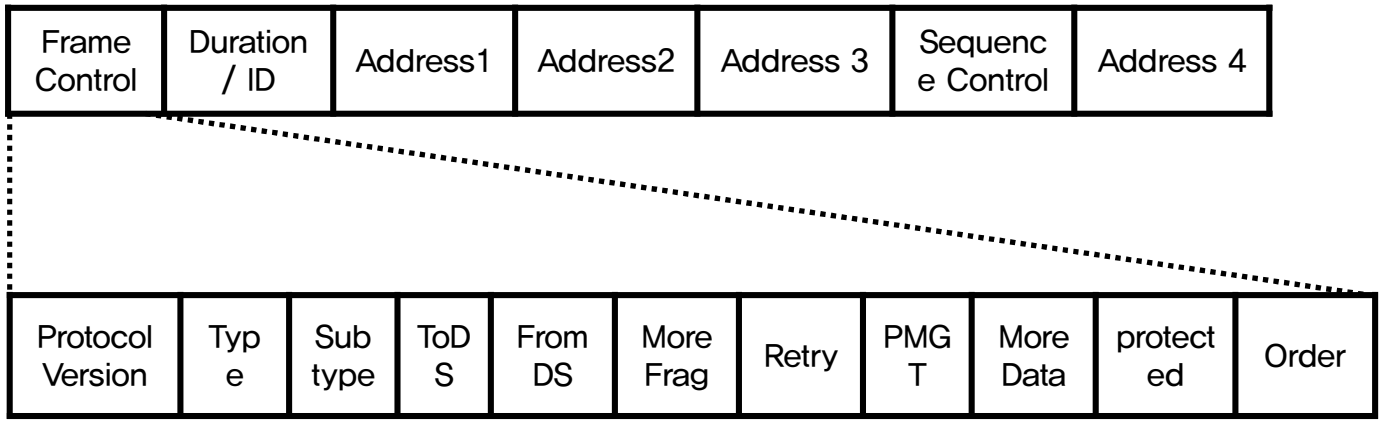
3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 계층별 헤더 구조

▪ 매체 접근 제어 계층(Media Access Control Layer)

- LAN 상에서 단말간 효율적이고, 질서 있는 데이터 전송을 위한 접근 제어 방식을 제공

▪ MAC 헤더 구조



3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 계층별 헤더 구조

▪ MAC 헤더 구조

- Frame Control
  - –Type : 역할에 따라 해당 프레임의 종류(관리 프레임, 제어 프레임,
  - –데이터 프레임) 구분
  - –서브타입 : 관리프레임에 대한 세부적인 프레임 종류 설정
  - –ToDS & FromDS : 프레임에 대한 전송 방향을 결정
  - –More Fragment : 프레임의 분할 여부를 나타냄
  - –Retry : 프레임의 재전송 여부를 표시하고, 중복 송신을 판별
  - –Power Management : 단말의 전원 절약 모드 및 활성 모드 판별
  - –More Data : AP가 단말에게 전달할 프레임이 더 있음을 표시
  - –Protected : 해당 프레임이 암호화 방식에 의해 암호화 유무 표시
  - –Order : 분할된 프레임들을 재조립할 때, 순서에 맞추어 처리 요청

3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 계층별 헤더 구조

▪ MAC 헤더 구조

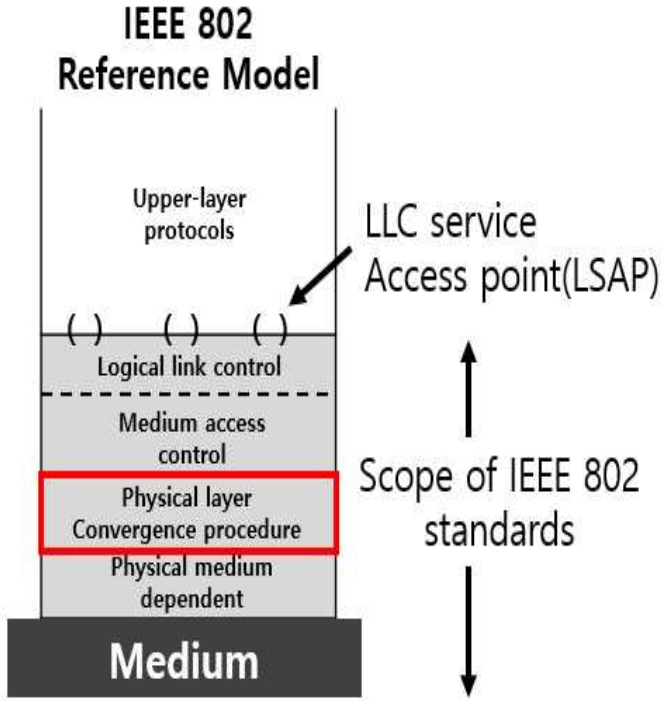
- Duration / ID
  - Duration : 무선 링크의 사용을 예약하여 NAV를 설정
  - AID : 전원 절약 모드의 단말이 주기적으로 깨어나 프레임 수신 요청
- Address 1, 2, 3, 4 (A1, A2, A3, A4)
  - ToDS와 FromDS 필드로 부터 각각의 주소 필드는 프레임 마다 다르게 설정됨
  - 필드에 사용되는 4개의 주소는 최종 목적지 주소, 최초 송신측 주소, 경유 AP 주소, AP의 MAC주소로 분류
- Sequence Control
  - 프레임 별 순서 번호 부여 및 분할된 프레임의 순서 구분

3.1 IEEE 802.11 WLAN

▪ IEEE 802.11 계층별 헤더 구조

- 물리계층은 기술 방식 별로 프레임을 다르게 정의
- 물리계층 DSSS의 PLCP PDU 구조

synchronization	SFD	Signal	Service	length	CRC



- Preamble 영역
  - Synchronization : 동기화를 위한 비트패턴
  - SFD : 실제 프레임의 시작을 알림
- PLCP 헤더 구조
  - Signal : 페이로드 전송속도
  - Service : 고정 클럭 및 변조 방식을 지시
  - Length : MPDU를 송신하는데 필요한 시간
  - CRC : PLCP 헤더의 에러 검출

3.1 IEEE 802.11 WLAN

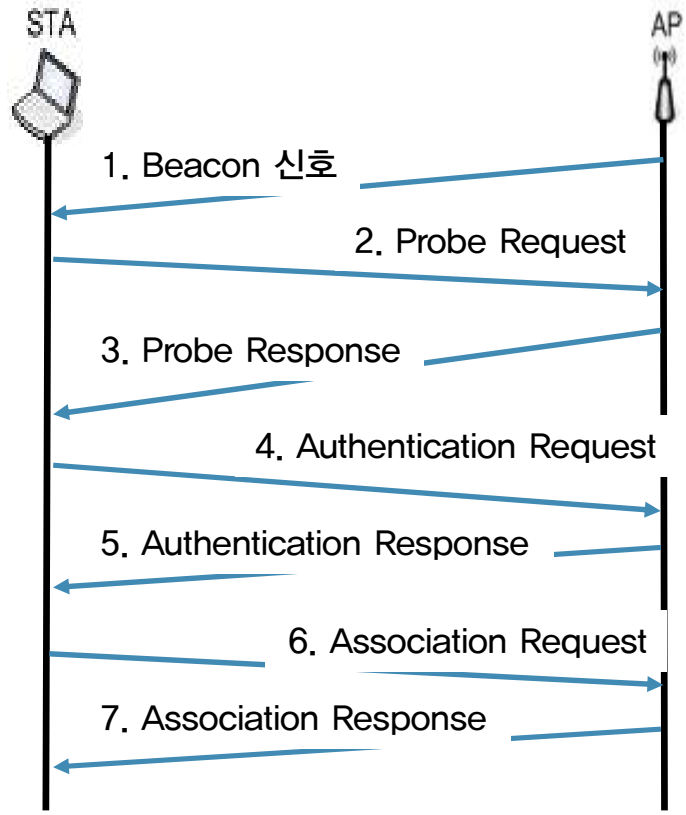
▪ MAC 계층의 연결 절차

- 단말은 MAC 관리 기능부에 의해 연결 절차가 수행됨
- 기능별 단계
  - 탐색 : 비컨 신호 또는 프로브 메시지를 사용하여 AP를 찾는 과정
  - JOIN : 탐색된 AP들 중 적절한 AP를 선택하고 동작값을 추출하는 과정
  - 인증 : AP에 접속하기 위하여, 암호 방식 및 인증 절차를 결정
  - 결합 : AP와 단말 간의 식별 가능한 연결을 설정하는 과정
  - 재결합 : 단말이 다른 AP로 새로운 결합을 설정하는 과정
  - 동기 : 비컨메시지를 참조하여 AP와 단말이 동일한 시각을 설정
  - 전원관리 : 단말이 Sleep 모드시, AP는 해당 단말의 프레임을 임시 저장
  - MIB(Management Information Base) : 물리계층과 MAC계층에서의 동작변수와 설정값들을 저장

3.1 IEEE 802.11 WLAN

▪ MAC 계층의 연결 절차

- **탐색 및 참여 단계**
  - AP로부터 비컨 신호 전송
  - 단말의 프로브 요청 전송
  - AP로부터 프로브 응답 전송
- **인증 단계**
  - 단말의 인증요청 메시지 전송
  - AP로부터 인증 응답 메시지 전송
- **결합 단계**
  - 단말의 연결 요청 메시지 전송
  - AP로부터 연결 응답 메시지 전송

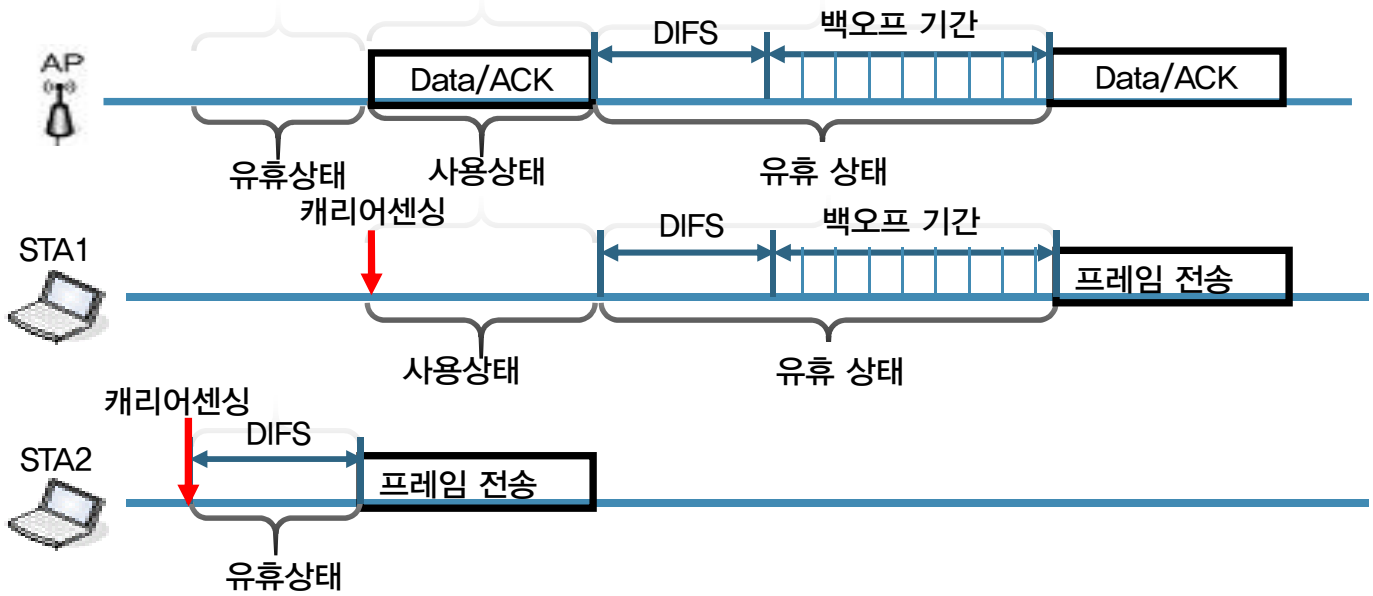


3.1 IEEE 802.11 WLAN

▪ MAC 계층의 데이터 프레임 전송

▪ 백오프 절차에 의한 프레임 전송 동작

- 프레임 송신시 캐리어를 센싱하여 채널이 유휴상태면, DIFS 기간 동안 전송을 지연하며, 여전히 유휴상태이면 프레임을 즉시 전송
- 채널이 사용 중이면, 유휴상태가 될 때 까지 대기후, 추가적으로 DIFS 기간과 백오프 기간 동안 전송을 지연하며 여전히 유휴상태이면 전송

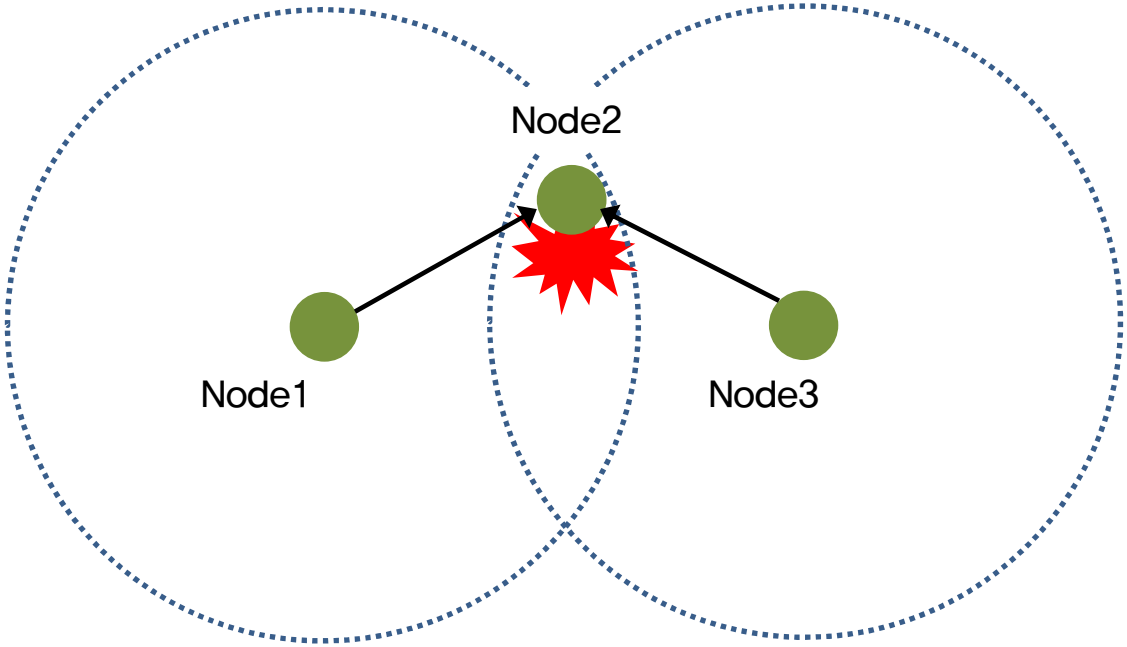


3.1 IEEE 802.11 WLAN

▪ RTS/CTS의 동작

▪ Hidden node problem

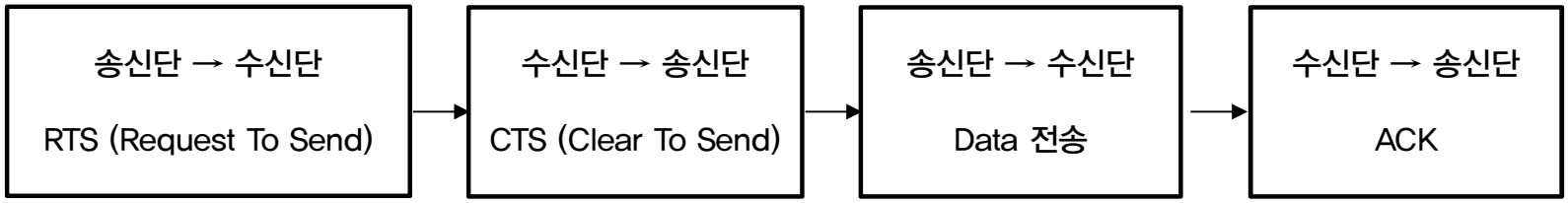
- Node1과 Node3은 통신 범위 밖의 노드로써, 직접적인 연결이 없음
- Node1과 Node3이 Node2로 동시에 데이터 전송할 시, 데이터 충돌
- RTS/CTS 기법을 통해 Hidden node problem 해결



3.1 IEEE 802.11 WLAN

▪ RTS/CTS의 동작

▪ RTS/CTS 기법을 통한 Hidden node problem 해결



- 송신측은 RTS 프레임을 수신측으로 전송
- 수신측이 유휴상태라면, CTS로 응답
- 수신측이 다른 노드와 통신중이라면, 백오프 알고리즘에 의하여 대기
- 수신측의 CTS 프레임이 송신측에 도착하면 즉시 데이터 전송
- 송신측은 데이터를 모두 수신한 뒤에 수신측으로 ACK 프레임을 전송

3.1 IEEE 802.11 WLAN

▪ RTS/CTS의 동작

▪ NAV 예약 및 MAC 필드값

- RTS
  - 데이터 패킷, CTS, ACK 패킷을 전송하기 위해 소요되는 SIFS를 합산하여 duration 필드에 기록하여 AP에게 송신
  - 해당 패킷을 수신한 모든 STA들은 자신의 NAV타이머를 설정하고 이 기간 동안 송신을 지연
- CTS
  - RTS 패킷을 수신한 AP는 CTS로 응답
  - 해당 CTS패킷의 duration 필드값 = RTS패킷의 duration 필드값 – (소비된 SIFS와 CTS 패킷의 전송시간)이며, 이 기간 동안의 채널 점유 시간을 예약
- ACK
  - AP는 수신한 데이터 패킷의 more 비트가 설정되어 있지 않다면, 모든 데이터가 송신 완료 된 것이므로 ACK 패킷의 duration 필드를 0으로 설정함으로써 다른 STA가 전송될 수 있도록 함

3.1 IEEE 802.11 WLAN

▪ WLAN 네트워크 구성 요소

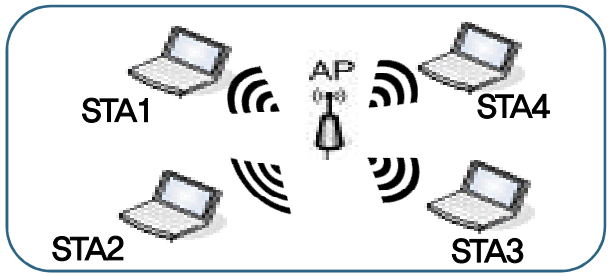
- **무선 링크(Wireless Link)**
  - IEEE 802.11 기술은 ISM(Industrial Scientific Medical) 대역을 사용한다.
- **액세스 포인트(Access Point)**
  - 기지국 역할 및 유무선 연동 브리지 기능을 수행하며 접근점을 제공한다.
- **단말(STA)**
  - IEEE 802.11표준의 MAC과 물리계층을 가지고 동작하는 기기
- **분배 시스템(Distribution System)**
  - 다수의 AP를 연결한 백본망
- **기본 서비스 집합(Basic Service Set)**
  - 단일 AP와 해당 AP에 접속된 단말(STA)로 구성된 그룹
- **확장 서비스 집합 (Extended Service Set)**
  - 확장 서비스 집합 (Extended Service Set)

3.1 IEEE 802.11 WLAN

▪ WLAN 네트워크 구성 요소

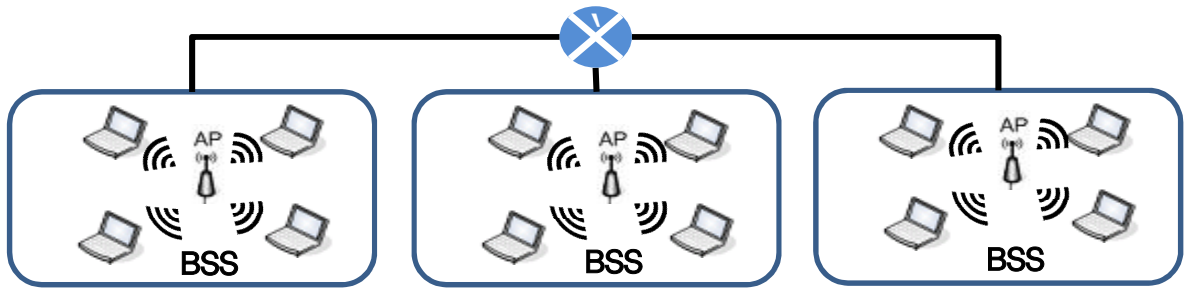
▪ 기본 서비스 집합(Basic Service Set)

- 단일 AP와 해당 AP에 접속된 단말(STA)들로 구성된 그룹



▪ 확장 서비스 집합 (Extended Service Set)

- 여러 개의 BSS와 LAN을 연결하여 하나의 확장 서비스 집합을 생성하고, 서로 다른 BSS간의 메시지 전송서비스를 제공하는 시스템



3.2 BLE(Bluetooth Low Energy)

▪ BLE 개념

- BLE(Bluetooth Low Energy)는 기존 Bluetooth 4.0 버전에 Low Energy 기술을 탑재된 기술로 Bluetooth smart라고도 지칭함
- Bluetooth는 크게 Bluetooth Classic, Bluetooth smart로 분류됨
  - Bluetooth (Bluetooth Classic)
  - Bluetooth Smart (Bluetooth Smart)
  - Bluetooth Smart Ready(Bluetooth Classic , Bluetooth Smart)

- 기존 Master, Slave 방식을 사용하는 Bluetooth를 Bluetooth Classic이라고 하며 이를 경량화하기 위한 노력으로 Bluetooth 4.0 개발
- Bluetooth Classic의 최대 단점인 과도한 배터리 소모 문제를 해결하기 위해 새로운 표준 채택(Bluetooth 4.0)

3.2 BLE(Bluetooth Low Energy)

▪ GAP(Generic Access Profile)

- 서로 다른 제조사가 만든 BLE 디바이스들끼리 서로 호환되어 통신할 수 있도록 함.
- 디바이스간의 Advertising 과 Connection에 대한 프레임워크를 제공
- BLE 통신을 위해 Role, Mode, Procedure, Security, Additional GAP Data Format 등을 정의

3.2 BLE(Bluetooth Low Energy)

▪ BLE 통신 모드

▪ Connection Mode는 1:1 방식으로 데이터를 교환하는 모드로 디바이스의 역할을 Central 과 Peripheral 로 나눈다.

▪ Central(Master)

- 충분한 전원 및 리소스를 갖춘 디바이스
- 주기적으로 Connectable Advertising Signal 스캔

▪ Peripheral(Slave)

- 저전력으로 동작하며 리소스가 제한된 디바이스
- Central 디바이스에 연결되어 동작하는 역할
- 주기적으로 Connectable Advertising Signal 송신

3.2 BLE(Bluetooth Low Energy)

▪ BLE 통신 모드

- Advertise Mode(= Broadcast Mode)는 1:N 방식으로 데이터를 교환하는 방식으로 디바이스의 역할을 Advertiser (= Broadcaster) 와 Observer로 나눈다.

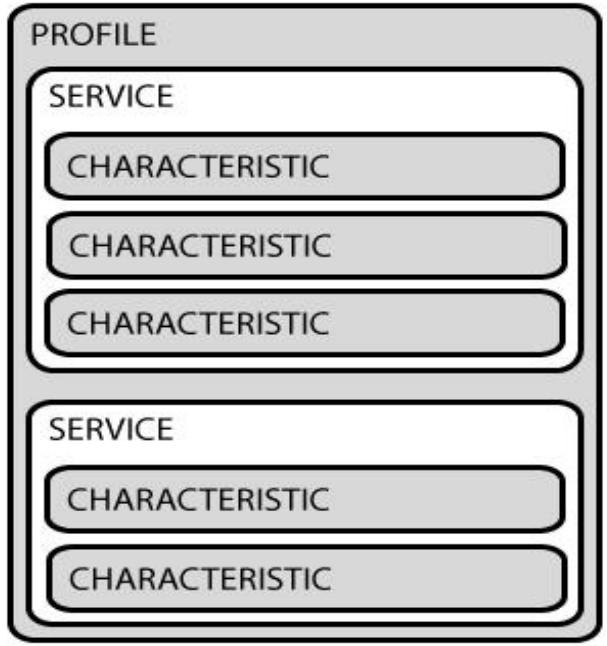
- Advertiser
  - 주기적으로 Non-Connectable Advertising Packet을 보내는 디바이스

- Observer
  - Advertiser가 보내는 Non-Connectable Advertising Packet를 듣기 위해 주기적으로 스캔하는 디바이스

3.2 BLE(Bluetooth Low Energy)

▪ GATT (Generic Attribute Profile) (1/2)

- BLE Data 교환을 관리하며 디바이스들이 Data를 발견하고, 읽고, 쓰도록 기초적인 Data Model과 Procedure를 정의
- 동작 구조는 프로파일(Profile), 서비스(Service), 특성(Characteristic) 에 기초함



3.2 BLE(Bluetooth Low Energy)

▪ GATT (Generic Attribute Profile) (2/2)

▪ Profile

- 센서 디바이스에 의해 미리 정의 된 서비스 묶음

▪ Service

- Data를 논리적인 단위로 나누는 역할
- 하나 이상의 Characteristic를 가짐
- UUID라는 16/128 bit 구분자를 가짐
- 특성(Heart Rate Measurement, Body Sensor Location, Heart Rate Control Point)을 가짐

▪ Characteristic

- 하나의 Data를 가짐
- 최하위 단위 특성
- UUID라는 16/128 bit 구분자를 가짐
- 특성(Heart Rate Measurement, Body Sensor Location, Heart Rate Control Point)을 가짐

3.2 BLE(Bluetooth Low Energy)

▪ Bluetooth 5

- Bluetooth 4.2보다 송수신 거리 최대 4배, 속도 최대 2배 , 브로드캐스팅 메시지 용량 최대 8배등 기능 향상
- Bluetooth 4.0에서 처음 도입된 Low Energy Bluetooth 기능은 Bluetooth 5.0에서도 제공
- Low Energy 기능만을 제공하는 Bluetooth 5는 Bluetooth 4.x(4.0, 4.1, 4.2)와 역방향 호환 가능
- 2.4GHz ISM 대역폭의 경계와 주변 LTE 대역폭에서 간섭을 감지하고 방지

3.2 BLE(Bluetooth Low Energy)

▪ Bluetooth 버전 소개

표 준 안	특 징	구분
Bluetooth 1.X	• 초창기 Bluetooth로 최대 전송속도 723kbps	• Classic Bluetooth • 기존 Bluetooth 기술
Bluetooth 2.X	• 최대 전송속도 3Mbps • SSP(Secure Simple Pairing) 기능 추가 • EIR(Extended Inquiry Response)를 통한 커넥션 필터링강화	
Bluetooth 3.X	• 최대 전송속도 24Mbps • 3.0+HS(High Speed)만 최대 전송속도 지원 • Bluetooth Link는 접속에만 사용 • PAL(Protocol Adaptation Layer)을 통한 Wi-Fi 고속 전송	• Bluetooth High Speed • Wi-Fi를 활용한 고속기술 중점
Bluetooth 4.X	• 전력소모 최소화, 배터리 수명 연장 • 사물인터넷 연결성 강화 • 개인정보보호 강화	• Bluetooth Low Energy • 전력소모의 소화 및 배터리 수명 중점
Bluetooth 5.X	• 버전4 에 비해 송수신거리, 속도, 브로드 캐스팅 용량 향상 • 사물인터넷 기능 증가에 집중	

Assignment

- Assignment 3

사물인터넷에서 활용될 수 있는 통신 프로토콜 별 장단점을 비교하고
각각 적합한 네트워크 환경을 조사하시오.

Reference

[3.2] BLE(BLE(Bluetooth Low Energy)

GATT (Generic Attribute Profile)

– 그림 출처 : https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gatt

4장. 사물인터넷 네트워크 II

Chapter 4. Internet of Things Network II

4.1 ZigBee
4.2 LPWA
4.3 IPv6
4.4 6LoWPAN

- Assignment
- Reference

강의 개요

▪ 강의 목표

다양한 사물 통신 프로토콜의 의 개념과 특징을 학습하고

공통점 및 차이점을 파악한다.

▪ 강의 내용

- IEEE 802.15.4 및 ZigBee 표준 기술
- LPWA 기술 개념 및 동작 구조
- IPv6 개념 및 특징
- 6LoWPAN 개념 및 특징

4.1 IEEE 802.15.4 및 ZigBee 표준기술 개요

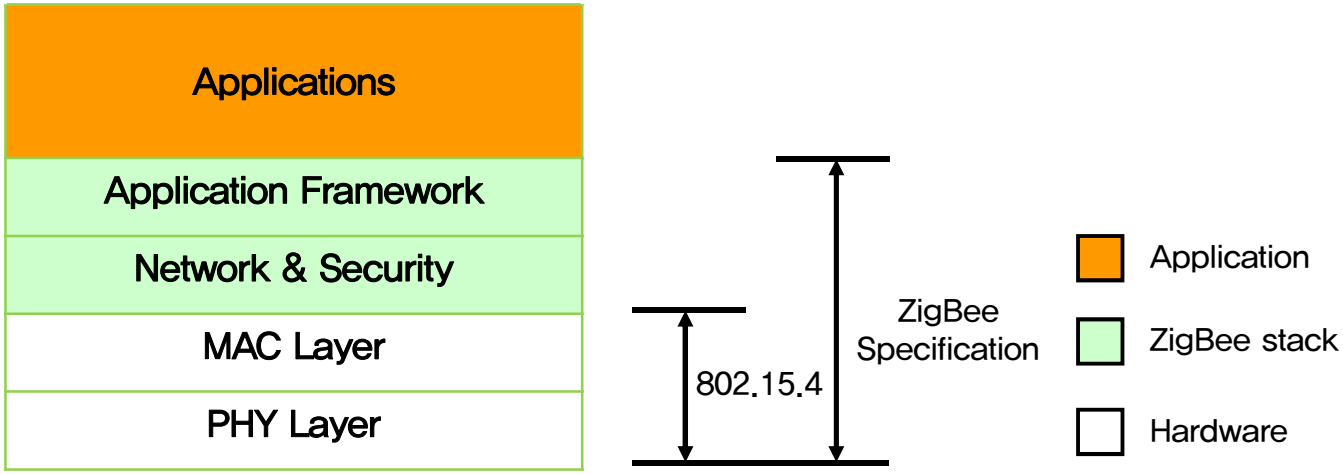
▪ ZigBee 계층 구조

▪ ZigBee 계층 구조

- 저전력, 장치간의 저렴한 가격, 저속도 통신을 지향하는 물리 계층과 MAC 계층을 정의하는 표준

▪ 지그비(Zigbee) 정의

- ZigBee Alliance에서 IEEE 802.15.4 표준 기술을 기반으로 상위 계층을 포괄하여 정의된 표준 기술
- 저렴한 가격, 저전력, 무선 메쉬 네트워크를 지향하는 표준기술



4.1 IEEE 802.15.4 및 ZigBee 표준기술 개요

▪ ZigBee와 IEEE 802.15.4 특징

▪ ZigBee와 IEEE 802.15.4 특징

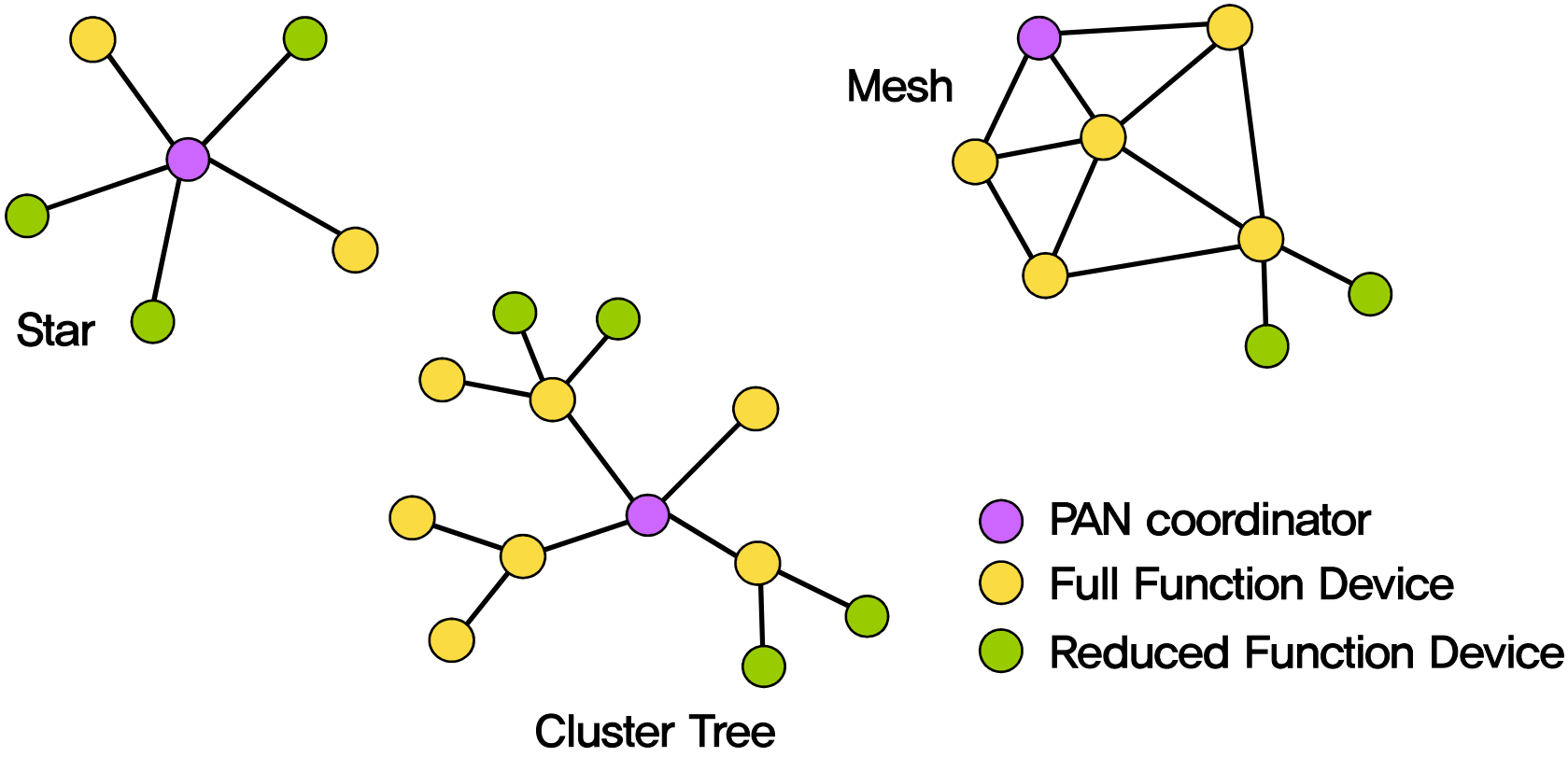
- ZigBee는 IEEE 802.15.4 표준 기술을 준수하여 안정적, 효율적인 물리 계층을 활용
- IEEE 802.15.4 표준 기술에 상위 계층인 논리적 네트워크, 보안기술, 응용 계층을 덧붙인 표준

▪ ZigBee 특징

- 센서 네트워크
  - 저전력, 저가격, 낮은 데이터 전송률, 다수의 네트워크 노드 지원
- 멀티 홉 지원
  - 스타, 클러스터 트리, 메쉬 토폴로지를 지원하며 데이터 전송 성공률이 높음
- 상황 인지 기능
  - 네트워크 상황 인지 및 위치기반 서비스 제공
- 칩 개발의 용이성
  - 프로토콜 구조가 단순하여 확장성을 가짐

4.1 IEEE 802.15.4 및 ZigBee 표준기술 개요

- ZigBee 네트워크 토폴로지
  - 스타 토폴로지, 메쉬 토폴로지, 클러스터 트리 지원



4.1 IEEE 802.15.4 및 ZigBee 표준기술 개요

▪ ZigBee 네트워크 구성 요소

▪ Coordinator (Full Function Device)

- 슈퍼프레임 비컨 전송
- ZigBee 네트워크 설정
- 네트워크 노드들을 관리

▪ Router (Full Function Device)

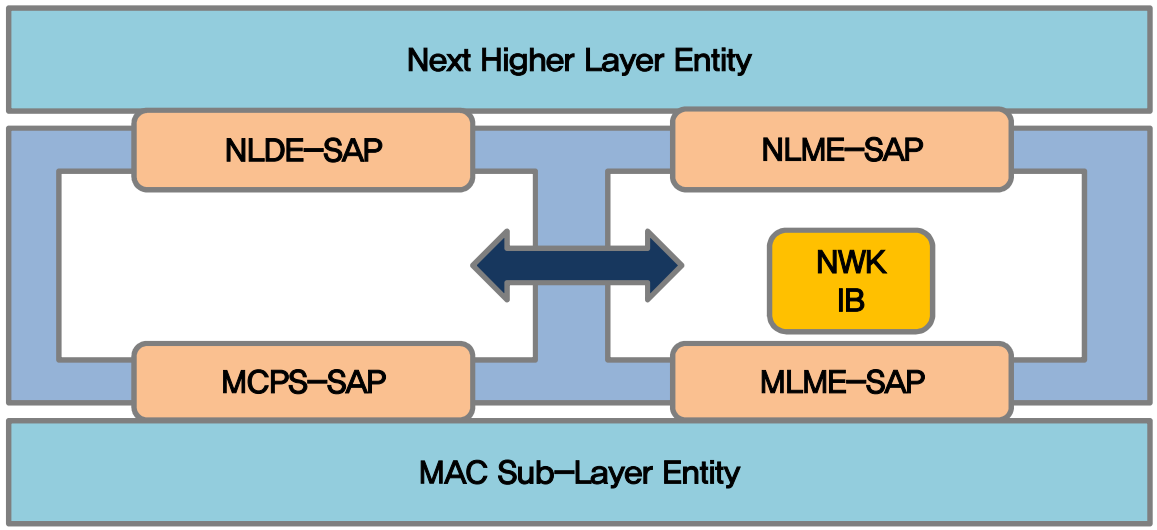
- 네트워크의 모든 구성요소와 통신하며, 패킷 중계
- 멀티홉 네트워킹 지원
- 네트워크 Coordinator 동작 지원

▪ End Device (Reduced Function Device)

- 제한된 저전력 장치들을 지칭
- 네트워크 Coordinator와만 통신

4.1 IEEE 802.15.4 및 ZigBee 표준기술 개요

- ZigBee 네트워크 구조
  - NLDE : Network Layer Data Entity
  - NLME : Network Layer Management Entity
  - NLDE-SAP : NLDE – Service Access Point
  - NLME-SAP : NLME – Service Access Point
  - NWK IB : Network information base



4.1 IEEE 802.15.4 및 ZigBee 표준기술 개요

ZigBee 네트워크 구성 요소

네트워크 계층 Data Entity(NLDE)

- 네트워크 계층의 PDU(NPDU) 생성
- 토폴로지에 따라 NPDU를 라우팅

네트워크 계층 Management Entity(NLME)

- 새로운 디바이스의 설정
- 네트워크의 동작 시작
- 디바이스의 네트워크 참여 및 이탈
- 주소 할당
- 이웃 노드 탐색
- 라우팅 경로 탐색

NWK information base(NIB)

- 디바이스의 네트워크 계층 관리에 필요한 정보 유지

4.1 IEEE 802.15.4 및 ZigBee 표준기술 개요

▪ ZigBee 프레임 구조

▪ 일반적인 NPDU프레임의 구조

Octets : 2	2	2	0/1	0/1	Variable
Frame Control	Destination Address	Source Address	Broadcast Radius	Broadcast Sequence Number	Frame Payload
	Routing Fields				
NWK Header					NWK Payload

▪ 프레임 Control 필드

Bits : 0–1	2–5	6	7–8	9	10–15
Frame type	Protocol version	Discover route	Reserved	Security	Reserved

4.1 IEEE 802.15.4 및 ZigBee 표준기술 개요

▪ ZigBee 프레임 구조

▪ Data Frame format

Octets : 2	See Figure 7	Variable
Frame control	Routing fields	Data payload
NWK header		NWK payload

▪ NWK command frame forma

Octets : 2	See Figure 7	1	Variable
Frame control	Routing fields	NWK command identifier	NWK command payload
NWK header		NWK payload	

▪ NWK command frames

Command Frame identifier	Command name	Reference
0x01	Route request	8.1
0x02	Route reply	8.2
0x03	Route Error	8.3
0x00. 0x04 – 0xff	Reserved	–

4.1 IEEE 802.15.4 및 ZigBee 표준기술 개요

ZigBee 서비스 프리미티브

Confirmed service

- 요청(Request) : 상위 계층으로부터 해당 기능 요청
- 확인(Confirm) : 요청 받으면 해당 기능을 수행 후 확인 리턴
- 응답(Response) : 이벤트에 대한 응답
- 통지(Indication) : 특정 이벤트 발생시 상위 계층에 통지

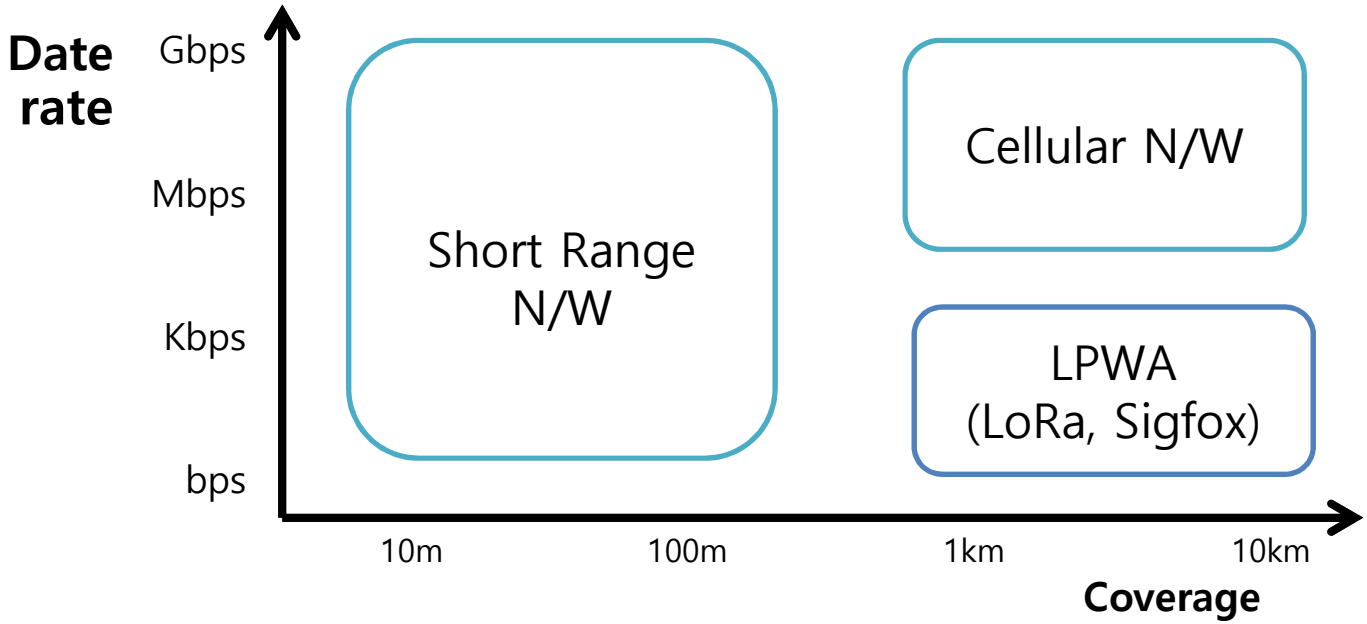
Unconfirmed service

- 별도의 응답 필요하지 않음

4.2 LPWA

▪ LPWA 개념

- LPWA(Low Power Wide Area)는 저전력으로 소량의 데이터를 멀리 보내는데 사용되는 기술
- 저전력 소모 설계, 저가 단말기 공급, 낮은 구축비용, 안정적 커버리지 제공, 대규모의 단말 접속 구현을 핵심 요구사항으로 충족해야 한다.



4.2 LPWA

▪ LPWA 필요성

- **사물인터넷 접속 기술은 Short Range Network에 적합하지 않다.**
  - Short Range Network에는 대표적으로 Wi-Fi, Bluetooth가 있다.

- **Bluetooth의 경우 근거리 접속만 가능하며 기기간 통신시 안전성이 떨어짐**

- **Wi-Fi의 경우 AP가 존재하는 곳에서만 접속이 가능하며 접속 범위가 AP 근방으로 제한되어 있다.**

- **사물인터넷에서는 시공간의 제약을 없애기 위해 Short Range Network 기술에서의 단점을 보완한 저전력 장거리(LPWA) 통신이 제안됨**

4.2 LPWA(LoRaWAN)

▪ LoRaWAN 개념 및 특징

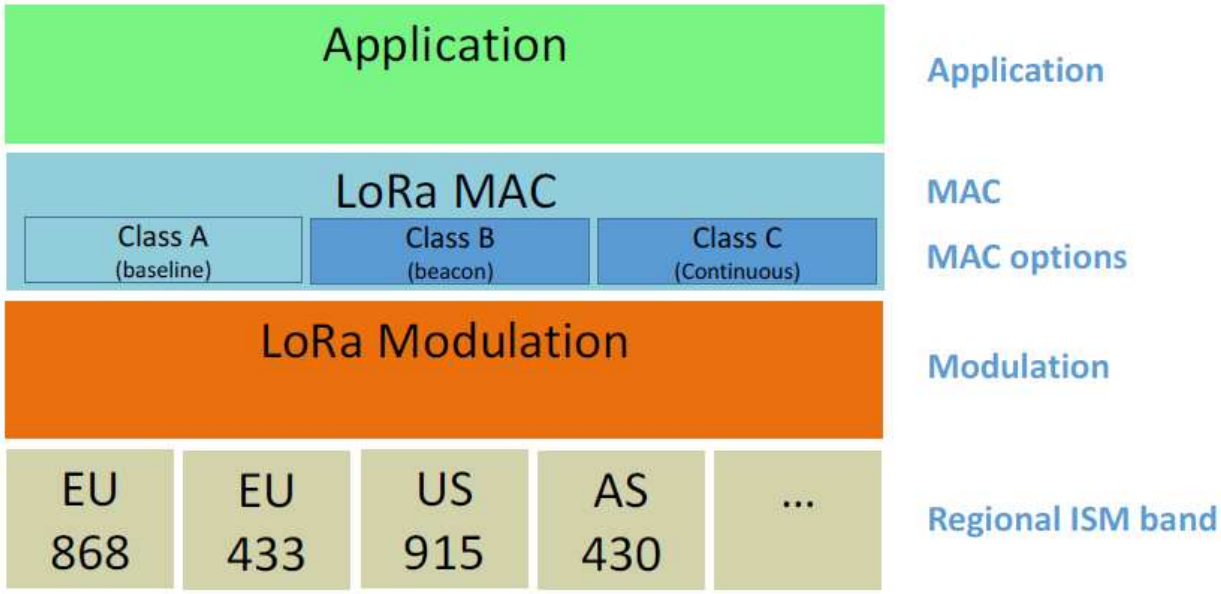
- LoRaWAN(Long Range Wide-Area Network)는 LPWA 기술 중 하나로 다국적 연합체인 LoRa Alliance에서 기술 개발을 주도
- A/B/C타입의 디바이스를 제공하여 상황에 적합한 타입 선택 가능
- ADR (adaptive Data Rate) 기술을 사용하여 배터리 보존 및 데이터 전송률 향상
- Channel Hopping ,Duty Cycle limited

4.2 LPWA(LoRaWAN)

▪ LoRaWAN 클래스

▪ 최적화 타입에 따른 디바이스 클래스를 지정해서 사용

- Class A : 소모전력 최적화
- Class B : 낮은 지연속도
- Class C : 지연속도 최적화 (소모전력이 높음)

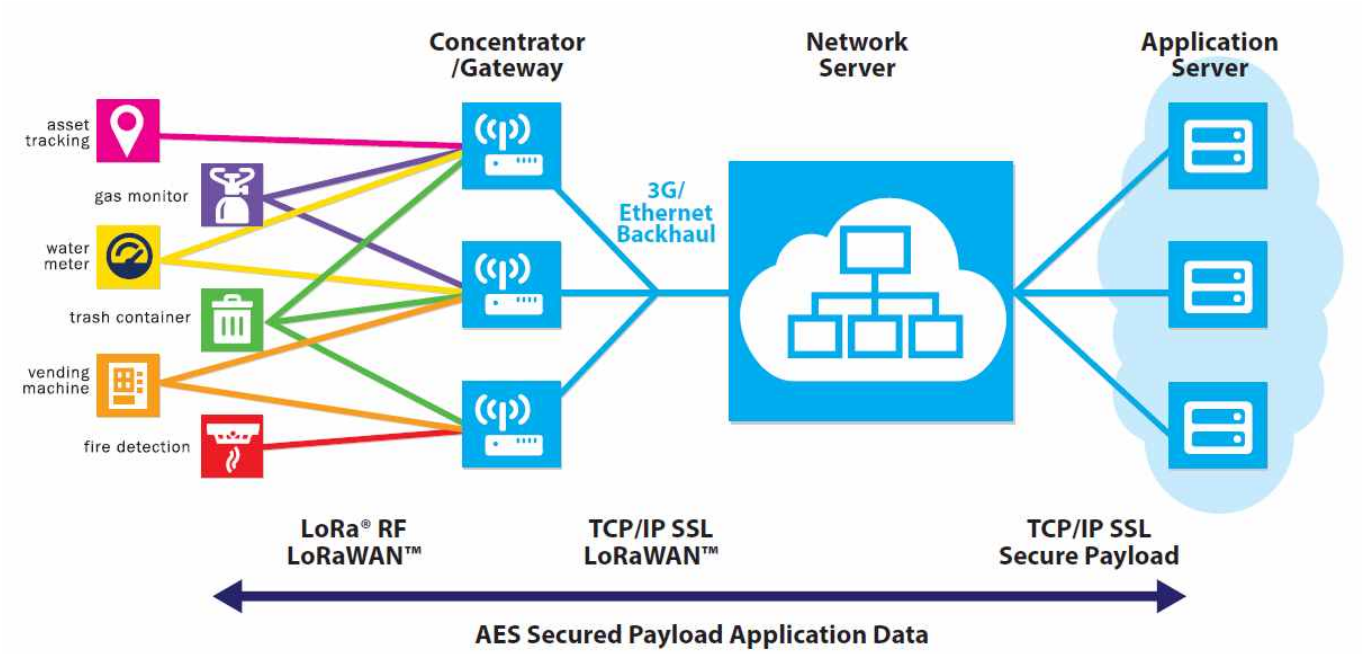


4.2 LPWA(LoRaWAN)

▪ LoRaWAN 토폴로지 구성

▪ LoRaWAN 토폴로지 구성

- 최적의 경로로 네트워크에서 디바이스로 데이터 전송
- 디바이스의 Hand-Over 과정 불필요

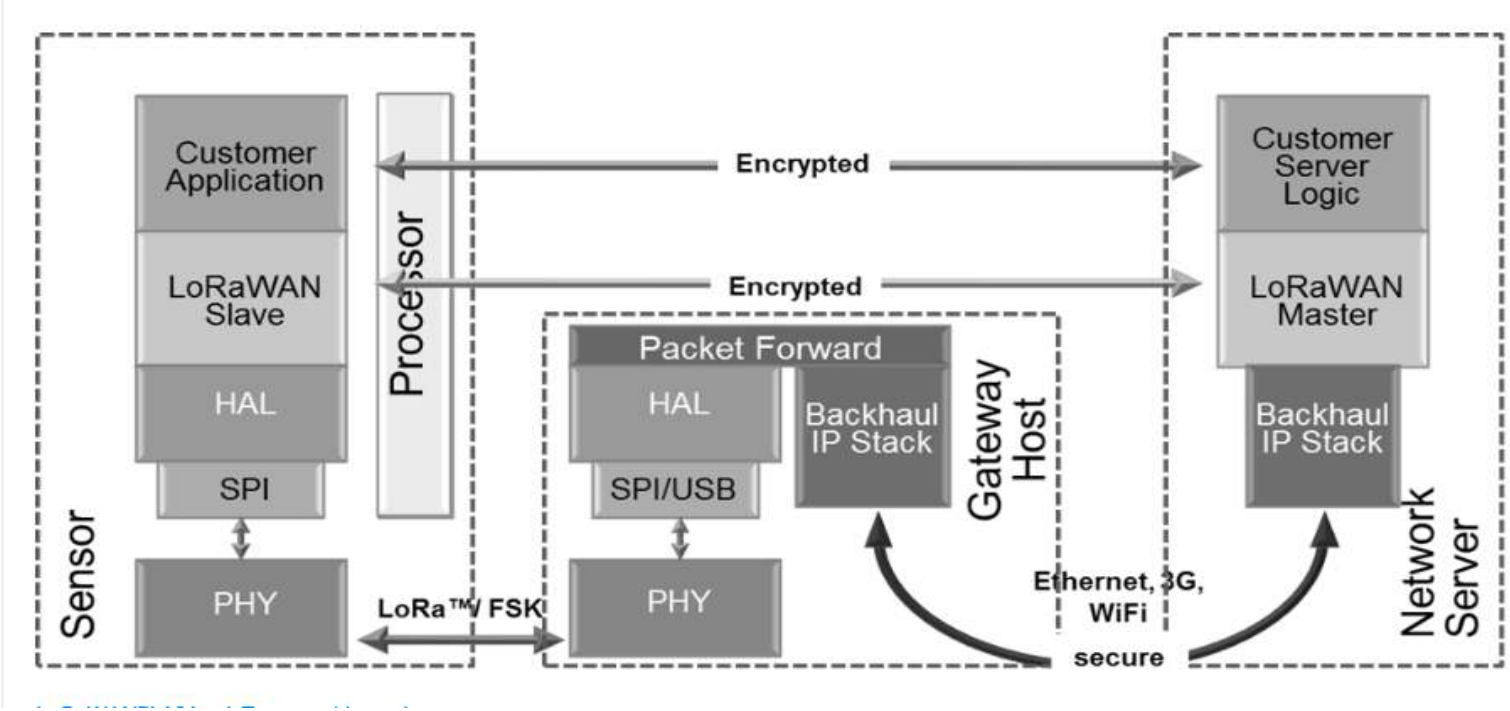


4.2 LPWA(LoRaWAN)

▪ LoRaWAN 프로토콜 스택

▪ 게이트웨이는 Forwarding 역할만 하여 단말과 네트워크 서버의 MAC 계층 메시지 처리

- 단순한 접속 절차로 대량의 디바이스 수용 용의



4.2 LPWA(LoRaWAN)

▪ LoRaWAN 프로토콜 스택

- ADR(Adaptive Data Rate) 기술은 PER(Packet Error Rate) 및 SNR(signal-to-noise ratio), RSSI(received signal strength indicator)값을 분석하여 디바이스 별로 Date Rate 과 송신 파워를 조절하는 기법

- 데이터 전송률 향상 및 배터리 보존을 동시에 충족가능

- 신호 간섭에 강하며 최적의 주파수를 활용 할 수 있음

4.2 LPWA(LoRaWAN)

LoRaWAN 식별 체계

- LoRa RF는 IP 식별 체계와 다름
- AppEUI
  - 글로벌 어플리케이션 ID
- DevEUI
  - 글로벌 디바이스 ID
- DevAddr
  - Lora Alliance(7bit), 네트워크 서버(25bit)를 할당해주는 네트워크 주소
- key
  - AppKey : 사전에 디바이스와 네트워크 서버가 공유된 키
  - Network Session Key , Application Session Key : Appkey를 기반으로 생성된 Session Key

4.2 LPWA(LoRaWAN)

- LoRaWAN 메시지 포맷

Radio PHY layer:

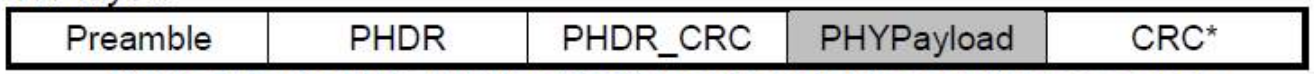


Figure 5: Radio PHY structure (CRC* is only available on uplink messages)

PHYPayload:

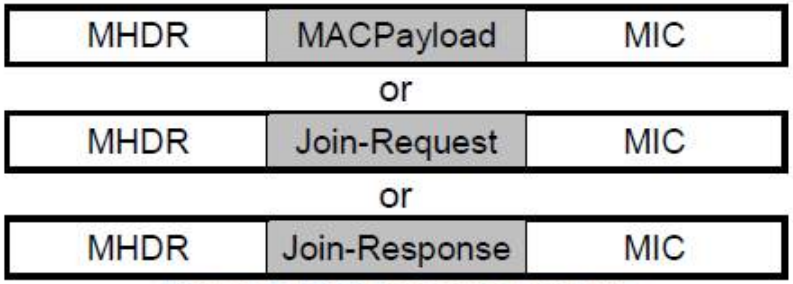


Figure 6: PHY payload structure

MACPayload:

Figure 7: MAC payload structure

FHDR:

Figure 8: Frame header structure

4.2 LPWA(SigFox)

- SigFox 개념
  - SigFox는 프랑스의 SIGFOX사에서 주도하는 저전력 장거리 통신 서비스
  - ISM Band 사용, 디바이스당 연간 1유로
  - 디바이스가 전송한 모든 데이터는 SigFox Cloud에 저장되며, 서비스 주체는 Open API를 이용하여 데이터 조회

4.2 LPWA(SigFox)

- SigFox 기술 특징
  - 양방향 통신 , 메시지 당 최대 12byte 전송 가능
  - 시간당 6개의 메시지, 하루 최대 140 메시지 전송 가능
  - RF Band Width
    - DownLink : 200khz
    - UpLink : 200khz
  - Data Rate
    - DownLink : 600bps
    - UpLink : 100bpsy

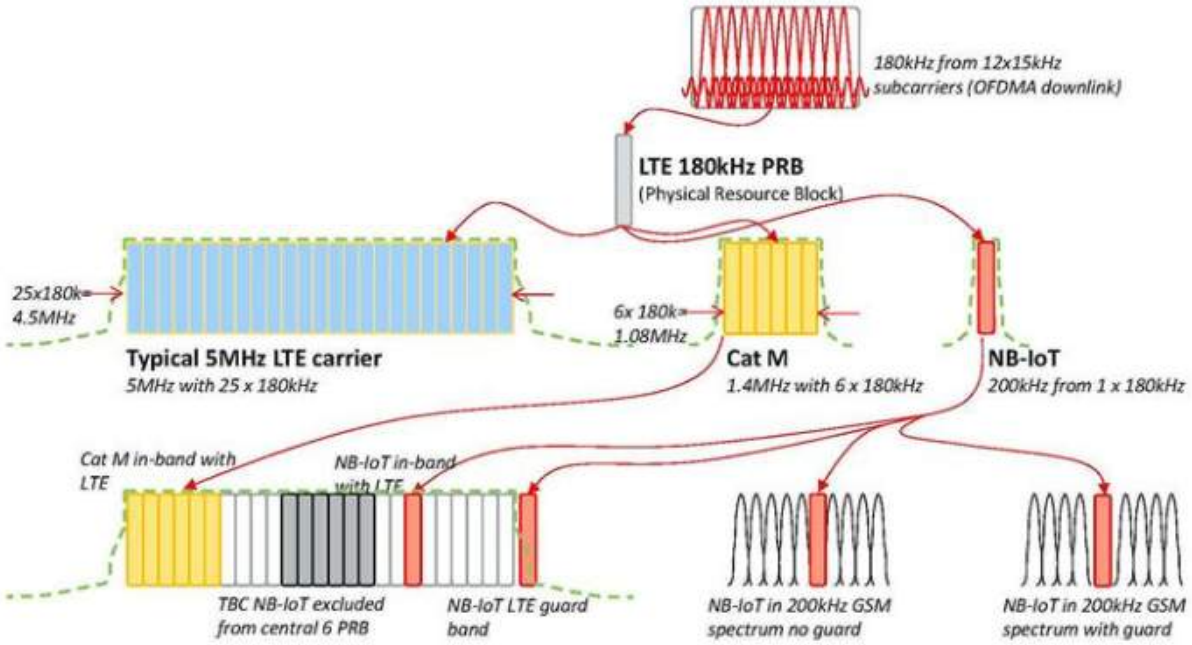
4.2 LPWA(NB-IoT)

▪ NB-IoT 개념 및 특징

- NB-IoT(Narrow Band IoT, 협대역 사물인터넷)은 LTE, 3G등 기존 이동통신 방식보다 좁은 200 kHz의 대역폭을 이용하는 LPWA 기술
- 사물 간의 소량 데이터 통신에 특화된 원거리 저전력 통신 기술로 사물인터넷 서비스에 적합
- NB-IoT는 3GPP에서 Release 13 core part 표준화가 16년 6월 완료되었으며, 17년 3월 Release 14 core part 및 NB-IoT Enhance 표준화 완료를 목표로 함
- 면허 대역을 사용하며 동작 모드에 따라 다양한 대역 지원
- 구축된 LTE 네트워크를 그대로 활용 할 수 있어 원활한 기존 트래픽 처리 가능

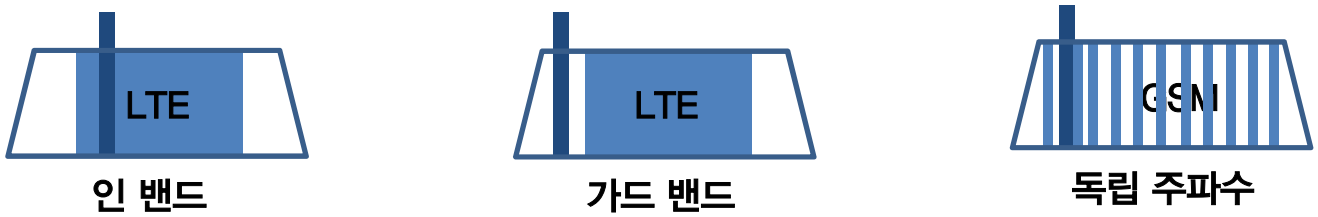
4.2 LPWA(NB-IoT)

- NB-IoT 표준
  - LTEm은 Cat. 1, Cat. 0, Cat. M 및 NB-IoT로 분류
  - Release 13은 LTE-M과 NB LTE-M 두가지 표준을 포함



4.2 LPWA(NB-IoT)

- NB-IoT 동작 모드
  - 인 밴드, 가드 밴드, 독립 주파수의 세가지 모드 지원
  - 인 밴드 모드는 LTE 대역 내 자원 중 일부를 NB-IoT에 할당하여 운용
  - 가드 밴드 모드는 LTE의 보호 주파수 대역을 활용하며, NB-IoT 캐리어는 LTE의 가장자리 부반송파에 되도록 가깝게 배치
  - 독립 주파수 모드는 GSM 대역 내 일부 캐리어를 별도로 할당하여 운용



4.2 LPWA(NB-IoT)

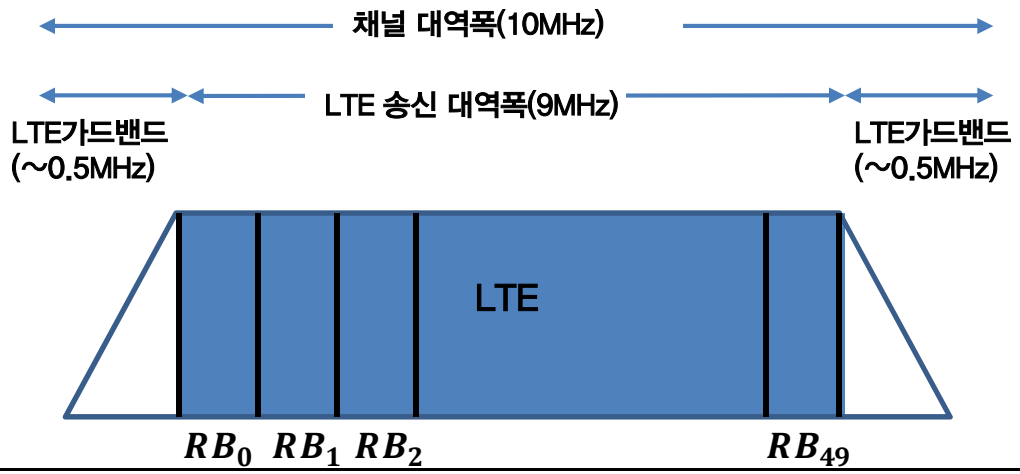
▪ NB-IoT 인 밴드 주파수

▪ 다운링크

- LTE와 동일한 RB 사용
- 앵커 캐리어의 중심 주파수는 100kHz 정수배 근처에 위치

▪ 업링크

- LTE와 동일한 RB 사용
- 일반적으로 RB #0~3, #47~49는 제어 채널(PUCCH)로 사용



4.2 LPWA(NB-IoT)

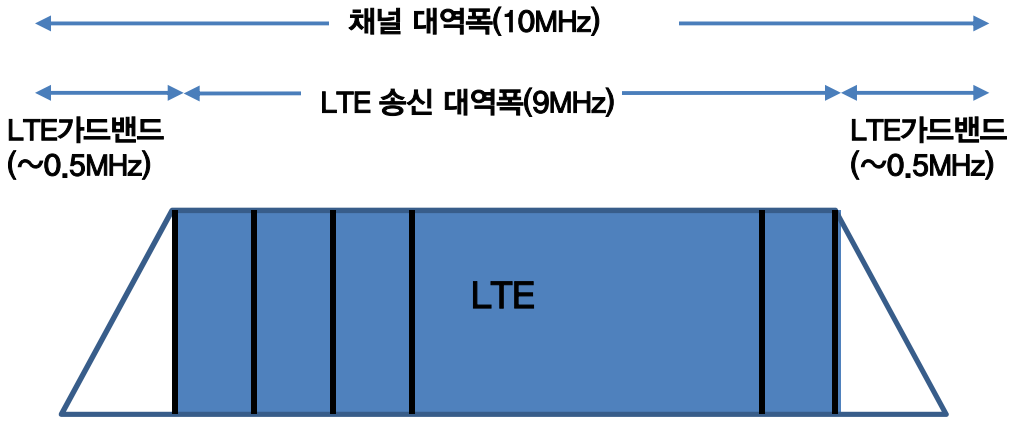
▪ NB-IoT 인 밴드 주파수

▪ 다운링크

- LTE와 부송파(subcarrier)간격이 15kHz의 정수배
- 앵커 캐리어의 중심 주파수는 100kHz 정수배 근처에 위치

▪ 업링크

- LTE와 부송파(subcarrier) 간격이 16kHz의 정수배
- NB-IoT 단말은 LTE와의 상호 공존을 위해, NB-IoT 대역 경계로부터 특정 오프셋 주파수만큼 떨어진 위치에서 LTE의 대역외 발사(Out of band emission)규격을 만족 해야함



4.3 IPv6

▪ IPv6 개념

- IPv6(Internet Protocol version 6)은 기존 인터넷(Internet)이 IPv4 프로토콜로 구축되어 왔으나 IPv4 프로토콜의 한계점으로 인해 지속적인 인터넷 문제가 예상되어 이에 대한 대안으로 제정됨

- IPv4 주소는 빠른 속도로 고갈되고 있으며 주소를 더 많은 네트워크에 할당하기 위해 네트워크 프래그멘테이션(Network Fragmentation)은 지속적으로 증가하여 라우터에 많은 부담을 주고 있다. 이러한 문제를 해결하고자 IPv6가 제안됨

- IPv4 주소 개수 2^{32} = 4,294,967,296

- IPv6 주소 개수 2^{128} = 340,282,366,920,938,463,463,374,607,431,768,211,456

4.3 IPv6

▪ IPv6 특징 (1/2)

▪ IP 주소의 확장

- 기존 IPv4의 32비트와 달리 IPv6는 128비트의 주소 길이를 제공

▪ 호스트 주소 자동 설정

- 호스트 주소 자동 설정

▪ 패킷 크기 확장

- IPv4에서 패킷 크기가 제한 되어 있지만 IPv6는 점보그램 옵션을 사용해 특정 호스트사이의 임의의 큰 패킷을 주고 받을 수 있도록 제한이 없앰

▪ 효율적인 라우팅

- IP 패킷 처리를 신속하게 할 수 있도록 고정된 단순헤더를 사용함과 동시에 확장헤더를 통해 옵션기능의 확장이 용이한 구조로 정의

4.3 IPv6

▪ IPv6 특징 (2/2)

▪ 플로우 레이블링(Flow Labeling)

- 특정 트래픽은 별도의 처리를 통해 높은 품질의 서비스를 제공

▪ 인증 및 보안기능

- 패킷 인증과 데이터 무결성 및 비밀보장 기능을 IP 프로토콜체계에 반영, 확장헤더를 통해 적용 가능

▪ 이동성

- IPv6 호스트는 네트워크의 물리적 위치에 제한 받지 않고 같은 주소를 유지하면서도 자유롭게 이동 가능

4.3 IPv6

▪ IPv6 주소 표현

- IPv6의 128비트 주소공간은 다음과 같이 16비트(2옥텟)를 16진수로 표현하여 8자리로 나타냄

• IPv6의 128비트 주소공간은 다음과 같이 16비트(2옥텟)를 16진수로 표현하여 8자리로 나타냄

- 대부분의 자리가 0의 숫자를 갖게 되므로, 0000을 하나의 0으로 생략하거나, 혹은 연속되는 0의 그룹을 없애고 ':' 으로 생략 가능

• Ex) 2001:0DB8::1428:57ab

- 0을 생략하고 ': ' 을 없애는 규칙은 두 번 이상 적용 불가

4.3 IPv6

▪ IPv6 주소 종류

▪ 유니캐스트(Unicast)

- 단일 인터페이스를 위한 식별자로 유니캐스트 주소로 보내진 패킷을 해당 주소에 의해 식별된 인터페이스로 전달
- 주소 공간 : fc00::/7

▪ 애니 캐스트(Anycast)

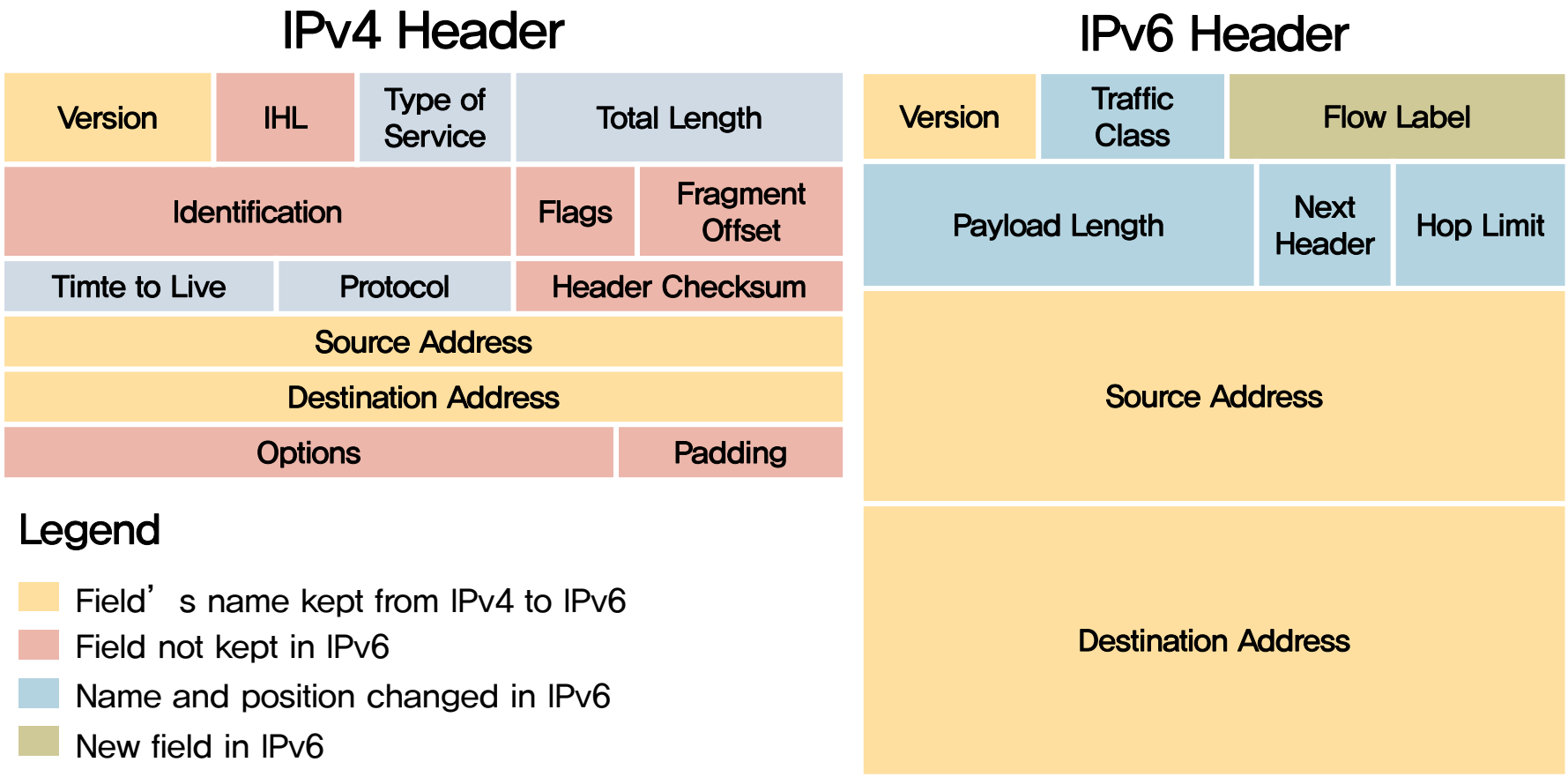
- 하나이상의 인터페이스에 할당되는 주소로서 애니캐스트 주소로 보내진 패킷은 해당 애니캐스트 주소를 가장 가까운 인터페이스로 전달
- 주소 형식상 유니캐스트와 구분 불가능

▪ 멀티 캐스트(Multicast)

- 인터페이스들의 집합을 위한 식별자로서 멀티 캐스트 주소로 보내진 패킷은 해당 주소에 의해 식별된 모든 인터페이스로 전달
- IPv4주소의 브로드 캐스트 주소를 대체 (IPv6는 브로트 캐스트가 없음)
- 주소 공간 : ff00::/8

4.3 IPv6

- IPv6 주소 헤더 구조



4.4 6LoWPAN

▪ 6LoWPAN 개념

- 6LoWPAN(IPv6 over Low-Power Wireless Personal Area Network)은 IETF의 인터넷 영역에 대한 워킹 그룹이며 IEEE 802.15 기반 네트워크를 사용해 통신을 할 수 있도록 IPv6 패킷을 허용하는 헤더 압축 메커니즘을 목적으로 두고 있다

- 6LoWPAN은 Adaption Layer에 해당하며 저전력 무선 네트워크로 IEEE 802.15.4 MAC 계층과 IPv6계층 사이에 구성된다.

- Adaption Layer의 주요 기능은 헤더 압축(Header compression), 패킷 단편화(Packet fragmentation), 계층-2 포워딩(Layer-two forwarding) 등이 있다

4.4 6LoWPAN

▪ 6LoWPAN 특징 (1/2)

- 저전력 및 250kpbs의 적은 대역폭
- Star, Mesh Topology 지원
- 센서 노드가 능동적으로 외부 IP 네트워크와 통신 수행 가능
- WSN(Wireless Sensor Network)의 특징들을 대부분 가지고 있으며 추가적으로 Adaption Layer를 통해 이미 구축되어 있는 IP 네트워크 인프라를 사용

4.4 6LoWPAN

▪ 6LoWPAN 특징 (2/2)

- IPv6 패킷의 라우팅, 단편화 및 재조립 담당
- 패킷의 포맷은 Mesh Hop 헤더, Fragment 헤더, IPv6 압축 헤더
  UDP 압축 헤더로 구성
- IEEE 802.15.4의 16/64비트 주소를 이용한 IPv6 주소 자동생성
  지원 (128비트 IPv6주소 사용이 아닌 64비트 EUI-64 식별자로 수행)
- 6LoWPAN과 외부IP네트워크와의 상호운영성을 제공하기 위해서는 게이트웨이가 필요,
  게이트웨이는 패킷 압축/해제와 sub IP 계층에서 패킷 단편화, 재조립 기능이 요구됨

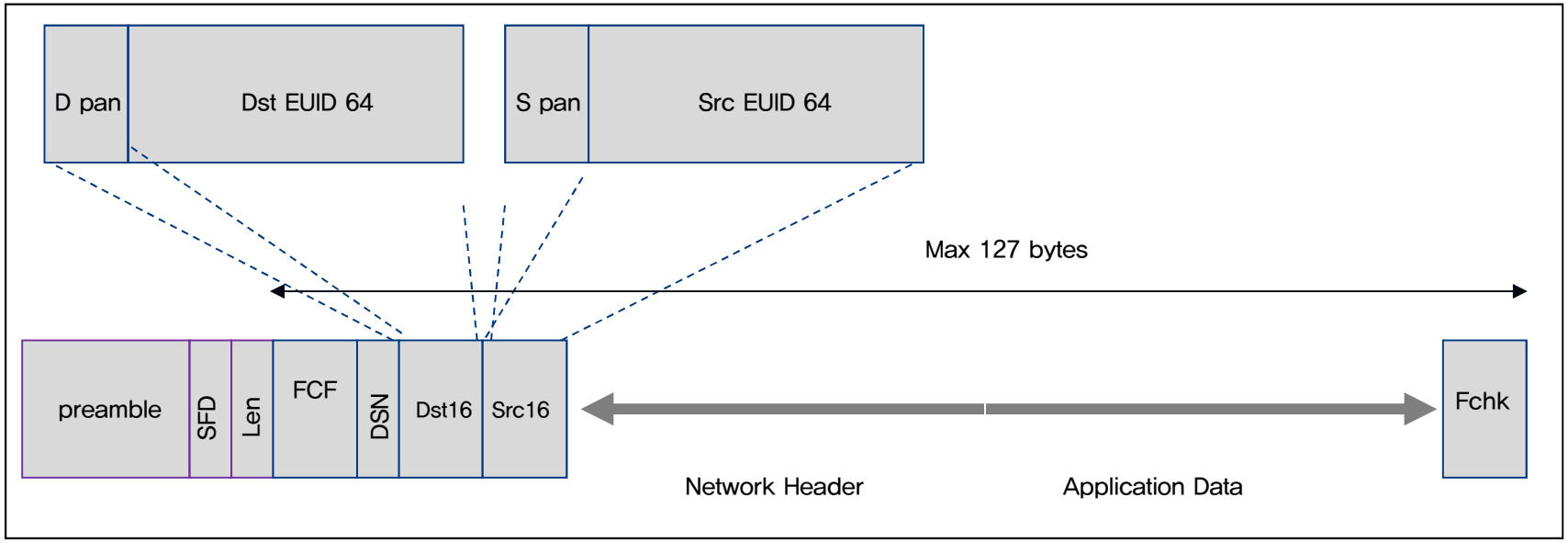
4.4 6LoWPAN

▪ 6LoWPAN 필요성

- IEEE 802.15.4 IPv6의 최소 MTU는 1280byte이기 때문에 센서 네트워크에 적합하지 않다.
  - 6LoWPAN의 최대 MTU는 127byte
- 헤더 압축 기술 지원, 단편화 기능 지원
- IP 토폴로지 아래에서 링크 레이어 Mesh Routing 허용
  (Mesh Under IP Routing)
- 802.15.4 Mesh 노드 위에서 IP Routing 허용
  (Route Over IP Routing)

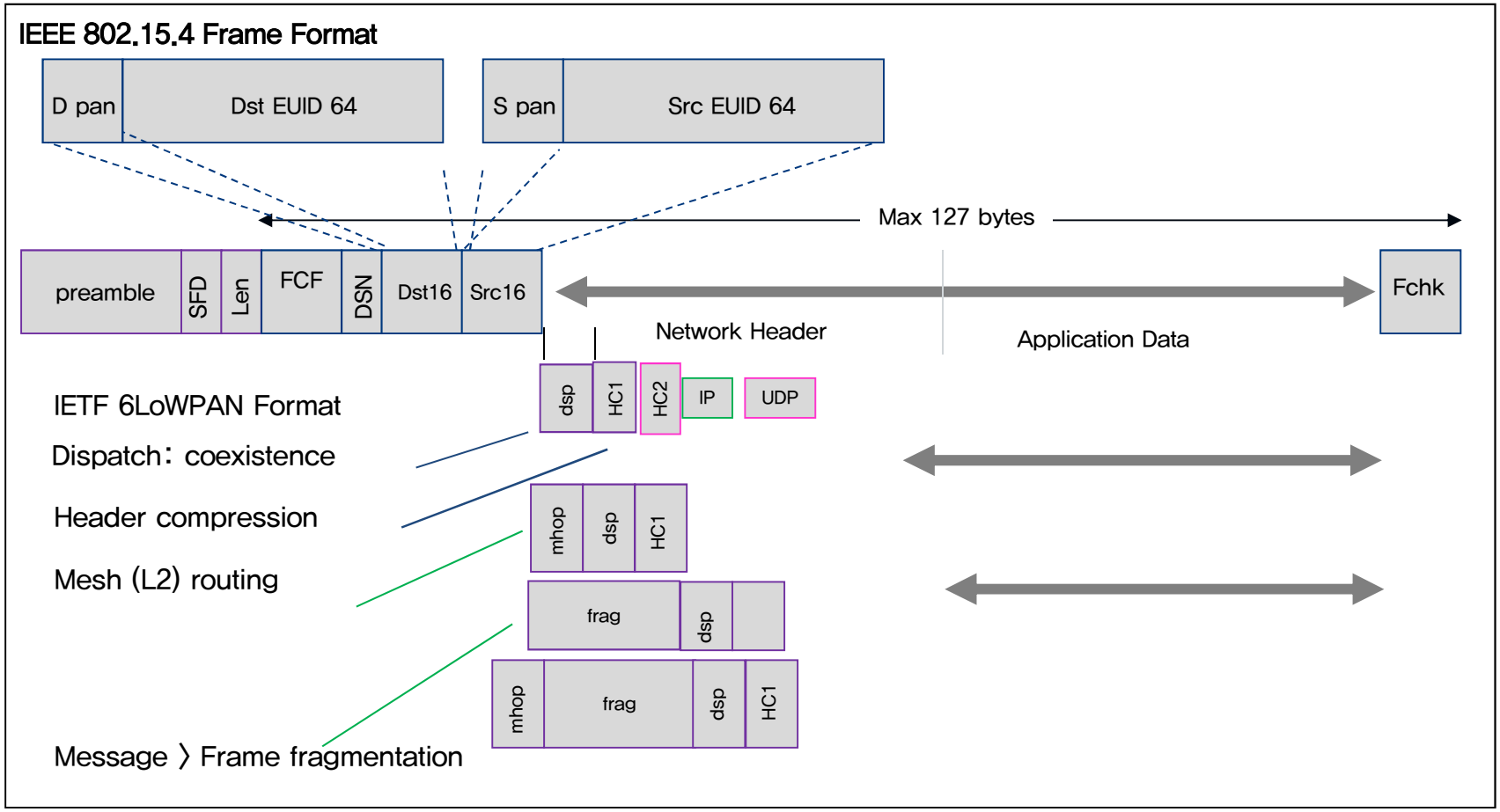
4.4 6LoWPAN

- IEEE 802.15.4 프레임 포맷



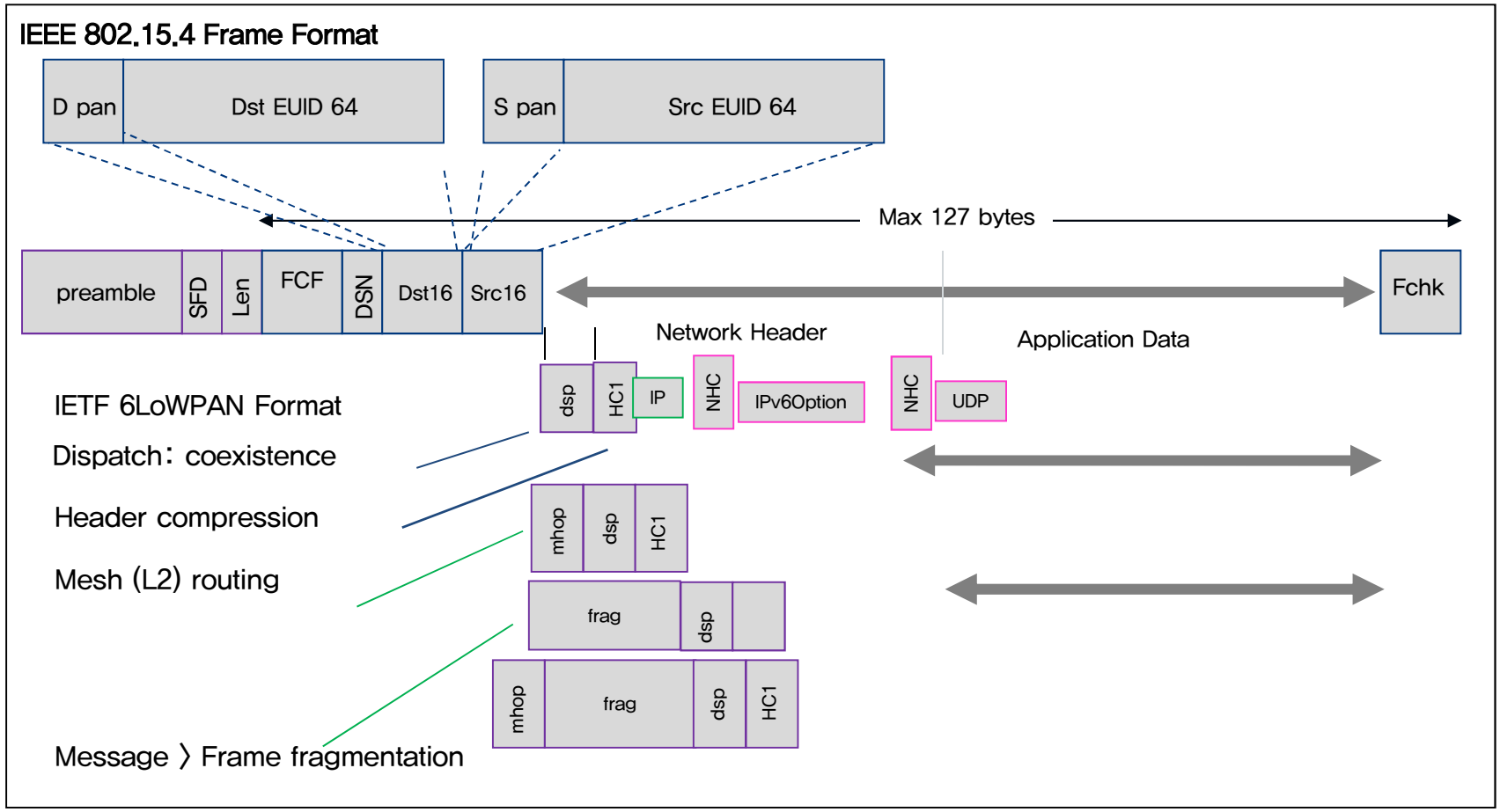
4.4 6LoWPAN

▪ 6LoWPAN 포맷 디자인 1 (HC1)



4.4 6LoWPAN

- 6LoWPAN 포맷 디자인 2 (IPHC)



Assignment

- Assignment 4

학습한 사물인터넷 통신 프로토콜 간의 공통적인 특징
및 차이점에 대해 조사하고 정리해오시오.

Reference

[4.2] LPWA(LoRaWAN)

LoRaWAN 클래스

그림 출처 : https://www.lora-alliance.org/For-Developers/LoRaWANDevelopers

LoRaWAN 토폴로지 구성

그림 출처 : http://www.semtech.com/wireless-rf/internet-of-things/what-is-lora/

LoRaWAN 프로토콜 스택

그림 출처 : https://www.lora-alliance.org/What-Is-LoRa/Technology

LoRaWAN 메시지 포맷

그림 출처 : https://www.lora-alliance.org/For-Developers/LoRaWANDevelopers

[4.2] LPWA(Sigfox)

Sigfox 개념

그림 출처 : http://www.libelium.com/sigfox-connectivity-waspmote-868mhz-europe-900mhz-us-long-range/

[4.2] LPWA(NB-IoT)

NB-IoT 표준

그림 출처 : http://www.newelectronics.co.uk/electronics-technology/lte-for-the-iot-not-one-standard-but-many/146360/

5장. oneM2M Release 1

Chapter 4. oneM2M Release 1

5.1 oneM2M 프로토콜
5.2 MQTT
5.3 HTTP
5.4 CoAP

- Assignment
- Reference

강의 개요

- **강의 목표**

  oneM2M에 대한 개념 이해 후 oneM2M Release 1 프로토콜에 대해 학습 한다.

- **강의 내용**

  - oneM2M Release 1 프로토콜 개념
  - MQTT 프로토콜
  - HTTP 프로토콜
  - CoAP프로토콜

5.1 oneM2M 프로토콜

▪ oneM2M Release 1 프로토콜

- oneM2M는 IoT/M2M(Machine to Machine)서비스 플랫폼 표준을 개발하기 위해 세계 주요 표준화 기관이 공동으로 설립한 글로벌 표준화 기구이다.

- oneM2M Release 1 규격은 기술규격(TS: Technical Specification)들을 패키지로 제공하며 워킹그룹3(WG3/Protocols)에서는 전송계층 프로토콜(MQTT, HTTP , CoAP)과의 바인딩을 정의하였다.

- oneM2M Release 1 프로토콜 실습 간에는 사물인터넷의 디바이스의 제한적인 환경을 위해 구현된 CoAP, MQTT 프로토콜만을 배운다.

5.1 oneM2M 프로토콜

- oneM2M Release 1 specifications

oneM2M Release 1 specifications

Latest	Reference	Version	Title	Date	ARIB	ATIS	CCSA	ETSI	TIA	TSDSI	TTA	TTC
	TS 0001	1.6.1	Functional Architecture	01/2015		ATIS.oneM2M.TS0001V161-2015		TS 118 101 V1.0.0	TIA-5022.001		TTAT.MM-TS.0001	TS-M2M-0001v1.6.1
★	TS 0001	1.13.1	Functional Architecture	03/2016				TS 118 101 V1.1.0				TS-M2M-0001v1.13.1
★	TS 0002	1.0.1	Requirements	01/2015		ATIS.oneM2M.TS0002V101-2015		TS 118 102 V1.0.0	TIA-5022.002		TTAT.MM-TS.0002	TS-M2M-0002v1.0.1
	TS 0003	1.0.1	Security Solutions	01/2015		ATIS.oneM2M.TS0003V101-2015		TS 118 103 V1.0.0	TIA-5022.003		TTAT.MM-TS.0003	TS-M2M-0003v1.0.1
★	TS 0003	1.4.2	Security Solutions	03/2016				TS 118 103 V1.1.0				TS-M2M-0003v1.4.2
	TS 0004	1.0.1	Service Layer Core Protocol Specification	01/2015		ATIS.oneM2M.TS0004V101-2015		TS 118 104 V1.0.0	TIA-5022.004		TTAT.MM-TS.0004	TS-M2M-0004v1.0.1
★	TS 0004	1.6.0	Service Layer Core Protocol Specification	03/2016				TS 118 104 V1.1.0				TS-M2M-0004v1.6.0
	TS 0005	1.0.1	Management Enablement (OMA)	01/2015		ATIS.oneM2M.TS0005V101-2015		TS 118 105 V1.0.0	TIA-5022.005		TTAT.MM-TS.0005	TS-M2M-0005v1.0.1
★	TS 0005	1.4.1	Management Enablement (OMA)	03/2016				TS 118 105 V1.1.0				TS-M2M-0005v1.4.1
	TS 0006	1.0.1	Management Enablement (BBF)	01/2015		ATIS.oneM2M.TS0006V101-2015		TS 118 106 V1.0.0	TIA-5022.006		TTAT.MM-TS.0006	TS-M2M-0006v1.0.1
★	TS 0006	1.1.4	Management Enablement (BBF)	03/2016				TS 118 106 V1.1.0				TS-M2M-0006v1.1.4
	TS 0008	1.0.1	CoAP Protocol Binding	01/2015		ATIS.oneM2M.TS0008V101-2015		TS 118 108 V1.0.0	TIA-5022.008		TTAT.MM-TS.0008	TS-M2M-0008v1.0.1
★	TS 0008	1.3.2	CoAP Protocol Binding	03/2016				TS 118 108 V1.1.0				TS-M2M-0008v1.3.2
	TS 0009	1.0.1	HTTP Protocol Binding	01/2015		ATIS.oneM2M.TS0009V101-2015		TS 118 109 V1.0.0	TIA-5022.009		TTAT.MM-TS.0009	TS-M2M-0009v1.0.1
★	TS 0009	1.5.1	HTTP Protocol Binding	03/2016				TS 118 109 V1.1.0				TS-M2M-0009v1.5.1
	TS 0010	1.0.1	MQTT Protocol Binding	01/2015		ATIS.oneM2M.TS0010V101-2015		TS 118 110 V1.0.0	TIA-5022.010		TTAT.MM-TS.0010	TS-M2M-0010v1.0.1
★	TS 0010	1.5.1	MQTT Protocol Binding	03/2016				TS 118 110 V1.1.0				TS-M2M-0010v1.5.1
★	TS 0011	1.2.1	Common Terminology	01/2015		ATIS.oneM2M.TS0011V121-2015		TS 118 111 V1.0.0	TIA-5022.0011		TTAT.MM-TS.0011	TS-M2M-0011v1.2.1
★	TS 0013	1.0.0	Interoperability Testing	03/2016				TS 118 113 V1.0.0				TS-M2M-0013v1.0.0
★	TR 0025	1.0.0	Application Developer Guide	03/2016				TR 118 525 V1.0.0				TR-M2M-0025v1.0.0

5.2 MQTT

▪ MQTT 개념

- MQTT(Message Queue Telemetry Transport) 는 지연 및 손실이 심한 네트워크환경에서 검침기, 센서 등 작은 기기들의 신뢰성 있는 메시지 전달(원격 모니터링)을 위해서 IBM에서 1999년 에 개발한 메시지 프로토콜

- MQTT는 경량의 Publish/Subscribe 프로토콜. M2M(Machine-to-Machine) 및 IoT 환경에서 사용됨

5.2 MQTT

- MQTT 장단점
  - 단순한 메시지 포맷을 바탕으로 네트워크 대역 및 배터리소비가 적음
  - 저전력으로 동작되며 데이터 전송 지연 및 손실이 발생하는 네트워크 환경을 고려하여 설계되어 소형 기기의 제어와 센서 정보 수집에 유리
  - 소형 기기에 올려서 사용하기 적당하지만 기본적으로 클러스터를 지원하지 않아 대규모 메시지 시스템을 구축에 있어서 어려움을 가짐

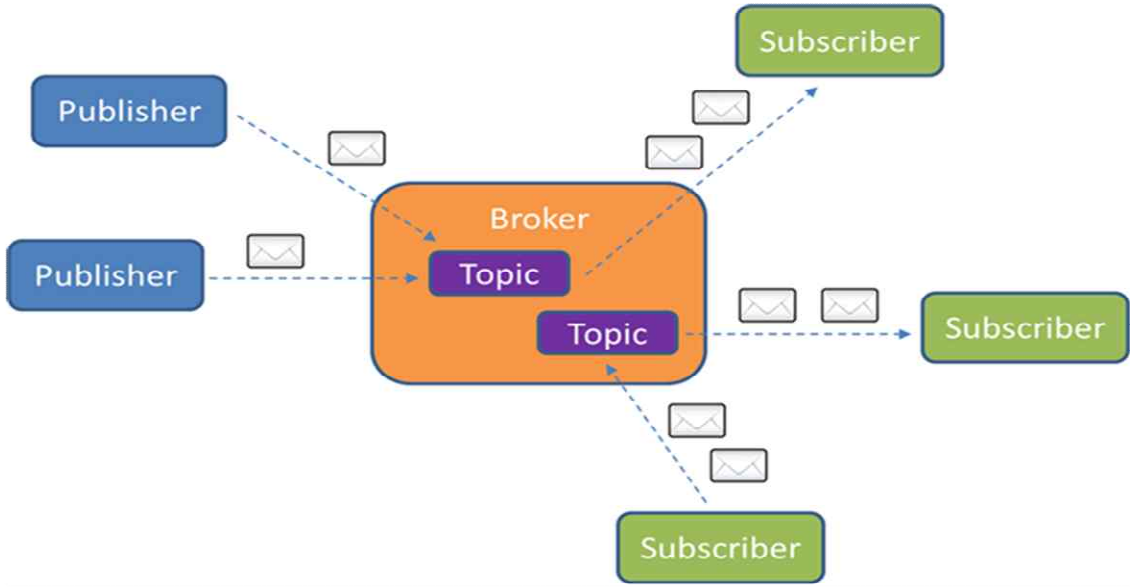
5.2 MQTT

▪ MQTT 특징 (1/3)

▪ Publish/Subscribe

- MQTT 프로토콜은 메지지를 발행(Publish) 하고 주제를 구독(Subscribe)을 원칙으로 함
- Publisher 및 Subscriber은 모두 Broker에 대한 클라이언트로 작동한다. Publisher는 토픽을 발행하기 위한 목적으로 Subscriber은 토픽을 구독하기 위한 목적으로 Broker 서버에 연결
- 하나 이상의 Publisher 와 Subscriber 가 Broker 서버에 연결해서 토픽을 발행 또는 구독할 수 있다. 또한 다수의 클라이언트가 하나의 주제를 구독할 수 있음

▪ Publish/Subscribe 동작 환경



5.2 MQTT

▪ MQTT 특징 (2/3)

▪ 토픽(Topic)

- Publisher 와 Subscriber 는 메시지를 채널 단위로 발행/구독이 일어난다 이를 MQTT에서는 토픽(Topic)이라고 함
- 토픽은 슬래시(/) 로 구분된 계층구조를 가지며 메시지 발행/구독 시에 여러 토픽을 지정할 수 있도록 Wild Card 문자를 지원
- Wild Card 종류
  - + (single – level wild card) : 단 1개의 토픽을 임의로 대처
  - # (multi– level wild card) : 하위 토픽을 모두 지칭, 토픽 문자열 끝만 사용가능
- Wild Card 사용 예제
  - sensors / NODE_NAME / + / CPU
  - Sensors / NODE_NAME / #

▪ QoS (Quality of service)

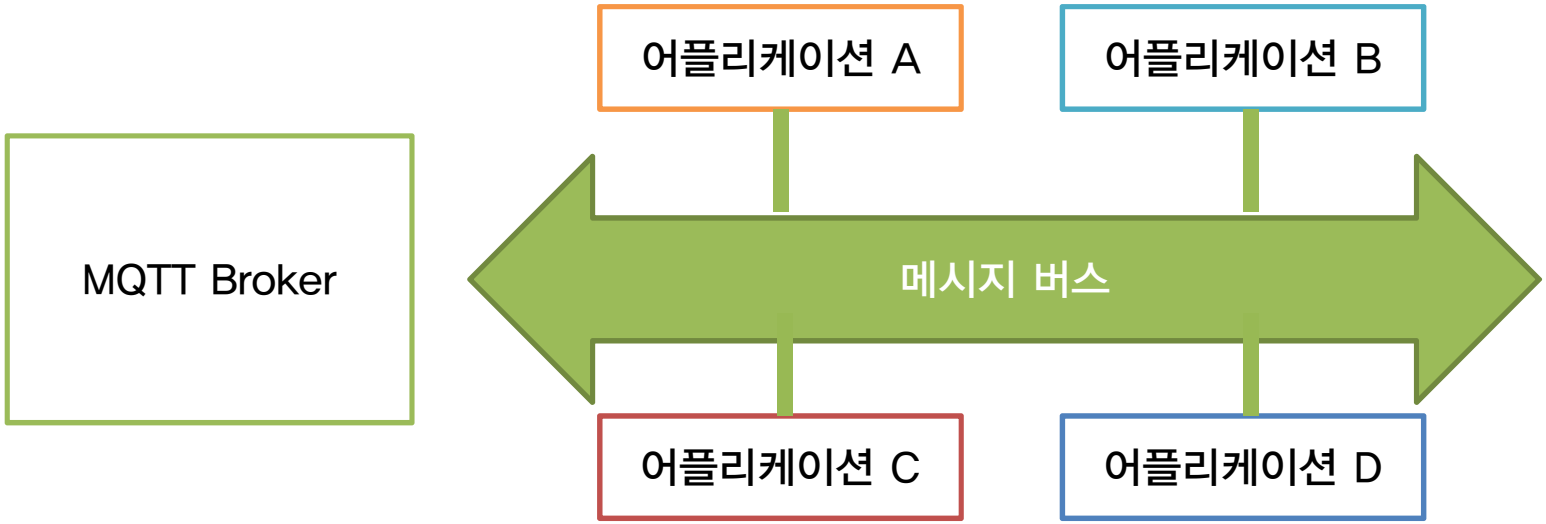
- MQTT는 3단계의 QoS(Quality of service)를 제공
  - 레벨 0 : 메시지 최대 1번 전달 , 유실 가능성 있음
  - 레벨 1 : 메시지 최소 1번 전달 , 중복 가능성 있음
  - 레벨 2: 메시지 단 한번만 정확하게 전달 , 품질을 보장하지만 시스템 성능이 요구됨

5.2 MQTT

▪ MQTT 특징 (3/3)

▪ 메시지 버스

- MQTT는 메시지 버스 시스템을 가진다. MQTT Broker 서버가 메시지 버스를 만들고 이 곳에 메시지를 보내면, 버스에 연결 되어 있는 애플리케이션들이 메시지를 읽어가는 방식
- 메시지 버스에는 다양한 주제의 메시지들이 흐를 수 있는데, 메시지를 구분하기 위해서 토픽(Topic)을 이름으로 하는 메시지 채널을 만듦



5.2 MQTT

▪ MQTT 메시지 포맷 (1/5)

- MQTT 메시지 포맷은 fixed Header 2byte를 제외하고 전부 옵션

	0	1	2	3	4	5	6	7
Byte1	*Message Type*				*DUP flag*	*QoS level*		*RETAIN*
Byte2	*Remaining Length*							

5.2 MQTT

▪ MQTT 메시지 포맷 (2/5)

▪ Message Type

Type	Enumeration	Description
Reserved	*0*	*Reserved*
CONNECT	*1*	*Client request to connect to Server*
CONNACK	*2*	*Connect Acknowledgment*
PUBLISH	*3*	*Publish message*
PUBACK	*4*	*Publish Acknowledgment*
PUBREC	*5*	*Publish Received (assured delivery part 1)*
PUBREL	*6*	*Publish Received (assured delivery part 2)*
PUBCOMP	*7*	*Publish Complete (assured delivery part 3)*
SUBSCRIBE	*8*	*Client Subscribe request*
SUBACK	*9*	*Subscribe Acknowledgment*
UNSUBSCRIBE	*10*	*Unsubscribe Acknowledgment*
UNSUBACK	*11*	*Unsubscribe Acknowledgment*
PINGREQ	*12*	*PING Request*
PINGRESP	*13*	*PING Response*
DISCONNECT	*14*	*Client is Disconnecting*
Reserved	*15*	*Reserved*

5.2 MQTT

▪ MQTT 메시지 포맷 (3/5)

▪ DUP Flag & QoS level & RETAIN

• RETAIN은 Publish 메시지에서만 사용
(Queue에 저장된 메시지 유지 여부)

Bit position	Name	Description
3	*DUP*	*Duplicate delivery*
2–1	*QoS*	*Quality of Service*
0	*RETAIN*	*RETAIN flag*

5.2 MQTT

▪ MQTT 메시지 포맷 (4/5)

▪ QoS level

QoS value	Bit 2	Bit 1	Description
0	*0*	*0*	*Fire and Forget (최대 한번)*
1	*0*	*1*	*Acknowledged delivery (적어도 한번)*
2	*1*	*0*	*Fire and Forget (정확히 한번)*
3	*1*	*1*	*Fire and Forget (최대 한번)*
4	*1*	*1*	*Reserved*

5.2 MQTT

▪ MQTT 메시지 포맷 (5/5)

▪ Remaining Length

• Variable header를 포함한 전체 메시지의 크기를 계산하기 위해서 사용한다. 1~4byte를 사용하며 데이터 크기에 따라 가변적

▪ Variable header

• 일부 MQTT 명령들은 가변(variable) 헤더 요소를 포함한다. 가변 헤더는 고정 헤더와 페이로드(payload) 사이에 위치.

5.3 HTTP

▪ HTTP 개념

- HTTP(HyperText Transfer Protocol)는 인터넷 상에서 데이터를 주고 받기 위한 서버/클라이언트 모델을 따르는 프로토콜 이다.

- Web Browser와 Web Server의 통신규약으로 요청/응답(request/response) 프로토콜이다.

▪ HTTP 특징 및 장단점

- 비연결 지향(Connectless)으로 서버에 연결 후 요청에 대한 응답을 받으면 연결을 종료한다.
  - 장점 : 다수를 대상으로 최소한의 접속 유지를 통해 효율적으로 서비스 가능하다.
  - 단점 : 연결 종료 시에 클라이언트의 이전 상태를 알 수 없어 클라이언트에 대한 정보를 유지 할 수 없다.

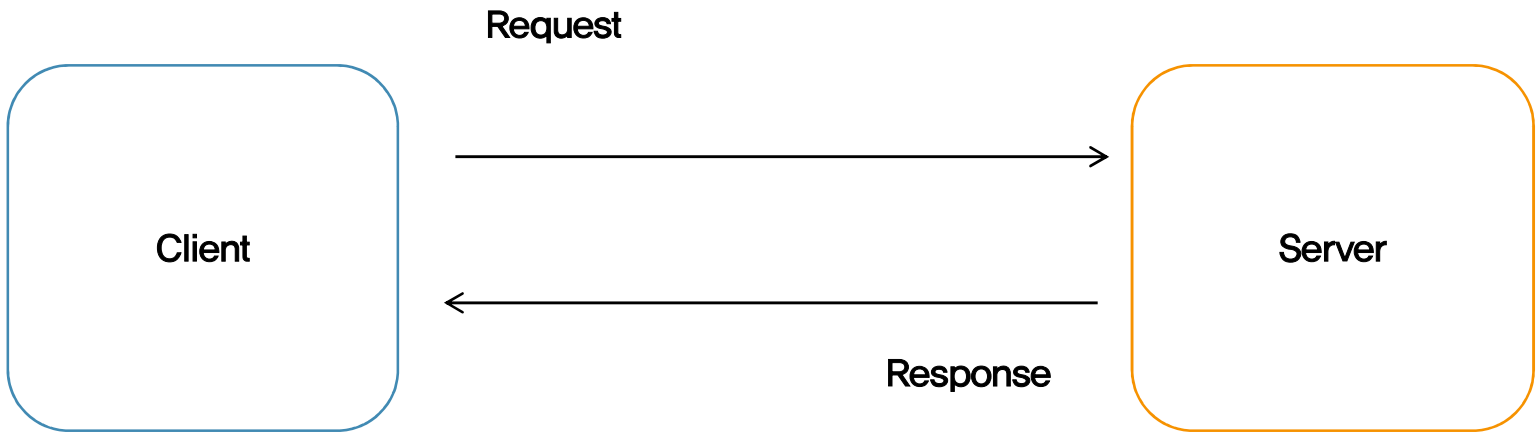
- 클라이언트에 대한 정보 유지 (Stateless)를 해결하기 위한 방법
  - Cookie : 클라이언트에 대한 상태 정보 값 저장하여 문제 해결

5.3 HTTP

▪ HTTP 동작환경

- Client : 서버에 클라이언트 소프트웨어가 설치된 컴퓨터로 URI를 이용해 서버에 접속하고 데이터를 요청(chrome, firefox, ie 등)

- Server: 클라이언트의 요청을 받아 요청을 해석하고 응답을 하는 소프트웨어가 설치된 컴퓨터 (Apache, nginx, IIS, lighttpd)



5.3 HTTP

- HTTP 메소드
  - GET : 정보를 요청
  - POST : 정보를 삽입
  - PUT : 정보를 업데이트
  - DELETE : 정보를 삭제
  - HEAD : 헤더 정보만을 요청
  - OPTIONS : 웹 서버가 지원하는 메소드의 종류 요청
  - TRACE : 클라이언트의 요청 반환

5.3 HTTP

▪ HTTP 응답코드

▪ 1XX 조건부 응답

▪ 2XX 성공

- 200 : 성공 , 201 : 생성됨, 202 : 허용됨, 204 : 컨텐츠 없음
- 205 : 컨텐츠 재설정, 206 : 일부 성공 , 207 : 다중 상태

▪ 3XX 리다이렉션

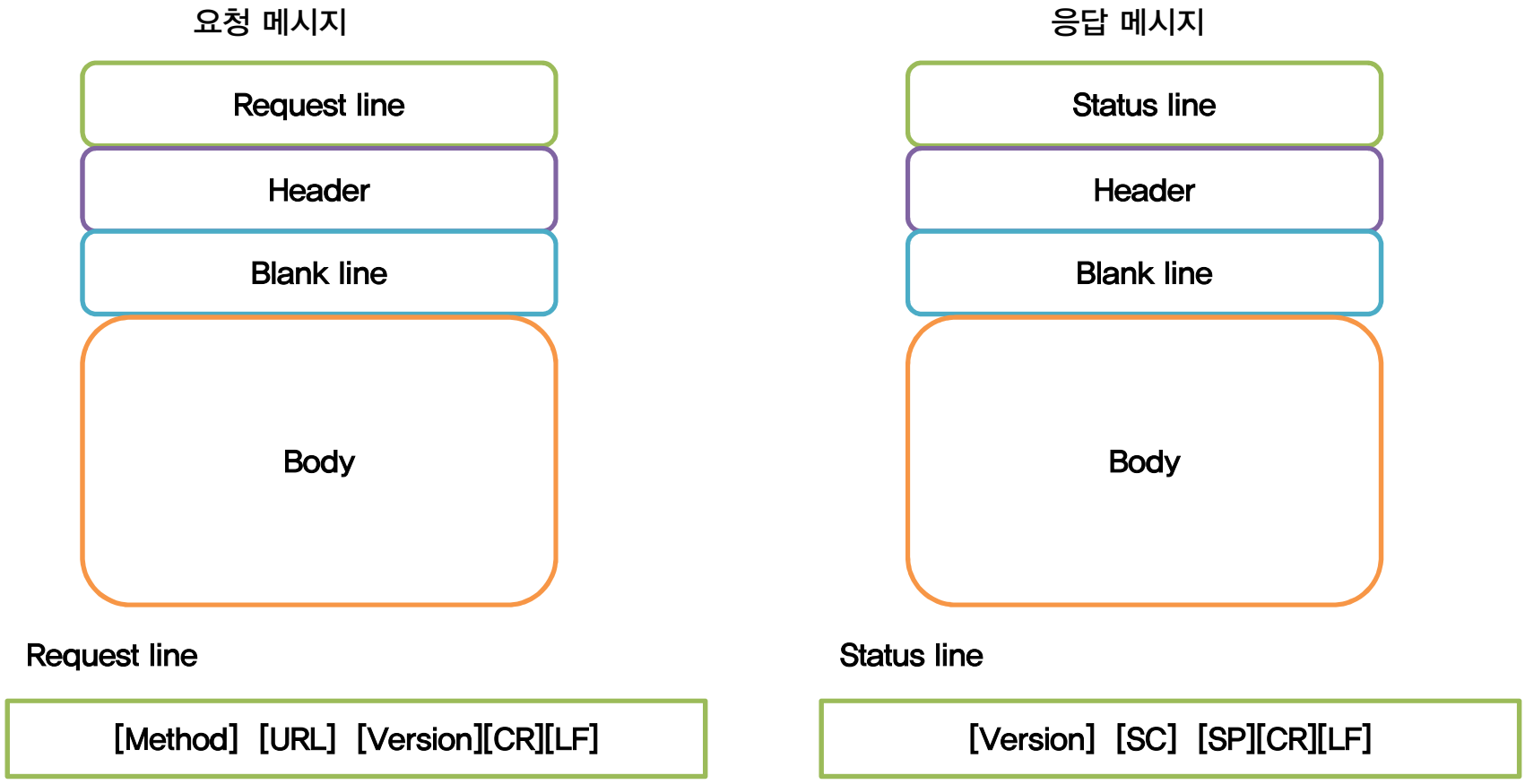
- 300 : 여러 선택 항목 , 301 : 영구이동 , 302 : 임시이동
- 303 : 기타위치보기, 304 : 수정되지 않음, 305 : 프록시 사용

▪ 4XX 요청오류

- 400 : 잘못된 요청 , 401: 권한 없음 , 403 : 금지 404 : 찾을 수 없음
- 405 : 허용하지 않는 방법 , 406 : 허용되지 않음 , 408 : 요청시간 초과
- 410 사라짐, 411 : 길이필요

5.3 HTTP

- HTTP 응답 및 요청 데이터 포맷



5.4 CoAP

▪ CoAP 개념

- CoAP(Constrained Application Protocol) 는 전송지연과 패킷 손실률이 높은 네트워크 환경에서 저사양의 하드웨어로 동작되는 센서 디바이스의 RESTful 웹 서비스를 지원하기 위한 경향 프로토콜

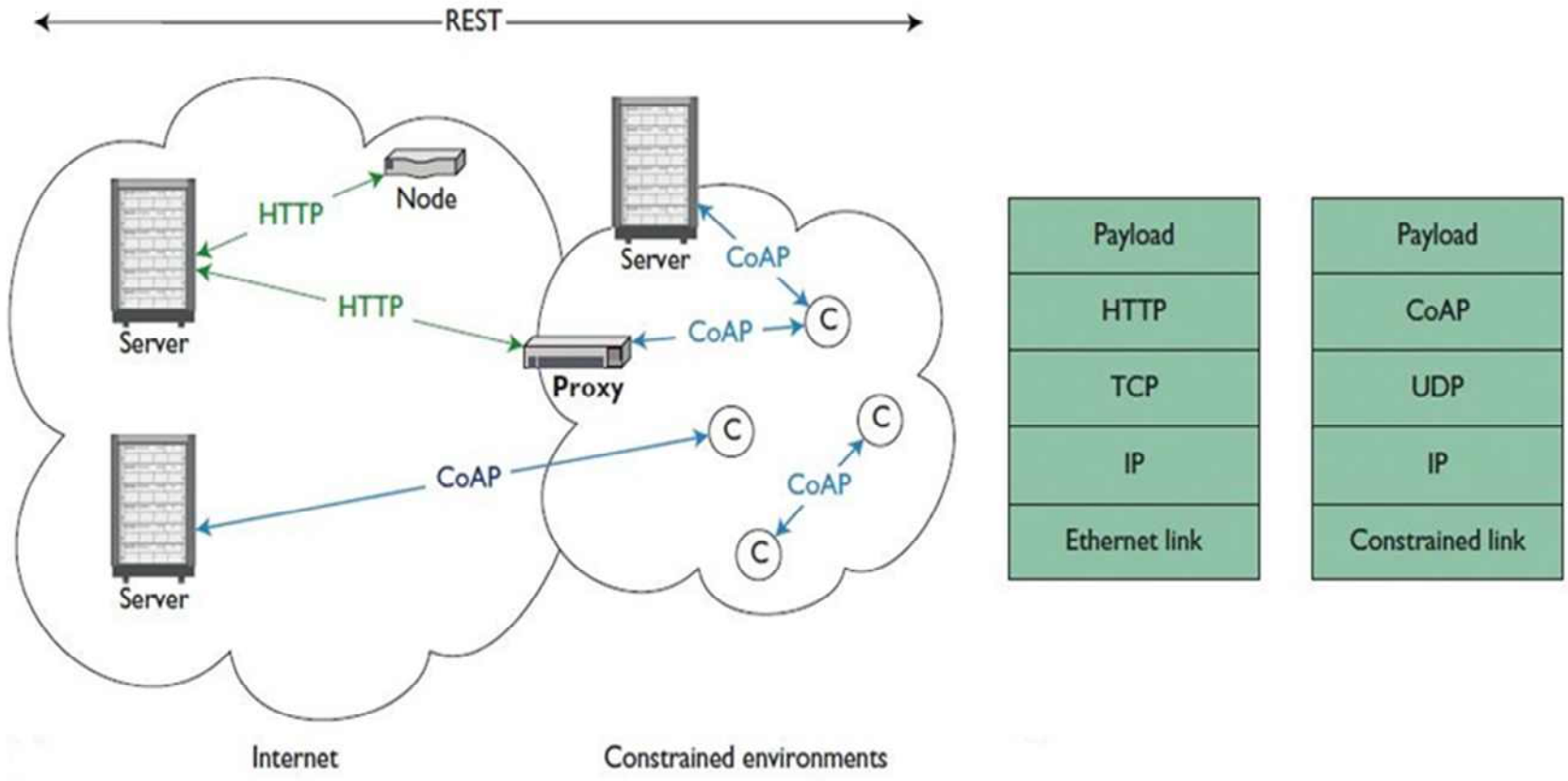
- 하위 프로토콜 스택으로 IEEE 802.15.4 표준을 기반으로 삼고, 네트워크 계층은 IPv6를 사용

5.4 CoAP

- CoAP 특징
  - RESTful 기반의 접근 방식으로 기존 HTTP 프로토콜과 쉽게 변환 및 연동이 가능
  - 메시지 단편화 가능성이 낮음
  - UDP 환경에서의 유니 캐스트와 멀티 캐스트를 지원
  - CoAP은 확인형(confirmable), 비확인형(non-confirmable), 승인(acknowledgement), 리셋(reset) 4가지 메시지 타입으로 정의
  - CoAP는 HTTP의 대체 프로토콜 ,일반적 HTTP 압축이 아님

5.4 CoAP

- CoAP 동작환경



5.4 CoAP

▪ CoAP 처리 메시지

▪ Confirmable (CON)

- 메시지 전달에 대한 응답 또는 메시지가 도달했는지 알기 위해 사용
- 응답 메시지로 ACK/RST 메시지를 받음

▪ Acknowledgment (ACK)

- CON 메시지에 대한 응답(ACK) 메시지 , 메시지 ID(mid)로 구분
- 응답 메시지로 ACK/RST 메시지를 받음

▪ Reset (RST)

- 클라이언트에게 CON 메시지를 받았지만 프로세싱에 실패를 알림
- 예) 노드의 재부팅

▪ Non-Confirmable (NON)

- ACK를 요구하지 않는 메시지 타입
- 예) 센서로 부터의 반복적인 값 읽음

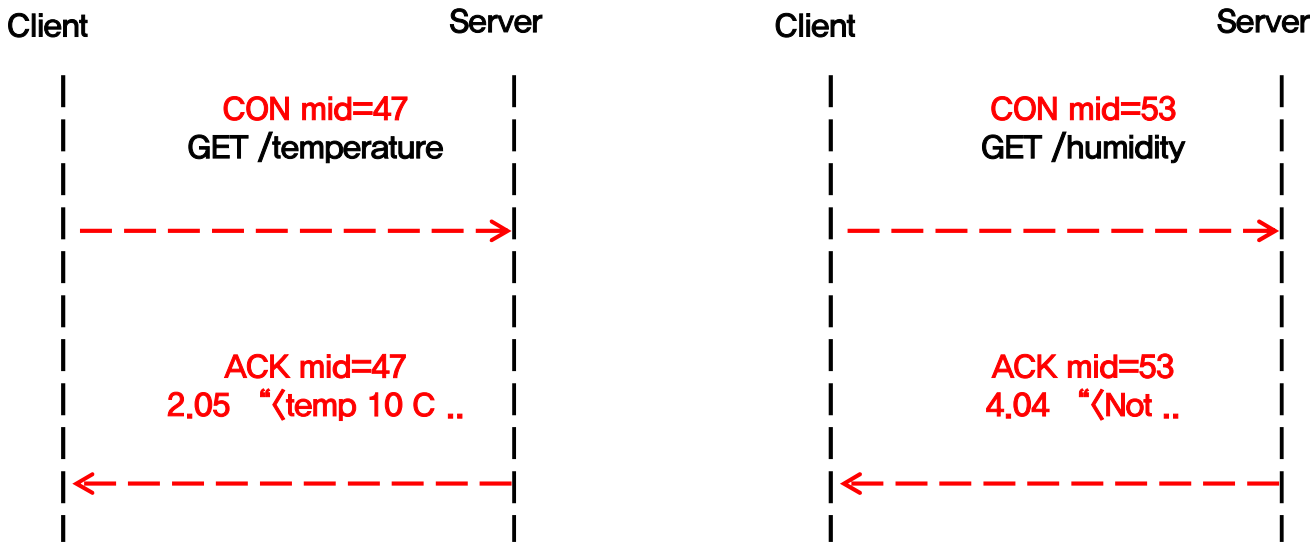
5.4 CoAP

- CoAP 메소드
  - GET : 요청한 URI에 의해 리소스 식별자에 대한 정보를 조회
  - POST : 요청한 URI 하위에 서버에 리소스 update 또는 create 요청
  - PUT : 요청한 URI에 의해 리소스 식별자를 message body와 같이 update 하기 위한 요청
  - DELETE : 요청한 URI에서의 리소스 식별자를 지우기 위한 요청

5.4 CoAP

▪ CoAP 메시지 교환

- CoAP interaction 은 Client / Server 모델과 유사
- HTTP와 같이 URI , Content-type 을 지원
- 메시지 교환 또한 HTTP와 유사ACK를 요구하지 않는 메시지 타입
  - 요청 동작을 클라이언트가 시작
  - Server는 URI의 식별자를 통해 리소스를 응답
  - 응답 메시지에는 응답코드와 함께 보냄



5.4 CoAP

▪ CoAP 처리(transaction) 모델

▪ UDP

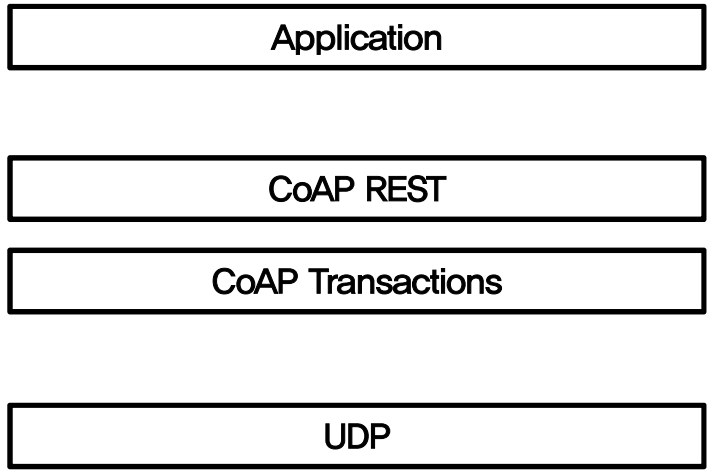
- CoAP은 UDP를 이용하여 사용하기 위해 정의

▪ Transaction

- end-point와의 단일 메시지 교환
- CON, NON, ACK, RST

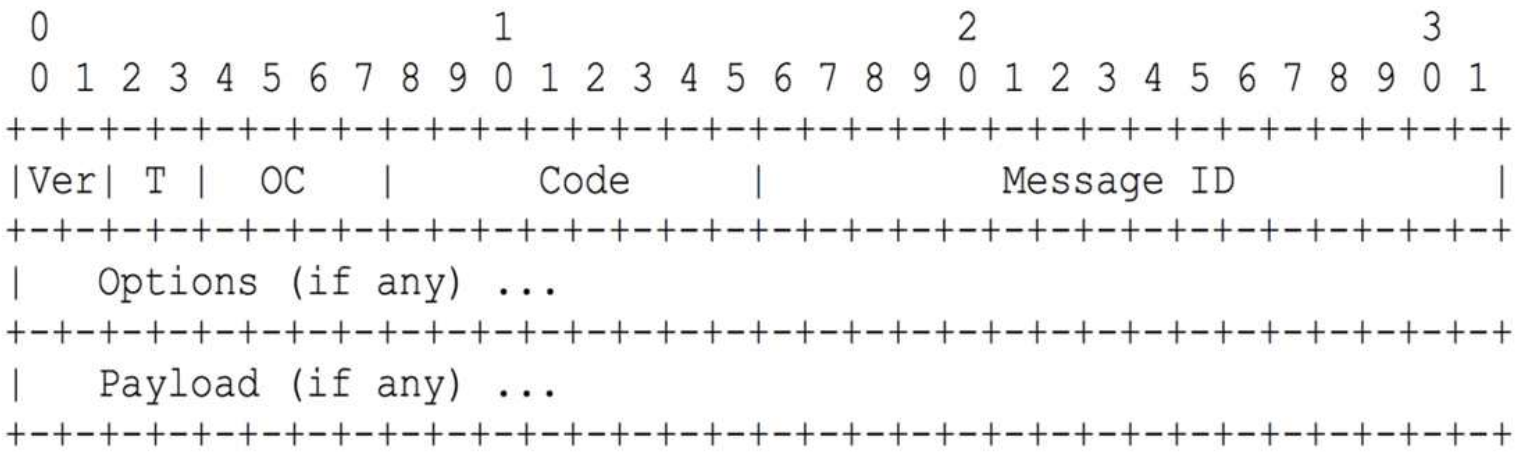
▪ REST

- Transaction의 piggyback
- 응답코드와 옵션
  (URI, content-type)



5.4 CoAP

- CoAP 메시지 포맷



Ver: Version,
T: Transaction Type (CON,NON,ACK...)
OC: Number of options
Code: Method or Response Code
Message ID: A unique ID assigned by the originator

5.4 CoAP

- CoAP 메시지 리턴 코드

	coap-03		coap-06	
	Code	Description	Code	Description
Request	1	GET	1	GET
	2	POST	2	POST
	3	PUT	3	PUT
	4	DELETE	4	DELETE
Response	40	100 Continue	-	-
	80	200 OK	69	2.05 Content
	81	201 Created	65	2.01 Created
	-	-	66	2.02 Deleted
	124	304 Not Modified	67	2.03 Valid
	-	-	68	2.04 Changed
	160	400 Bad Request	128	4.00 Bad Request
	-	-	129	4.01 Unauthorized
	-	-	130	4.02 Bad Option
	-	-	131	4.03 Forbidden
	164	404 Not Found	132	4.04 Not Found
	165	405 Method Not Allowed	133	4.05 Method Not Allowed
	-	-	141	4.13 Request Entity Too Large
	175	415 Unsupported Media Type	143	4.15 Unsupported Media Type
	200	500 Internal Server Error	160	5.00 Internal Server Error
	-	-	161	5.01 Not Implemented
	202	502 Bad Gateway	162	5.02 Bad Gateway
	203	503 Service Unavailable	163	5.03 Service Unavailable
	204	504 Gateway Timeout	164	5.04 Gateway Timeout
	-	-	165	5.05 Proxying Not Supported
	240	Token Option required	-	-
	241	Uri-Authority Option required	-	-
	242	Critical Option not supported	-	-

Source: Thomas Pötsch, ComNets, Master Thesis, 2011

Structure of the 8 bit Response code

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|class|  detail |
+-+-+-+-+-+-+-+-+
```

3 classes:

2 – Success
4 – Client error
5 – Server error

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|class|  detail |
+-+-+-+-+-+-+-+-+
|1 0 0|0 0 1 0 0|
+-+-+-+-+-+-+-+-+
```

4 : 04

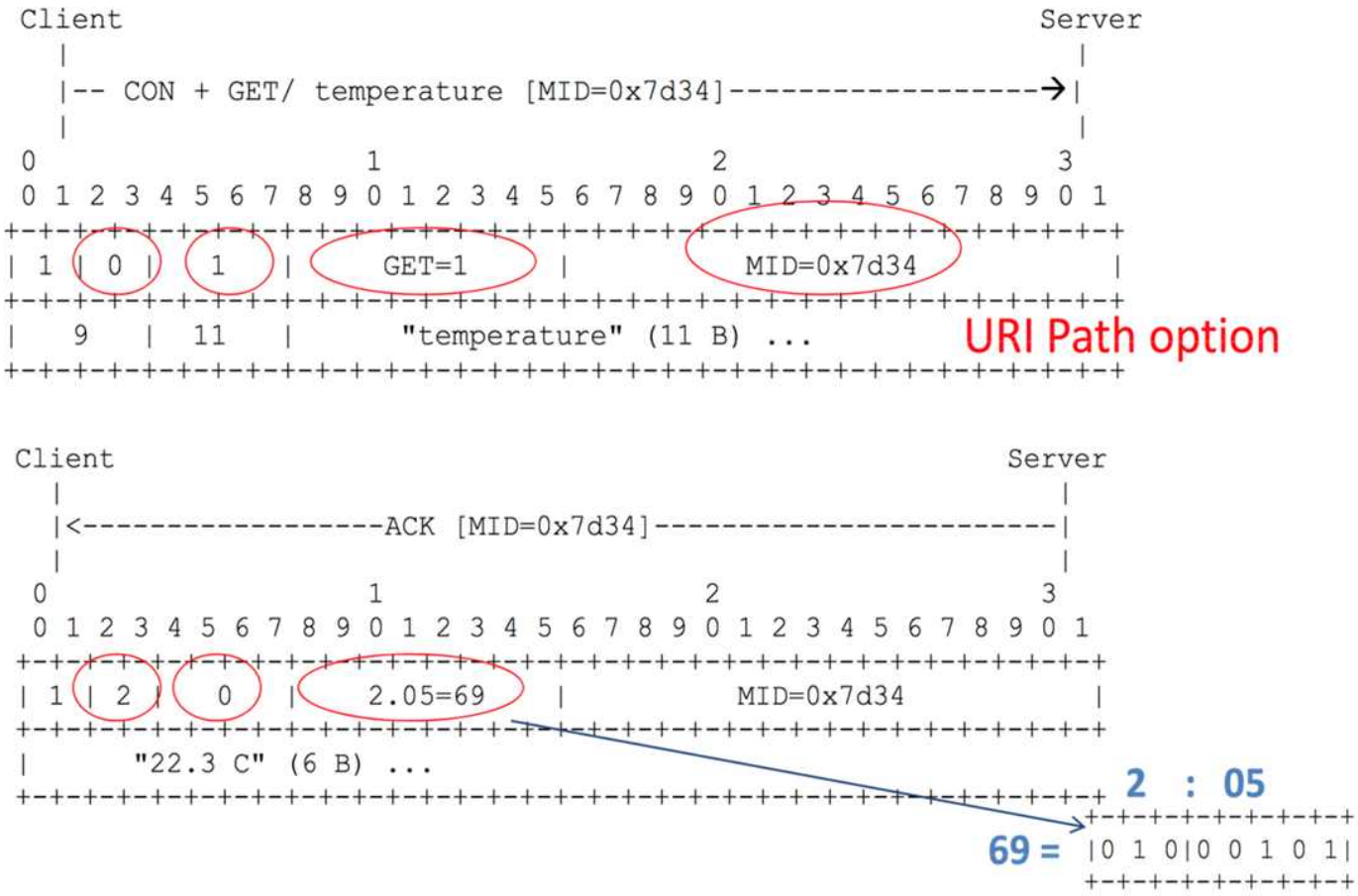
E.g., "Not Found" is written as 4.04 - indicating a value of decimal 132.

5.4 CoAP

- CoAP 옵션
  - Content-Type : Specifies the format of the application payload
  - Max-Age : Maximum age for a message to be cached
  - Proxy-Uri : Defines an absolute Uri to a proxy
  - Token : Matches an asynchronous response to a request
  - Uri-Host : Specifies the Internet host of a resource (only added if not representing the destination IP address of the request)
  - Uri-Path : Specifies the resource of the host
  - Uri-Port : Specifies the port of the host (only added if differs from the destination UDP port request)
  - Uri-Query : Indicates additional options for the request
  - Etc..

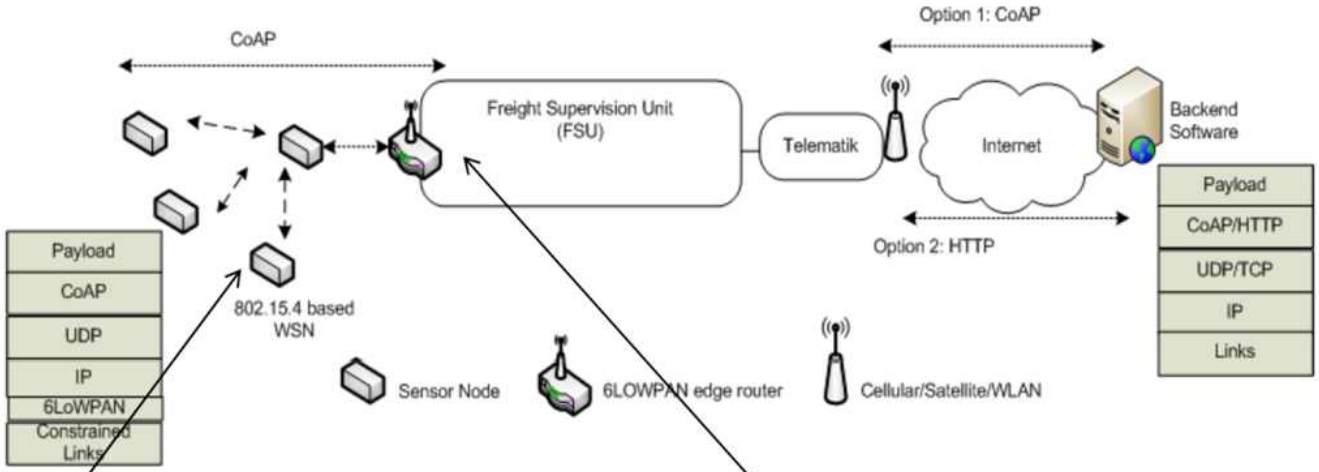
5.4 CoAP

- CoAP 메시지 및 동작 예제



5.4 CoAP

- CoAP 메시지 및 동작 예제



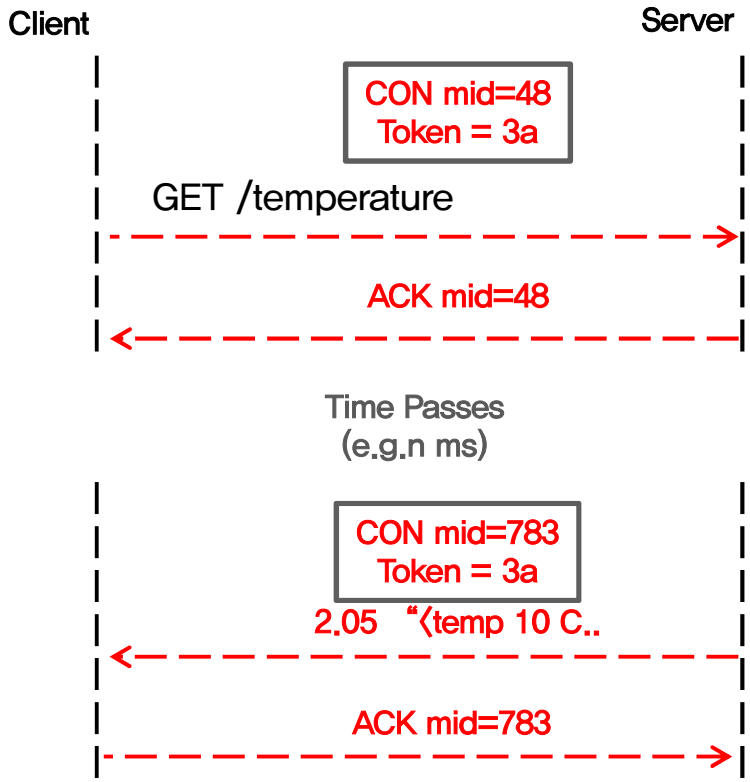
Resource	GET	PUT	Comments
/st	X		Temperature
/sh	X		Humidity
/sv	X		Voltage
/r	X		Temperature, humidity and voltage together
/l	X	X	LEDs
/ck	(X)	X	AES Encryption Key

Resource	GET	PUT	Comments
/ni	X		Inform about node integration into 6LoWPAN/RPL network
/warntemplow		X	Below Warning Temperature Low
/warntemphi		X	Above Warning Temperature High

More details: http://www.intelligentcontainer.com/

5.4 CoAP

- CoAP vs HTTP
  - UDP 프로토콜 위에서 신뢰성 있는 유니캐스트 및 멀티캐스트 지원
    - 재전송, 간단한 혼잡제어
  - 메시지 교환을 비동기적으로 가능
  - Application 층과 UDP 프로토콜 사이에 2개의 sublayer를 가짐
  - 서버는 Well-know 리소스 URI 을 통해 링크들의 목록을 제공.
    (Resource Discovery)



5.4 CoAP

CoAP vs HTTP

MQTT

- 발행 구독을 기반으로 중앙 Broker 서버와 연결(M:N)
- Broker 서버에는 메시지가 오며 복사하여 구독자에게 전달
- 지속적인 연결 지원, 실시간 데이터 전달
- TCP/IP 기반 프로토콜 (TCP 사용)
- 서비스 발견 기능이 없음, DNS-SD,SSDP등의 도움이 필요

CoAP

- 하나의 서버와 하나의 클라이언트가 참여하는 방식(1:1)
- 이벤트 기반 데이터 통신에 부적함
- 분산환경에서 상태 정보에 알맞음
- UDP 프로토콜만을 사용
- 서비스 발견 기능을 기본적으로 제공

Assignment

- Assignment 5

MQTT 을 통해 채팅 프로그램을 구현하려 한다. 채팅 프로그램을 구현하기 위한 발행/구독 구조를 그린 후 구현해 보시오. (채팅 프로그램이 아닌 MQTT 프로토콜을 사용하는 시스템에 대해서 구조를 그리고 구현하여도 무관)

Reference

[5.1] MQTT

MQTT 특징

그림 출처 : 한국 사물인터넷협회, IoT 지식능력검정 교육 자료

[5.4] CoAP

CoAP 동작환경

그림 출처 : 77th IETF core WG presentation

CoAP 처리(transaction) 모델

그림 출처 : https://tools.ietf.org/html/rfc7252

CoAP 메시지 리턴코드

그림 출처 : https://tools.ietf.org/html/rfc7252

CoAP 메시지 및 동작예제

그림 출처 : https://tools.ietf.org/html/rfc7252

CoAP 메시지 및 동작예제

그림 출처 : https://www.intelligentcontainer.com/

6장. oneM2M Release 2 및 실습

Chapter 6. oneM2M Release 2

6.1 oneM2M Release 2 프로토콜
6.2 Web Socket
6.3 MQTT 실습
6.4 CoAP 실습

- Assignment
- Reference

강의 개요

▪ 강의 목표

oneM2M에 대한 개념 이해 후 oneM2M Release 2 프로토콜에 대해 학습 한다.

▪ 강의 내용

- oneM2M Release 2 프로토콜 개념
- Web Socket 프로토콜
- MQTT 프로토콜
- CoAP 프로토콜

6.1 oneM2M Release 2 프로토콜

- oneM2M Release 2 specifications (1/2)

oneM2M Release 2 specifications

Latest	Reference	Version	Title	Date	ARIB	ATIS	CCSA	ETSI	TIA	TSDSI	TTA	TTC
★	TS 0001	2.10.0	Functional Architecture	08/2016								
★	TS 0002	2.7.1	Requirements	08/2016				TS 118 102 V2.7.1				
★	TS 0003	2.4.1	Security Solutions	08/2016				TS 118 103 V2.4.1				
★	TS 0004	2.7.1	Service Layer Core Protocol	08/2016								
★	TS 0005	2.0.0	Management Enablement (OMA)	08/2016				TS 118 105 V2.0.0				
★	TS 0006	2.0.1	Management Enablement (BBF)	08/2016								
★	TS 0007	2.0.0	Service Components	08/2016								
★	TS 0009	2.6.1	HTTP Protocol Binding	08/2016				TS 118 109 V2.6.1				
★	TS 0010	2.4.1	MQTT Protocol Binding	08/2016				TS 118 110 V2.4.1				
★	TS 0011	2.4.1	Common Terminology	08/2016				TS 118 111 V2.4.1				
★	TS 0012	2.0.0	oneM2M Base Ontology	08/2016				TS 118 112 V2.0.0				
★	TS 0014	2.0.0	LWM2M Interworking	08/2016				TS 118 114 V2.0.0				
★	TS 0015	2.0.0	Testing Framework	08/2016				TS 118 115 V2.0.0				
★	TS 0020	2.0.0	Websocket Protocol Binding	08/2016				TS 118 120 V2.0.0				

6.1 oneM2M Release 2 프로토콜

- oneM2M Release 2 specifications (2/2)

★	TS 0021	2.0.0	oneM2M and AllJoyn Interworking	08/2016	TS 118 121 V2.0.0
★	TS 0023	2.0.0	Home Applisnces Information Model and Mapping	08/2016	TS 118 123 V2.0.0
★	TS 0024	2.0.0	OIC Interworking	08/2016	TS 118 124 V2.0.0
★	TR 0001	2.4.1	Use Cases Collection	08/2016	
★	TR 0007	2.11.1	Study of Abstraction and Semantics Enablements	08/2016	
★	TR 0008	2.0.0	Security	08/2016	
★	TR 0012	2.0.0	oneM2M End-to-End Security and Group Authentication	08/2016	TR 118 512 V2.0.0
★	TR 0016	2.0.0	Study of Authorization Architecture for Supporting Heterogeneous Access Control Policies	08/2016	TR 118 516 V2.0.0
★	TR 0017	2.0.0	Home Domain Abstract Information Model	08/2016	TR 118 517 V2.0.0
★	TR 0018	2.0.0	Industrial Domain Enablement	08/2016	TR 118 517 V2.0.0
★	TR 0022	2.0.0	Continuation & integration of HGI Smart Home activities	08/2016	TR 518 522 V2.0.0
★	TR 0024	2.0.0	3GPP Release 13 Interworking	08/2016	TR 518 524 V2.0.0

6.2 Web Socket

▪ Web Socket 개념

- HTTP를 기반으로 HTTP의 단점을 해결을 목표로 한 프로토콜

- HTTP는 연결을 유지하지 않는 특성으로 인해 10번의 요청이 발생 할 시 10번의 연결을 맺고 끊는 과정이 필요하며 모든 요청에 헤더파일이 중복되어 들어간다. 또한 실시간 상호작용성이 떨어진다.

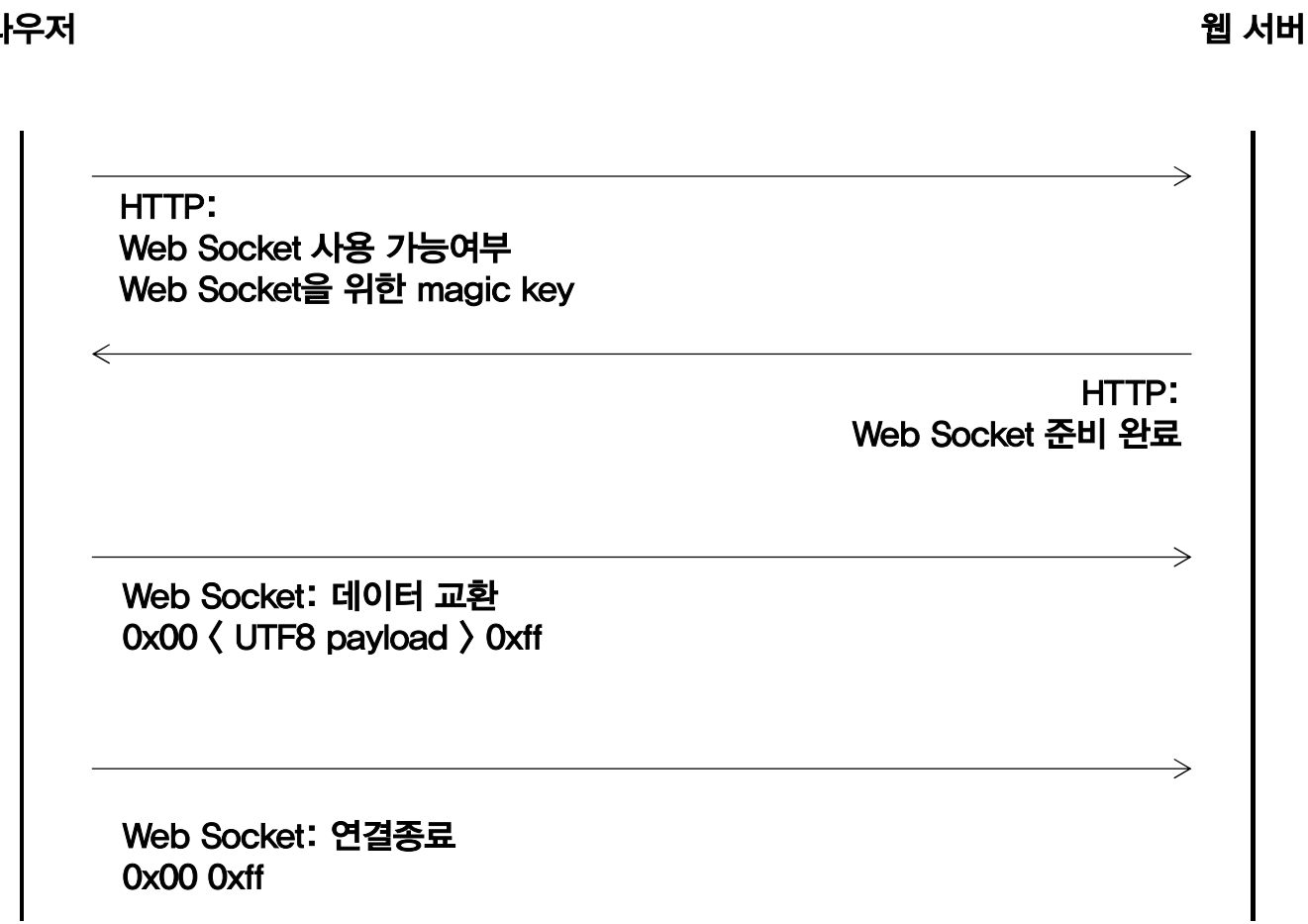
- Web Socket은 실시간 양방향 통신을 지원하며 UTF8포맷 메시지 스트림만을 허용한다.

6.2 Web Socket

- Web Socket 특징
  - 웹 통신에서의 실시간 고속 양방향 통신 지원
  - 많은 수의 동시 접속자 수용 용이
  - HTML5 기반의 스트리밍 , 커뮤니케이션 작성 지원
  - 기존 실시간 데이터 처리를 위해서 HTTP상에서 데이터를 Push 하기 위한 방식인 Comet를 대체

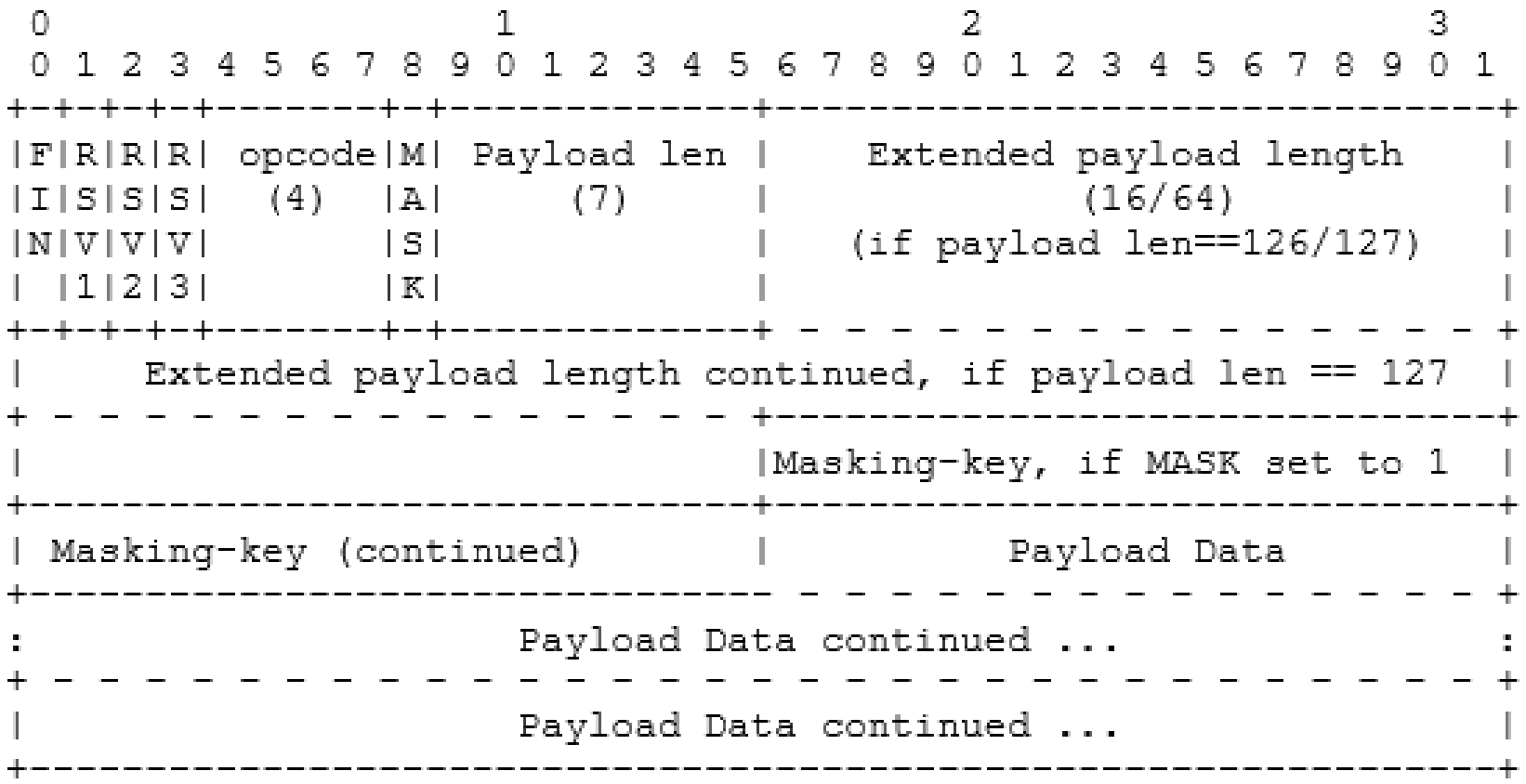
6.2 Web Socket

- Web Socket 과정



6.2 Web Socket

- Web Socket 메시지 포맷 (1/2)



6.2 Web Socket

Web Socket 메시지 포맷 (2/2)

- Opcode : 현재 프레임의 상태
- Mask : 해당 필드 값이 1일 경우 Payload Data 필드를 Masking-key 필드로 XOR 연산
- Payload len : Payload Data의 길이
- Payload Data : 데이터 값 필드

6.3 MQTT 실습

▪ MQTT 실습 목표

▪ MQTT 프로토콜을 이해를 돕기 위한 간단한 채팅 프로그램 구현

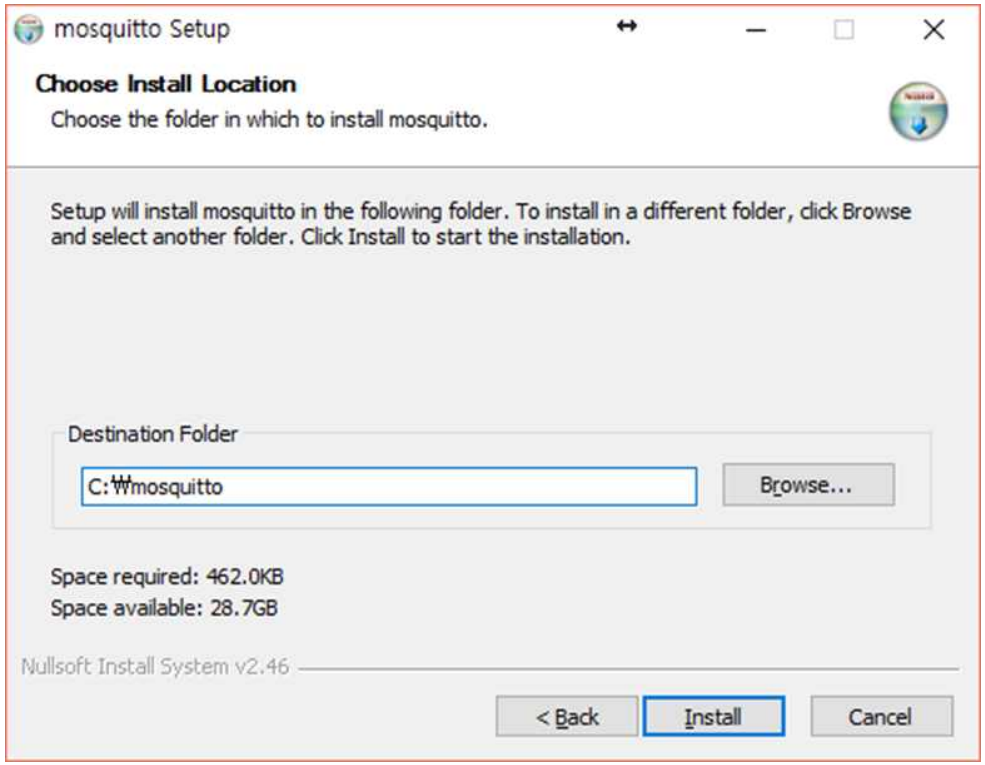
▪ 실습환경

- 윈도우 10 64bit
- Ruby 2.3.3
- Mosquitto 1.4.7
- (optional) Raspberry pi
- 첨부 설치파일

6.3 MQTT 실습

▪ MQTT 환경 설치 (1/5)

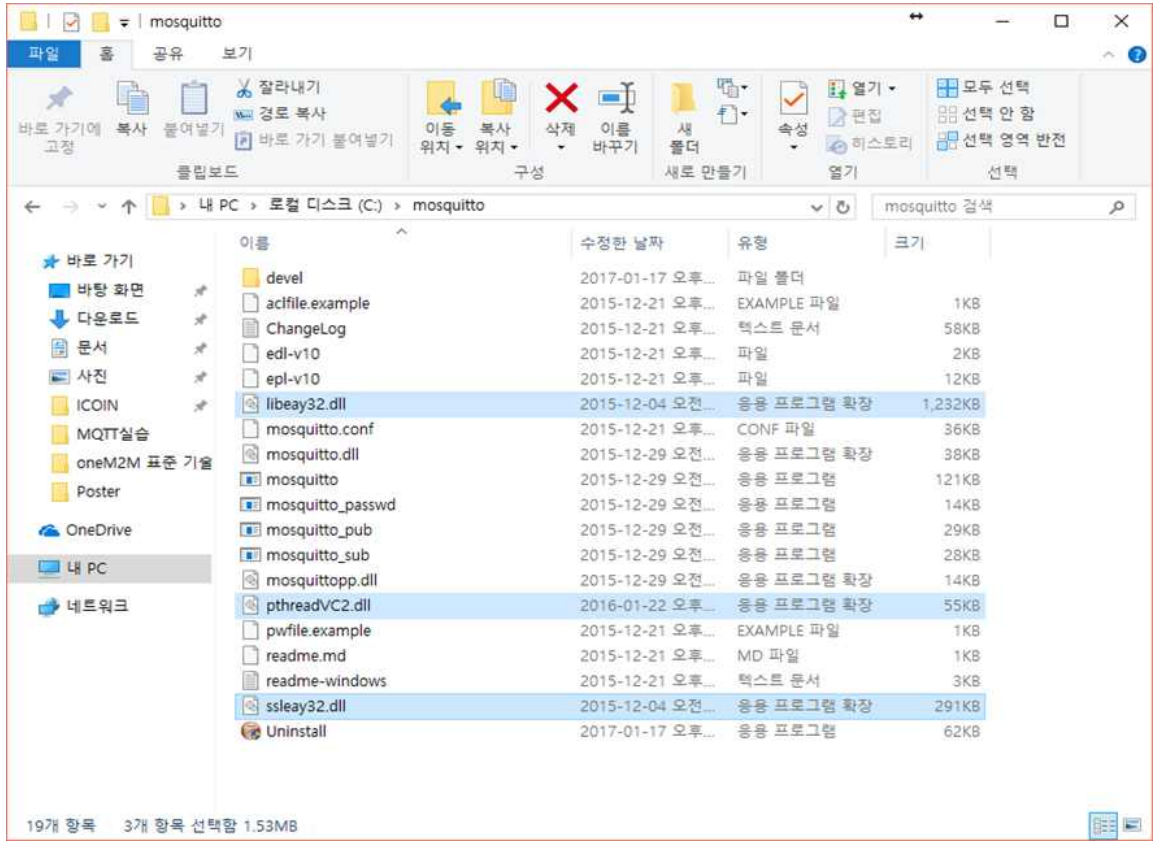
- 첨부파일 mosquitto-1.4.7-install-win32.exe 파일을 실행하여 mosquitto 설치, 설치 시 경로를 C:₩mosquitto 로 변경



6.3 MQTT 실습

▪ MQTT 환경 설치 (2/5)

▪ 설치 후 설치 경로로 이동하여 첨부파일 넣기 (libeay32.dll, ssleay32.dll, pthreadVC2.dll)

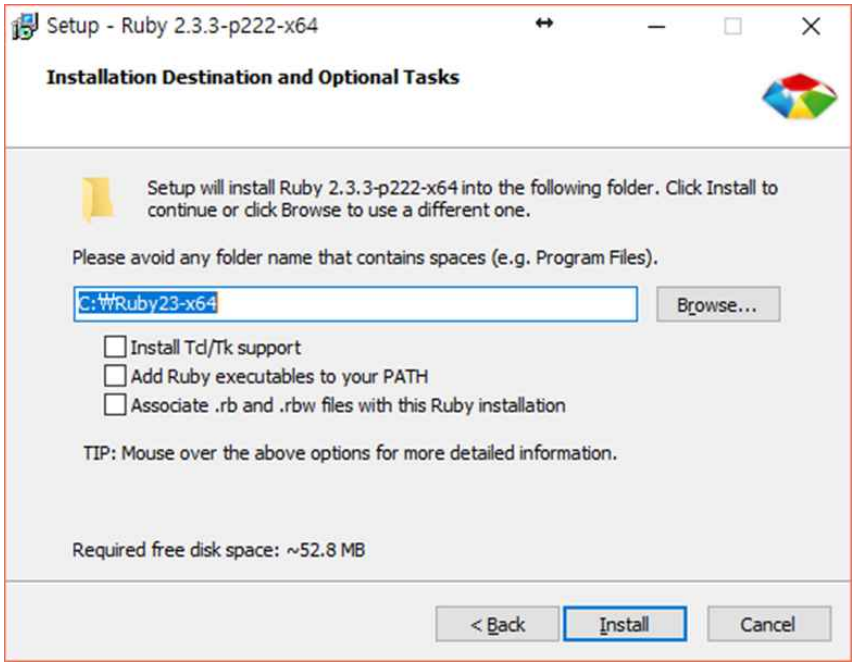


6.3 MQTT 실습

▪ MQTT 환경 설치 (3/5)

- MQTT 클라이언트 제작을 위한 Ruby 사용

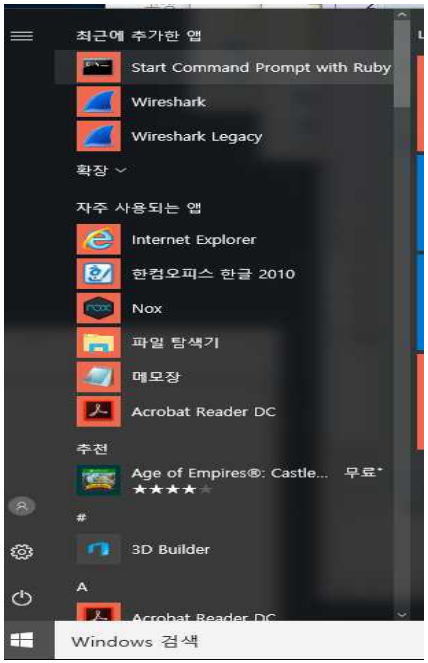
- 첨부파일 rubyinstaller-2.3.3-x64.exe 파일을 실행하여 ruby 설치

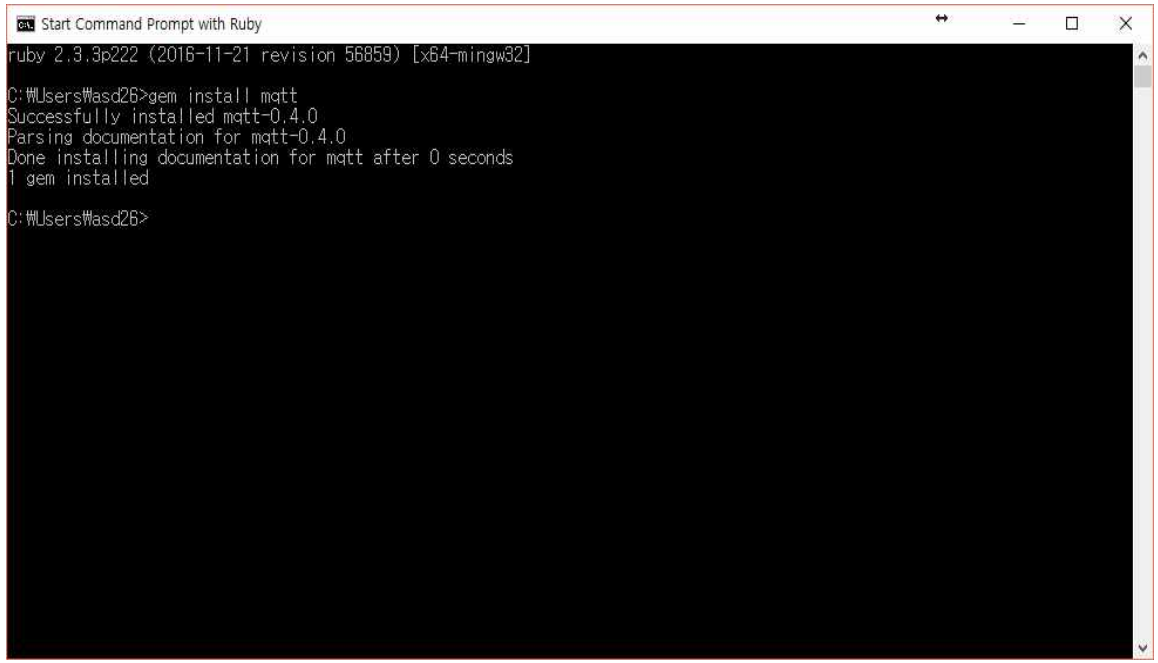


6.3 MQTT 실습

▪ MQTT 환경 설치 (4/5)

- Window에 설치된 Start Command Prompt with Ruby를 실행 , 다음 아래 그림 및 명령어를 사용하여 mqtt 라이브러리 설치
- gem install mqtt





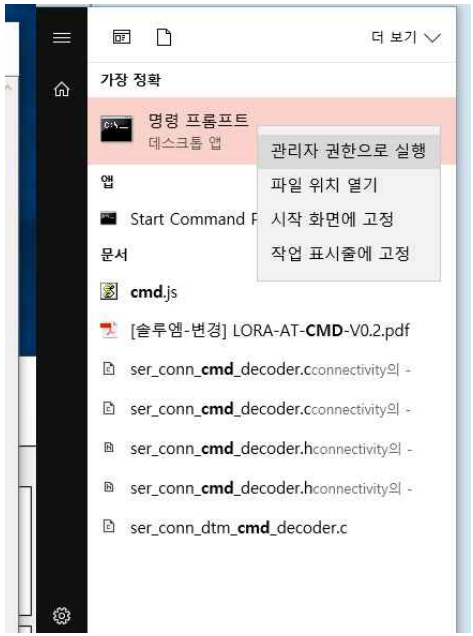
6.3 MQTT 실습

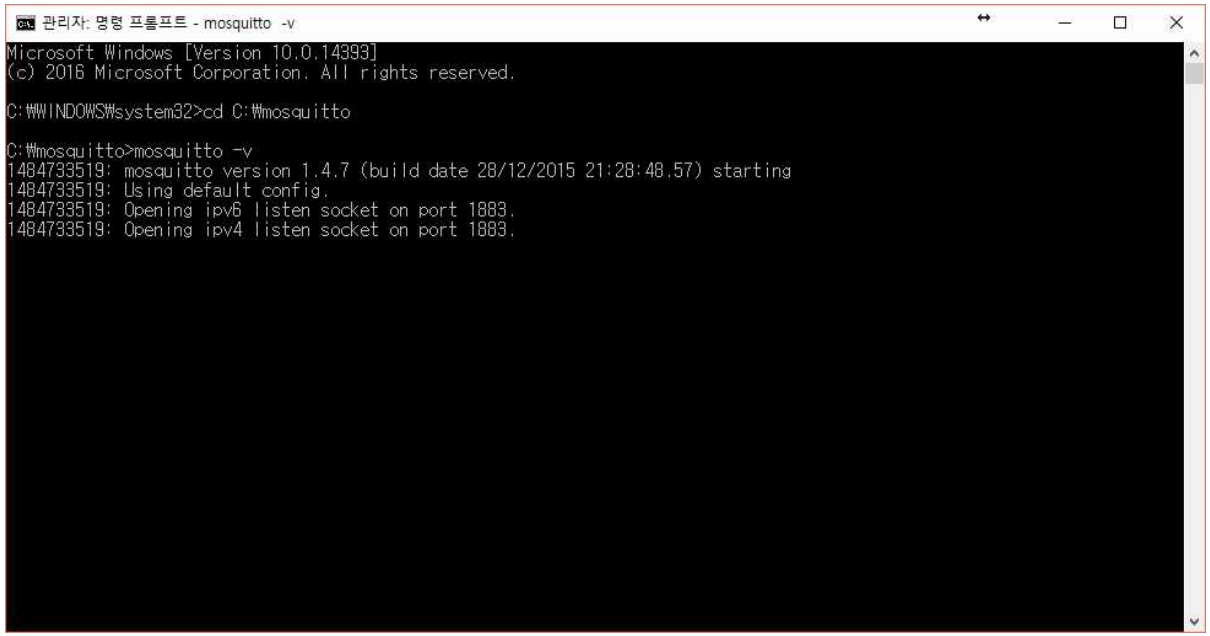
MQTT 환경 설치 (5/5)

- Window 용 ruby : http://rubyinstaller.org/downloads/
- libeay32.dll, ssleay32.dll : http://slproweb.com/products/Win32OpenSSL.html
- pthreadVC2.dll :
  ftp://sources.redhat.com/pub/pthreads-win32/dll-latest/dll/x86/
- MSVCR100.dll 에러 32 비트 :
  http://www.microsoft.com/download/en/details.aspx?id=8328
- MSVCR100.dll 에러 64 비트 :
  http://www.microsoft.com/download/en/details.aspx?id=13523

6.3 MQTT 실습

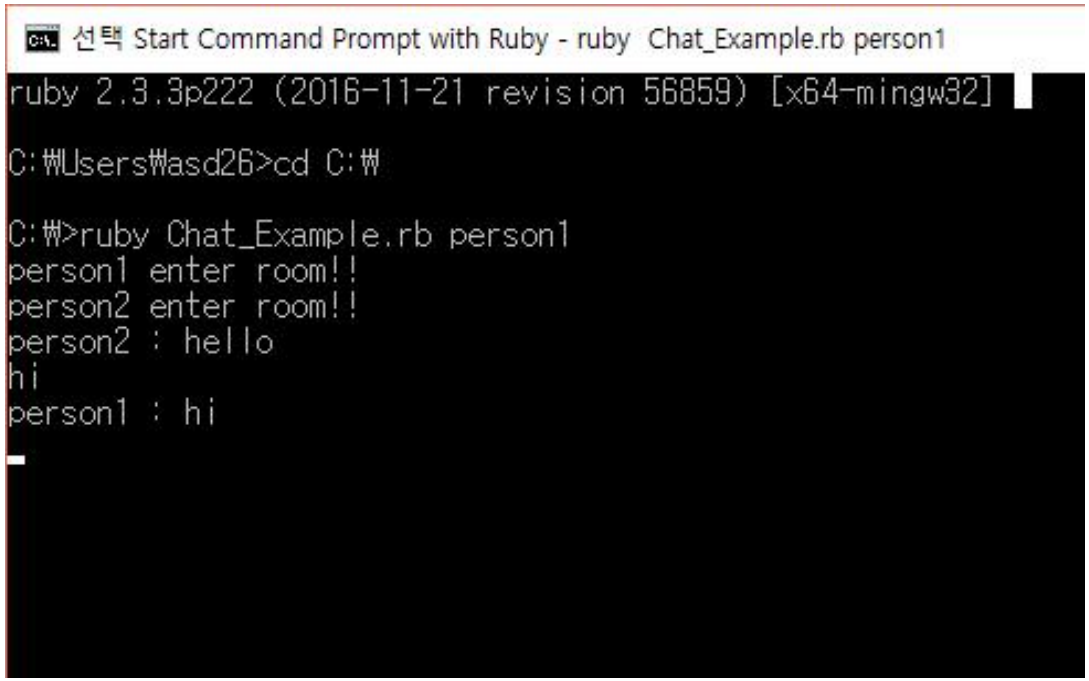
- MQTT Broker 서버 실행 (mosquitto)
  - **관리자 권한으로 명령 프롬프트 시행 후 아래 명령어를 사용하여 서버 실행**
  - cd C:₩mosquitto
  - mosquitto –v





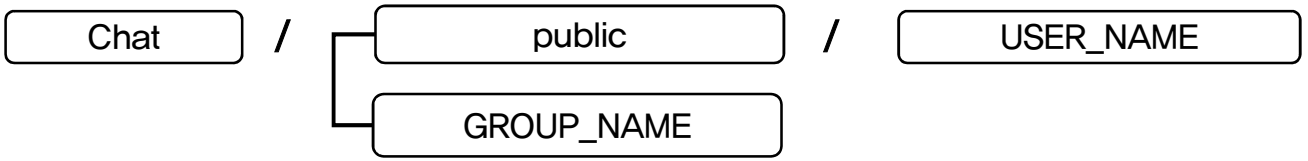
6.3 MQTT 실습

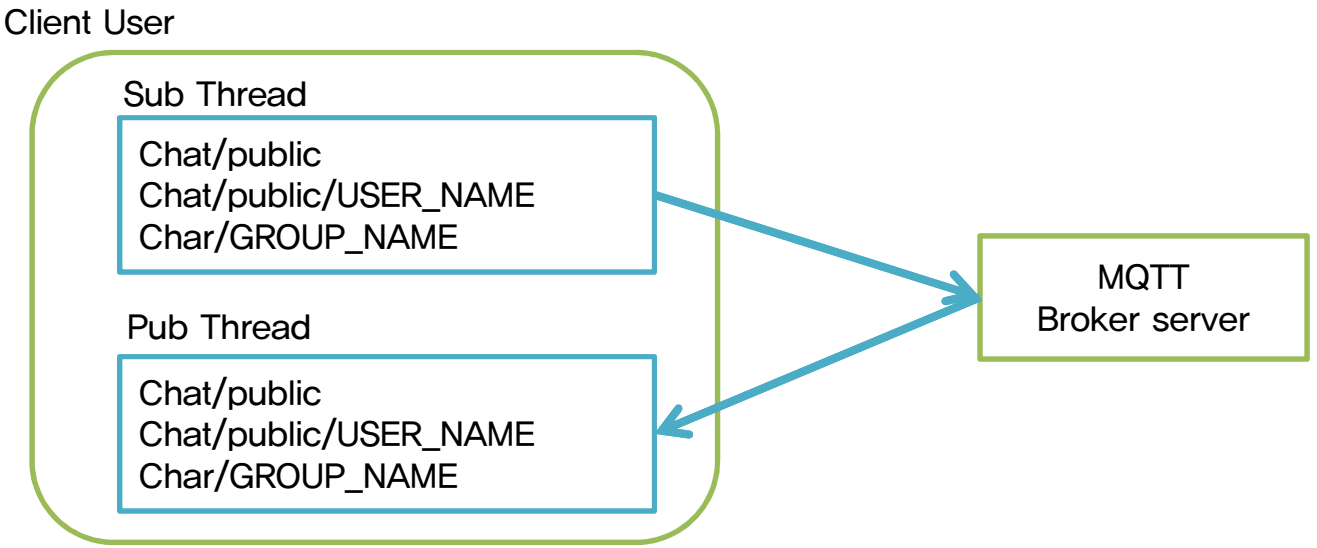
- MQTT Chatting Client 실행 (Ruby)
  - Start Command Prompt with Ruby 파일을 통해 창 실행 후 아래 명령어 사용
  - cd C:₩
  - ruby 파일명.rb 인자값



6.3 MQTT 실습

- Ruby를 사용한 MQTT Chatting Client 제작
  - MQTT 채팅 토픽 트리 및 시스템 구조도





6.3 MQTT 실습

▪ Ruby를 사용한 MQTT 라이브러리 API 사용법 (1/2)

▪ API 라이브러리 선언

• require 'rubygems'
require 'mqtt ' # 해당 코드 라인을 통해 mqtt 라이브러리를 로드

▪ MQTT 클라이언트 오브젝트 생성

• mqtt = MQTT::Client.new('MQTT broker ip 주소')

▪ MQTT 클라이언트 오브젝트를 사용한 접속

• mqtt.connect('test.mosquitto.org') do |client|
perform operations
end

6.3 MQTT 실습

▪ Ruby를 사용한 MQTT 라이브러리 API 사용법 (2/2)

▪ 발행 API (Publish) : 발행할 토픽 와 메시지를 입력

- client.publish topic, payload

▪ 구독 API (Subcribe) : 구독할 토픽을 인자 값으로 입력

- client.subscribe('topic1 ‘)
- client.subscribe('topic1', 'topic2')
- client.subscribe('foo/#')

▪ 메시지 받기 : 토픽과 해당 토픽의 메시지 수신

- 메시지 받기 : 토픽과 해당 토픽의 메시지 수신

6.3 MQTT 실습

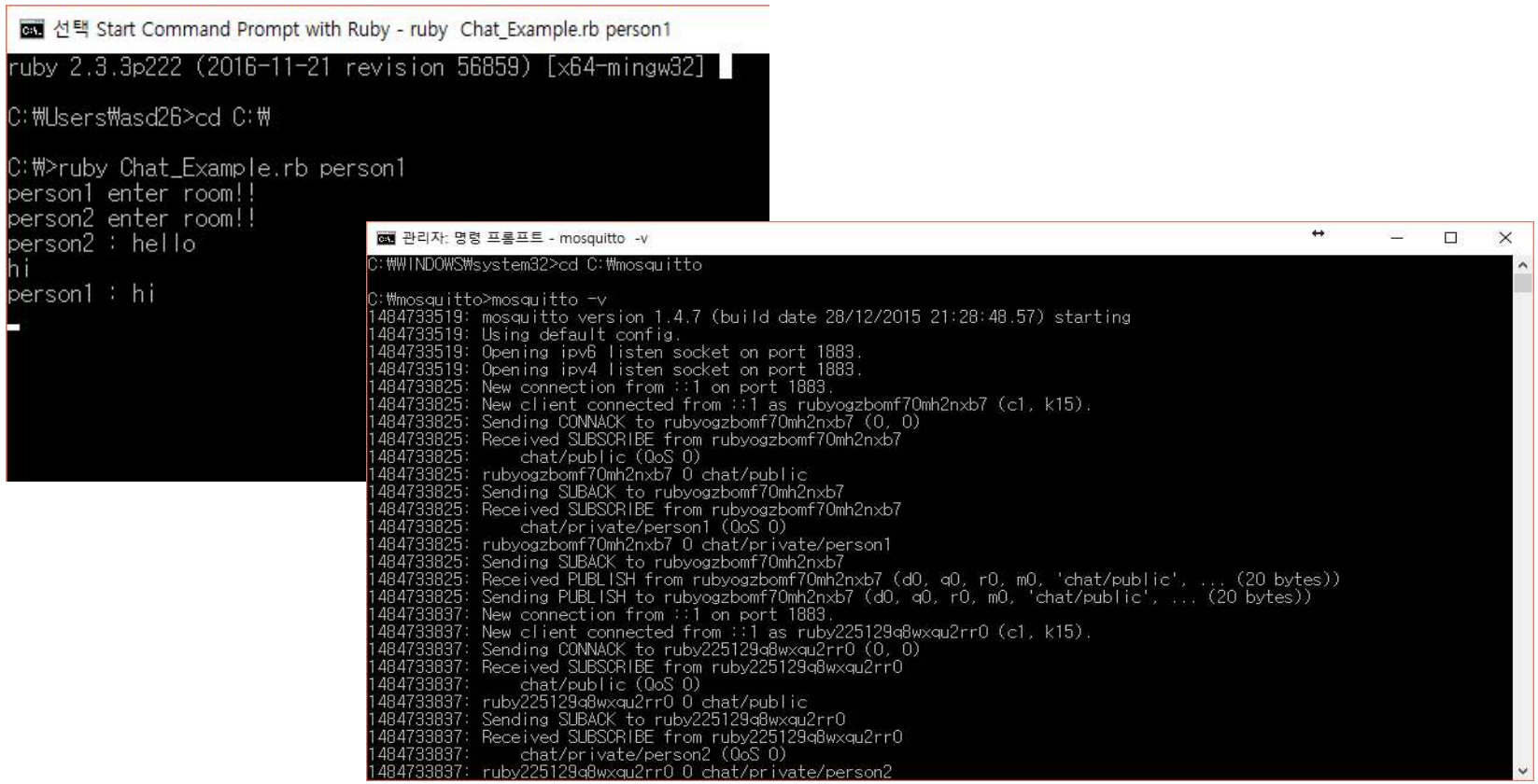
- MQTT 실습 예제 코드

```
var net = require('net');
var util = require('util');
var fs = require('fs');
var xml2js = require('xml2js');
var gpio = require('rpi-gpio');
var raspi = require('raspi');
var PWM = require('raspi-pwm').PWM;
var I2C = require('raspi-i2c').I2C;
var i2c;
var pwm;

var sh_timer = require('./timer');

var usecomport = '';
var usebaudrate = '';
var useparentport = '';
var useparenthostname = '';

var upload_arr = [];
var download_arr = [];
raspi.init(function() {
                    i2c = new I2C();
                    pwm = new PWM();
});
```

- MQTT 채팅 클라이언트 소스
- 귓속말: ₩w usr_name 메시지
- 나가기: ₩q 메시지
- 실행 방법 :
  ruby Chat_Example.rb 유저이름

6.3 MQTT 실습

- Ruby를 사용한 MQTT Chatting Client 실행 결과



6.3 MQTT 실습

▪ Raspberry pi 환경 MQTT Chatting Client 환경 설치 (1/4)

▪ 윈도우에서 mosquitto 서버 실행

▪ MQTT Chatting Client 예제 소스 변경

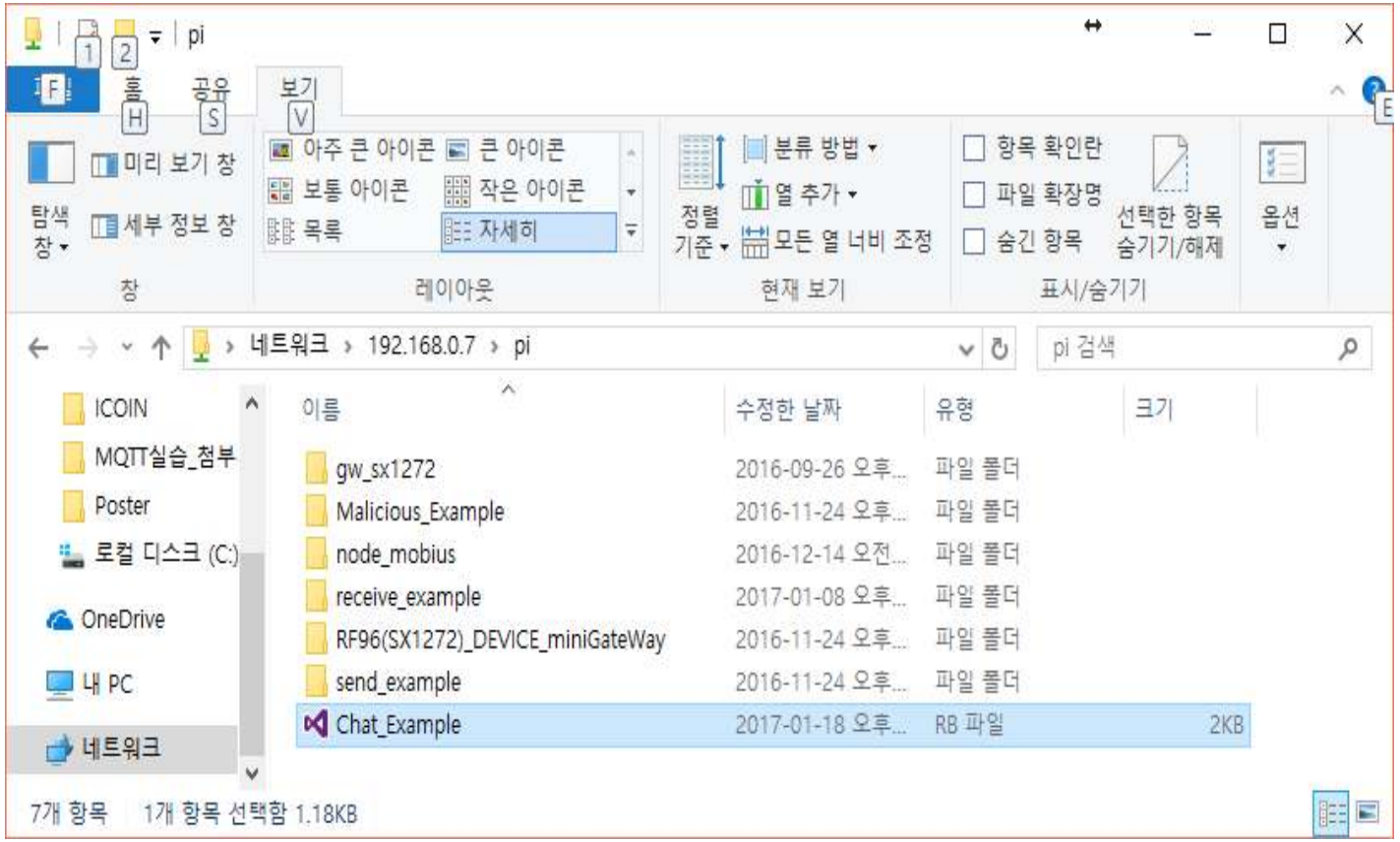
- 변경 전 : mqtt = MQTT::Client.new('localhost')
- 변경 후 : mqtt = MQTT::Client.new('mosquitto 서버 ip')

```
# 유저이름 usr_name
usr_name=ARGV[0]

# localhost내에서의 mqtt 서버에 접속
#mqtt = MQTT::Client.new('localhost')
mqtt = MQTT::Client.new('210.107.192.185')
```

6.3 MQTT 실습

▪ Raspberry pi 환경 MQTT Chatting Client 환경 설치 (2/4)

▪ Pi 의 home 폴더에 samba를 사용하여 ruby 소스 파일 저장



6.3 MQTT 실습

▪ Raspberry pi 환경 MQTT Chatting Client 환경 설치 (3/4)

- putty 또는 LCD 스크린을 사용해 Raspberry pi 접속 후 아래 명령어 사용

- sudo gem install mqtt
  - Rubygems 패키지를 통한 mqtt 라이브러리 설치

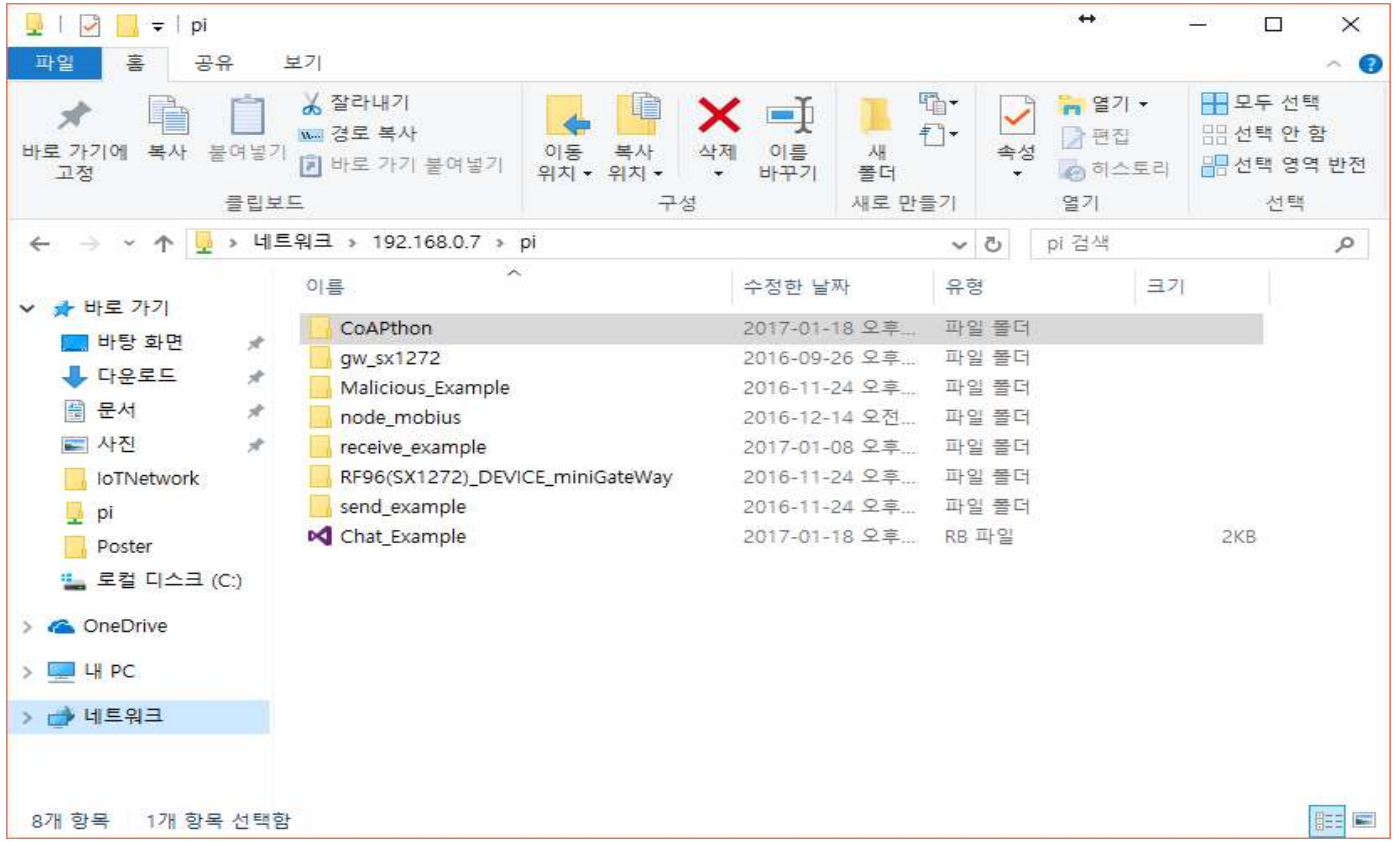
6.3 MQTT 실습

- Raspberry pi 환경 MQTT Chatting Client 환경 설치 (4/4)
  - 아래 명령어를 사용하여 MQTT Chatting Client 실행
  - sudo ruby ./파일명 유저이름

6.4 CoAP 실습

▪ CoAP 서버 설치 (1/3)

▪ 첨부 파일 CoAPthon 폴더를 파이 홈 폴더에 samba를 사용하여 저장

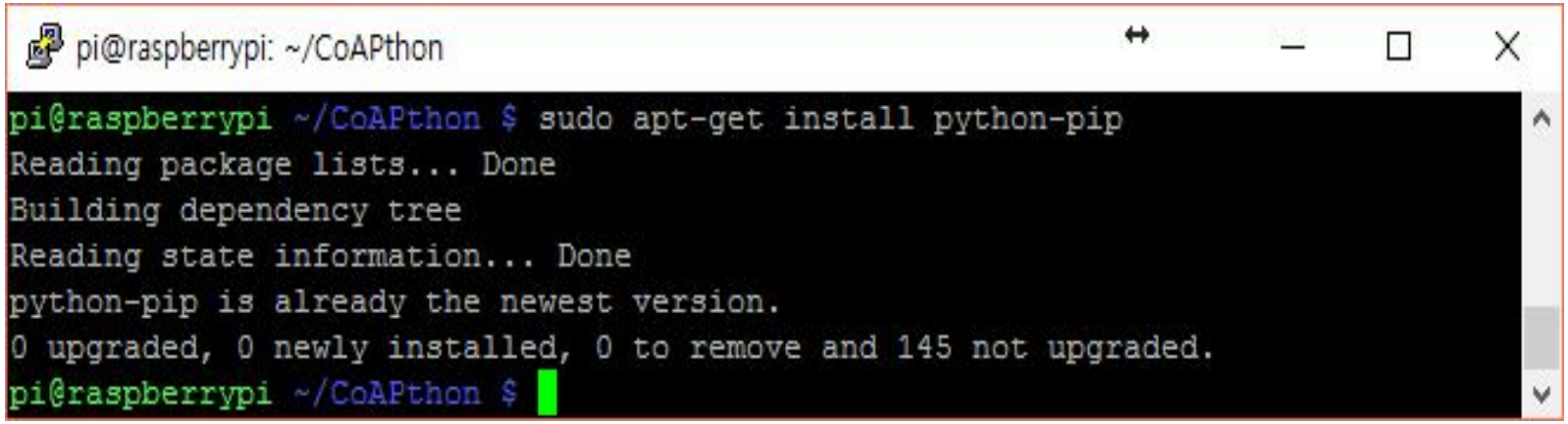


6.4 CoAP 실습

▪ CoAP 서버 설치 (2/3)

- putty 또는 LCD 스크린을 사용해 Raspberry pi 접속 후 아래 명령어 사용

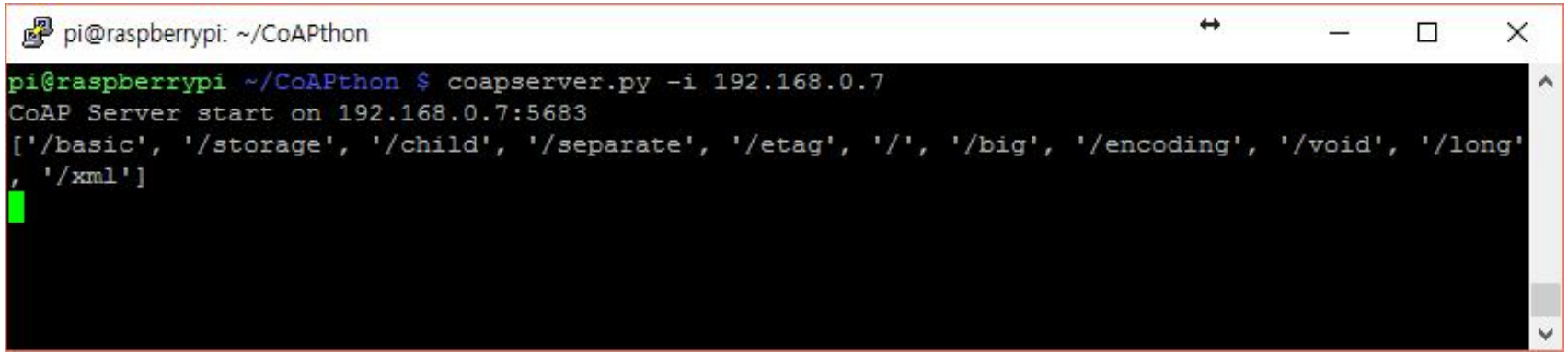
- sudo apt-get install python-pip
  - 설치 완료 시 아래 그림과 같이 출력



6.4 CoAP 실습

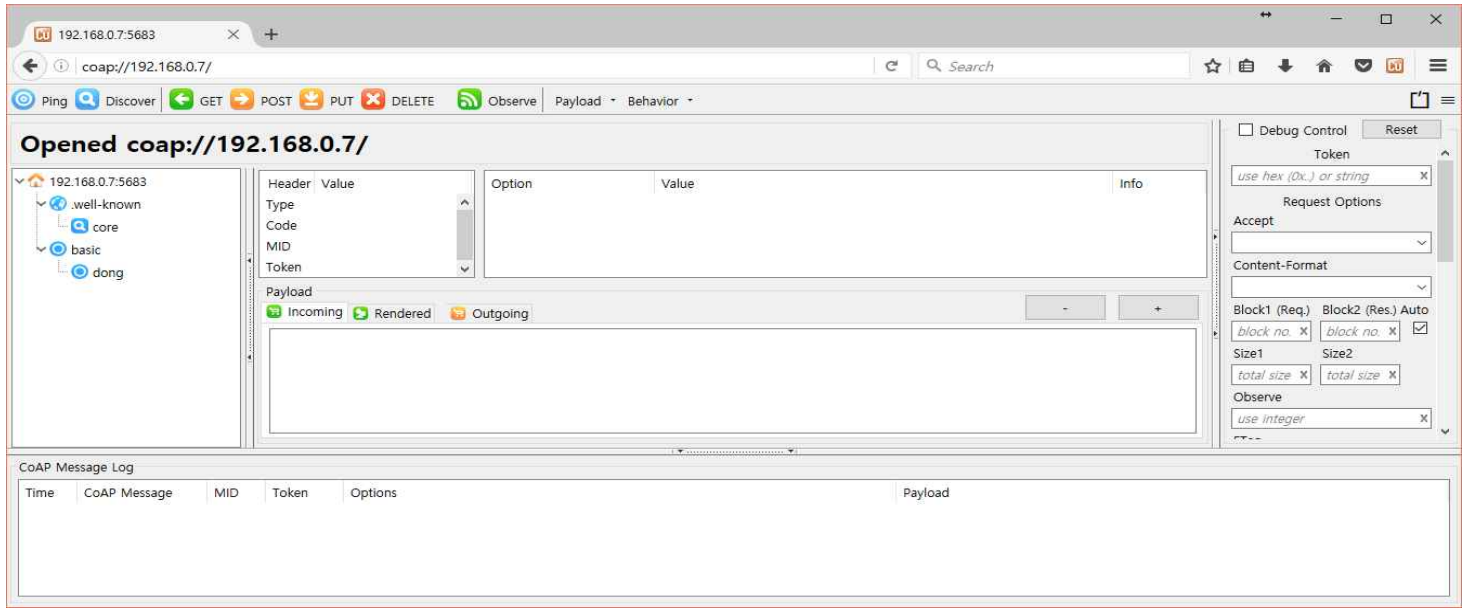
▪ CoAP 서버 설치 (3/3)

- 아래 명령어를 차례로 입력하여 CoAPthon 설치 후 아래 그림과 같이 CoAP 서버 실행
- cd CoAPthon
- sudo python setup.py sdist
- sudo pip install dist/CoAPthon-4.0.0.tar.gz -r requirements.txt
- coapserver.py -i 라즈베리파이ip



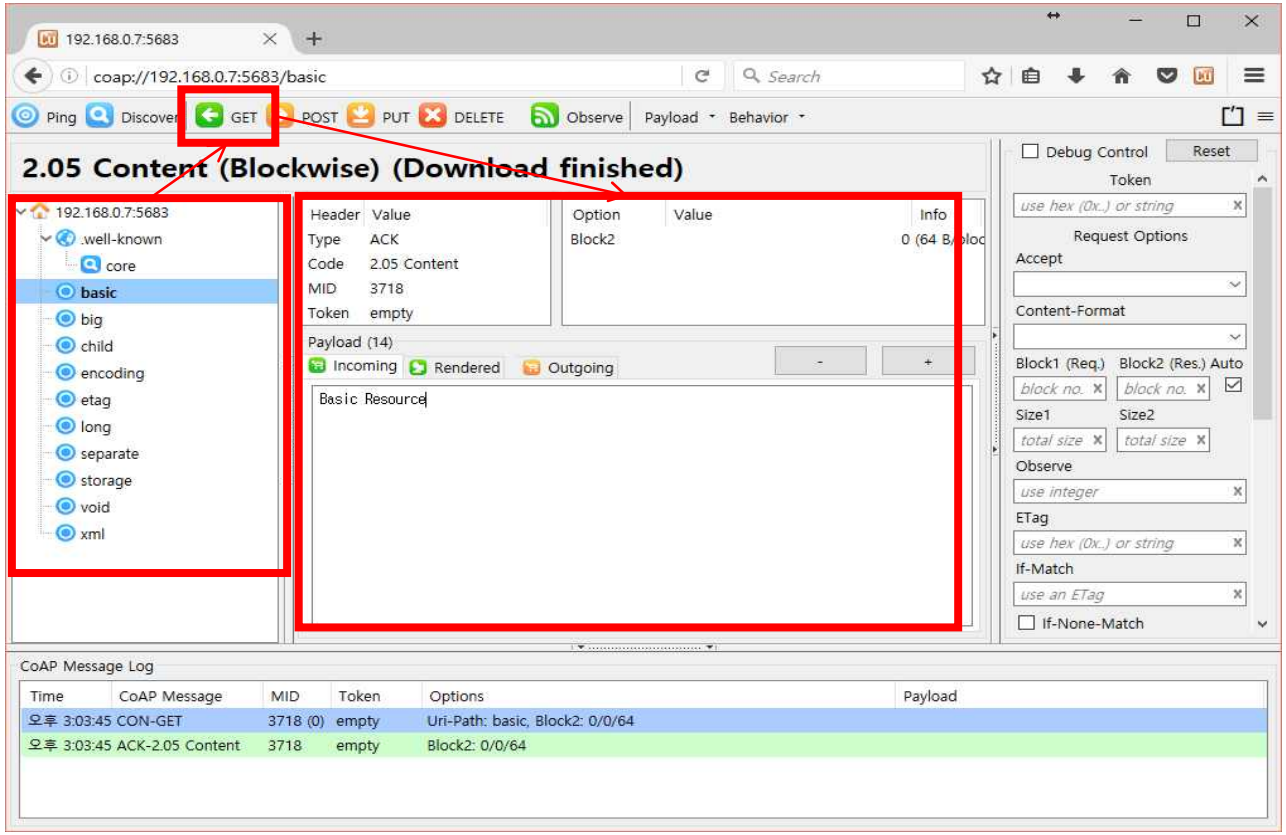
6.4 CoAP 실습

- 윈도우 CoAP 클라이언트 설치
  - Firefox 인터넷 브라우저를 설치
  - 아래 주소로 이동하여 Add-On 설치Copper 설치
  - https://addons.mozilla.org/en-US/firefox/addon/copper-270430/
  - 그림과 같이 주소 창에 “coap://라즈베리파이주소” 입력



6.4 CoAP 실습

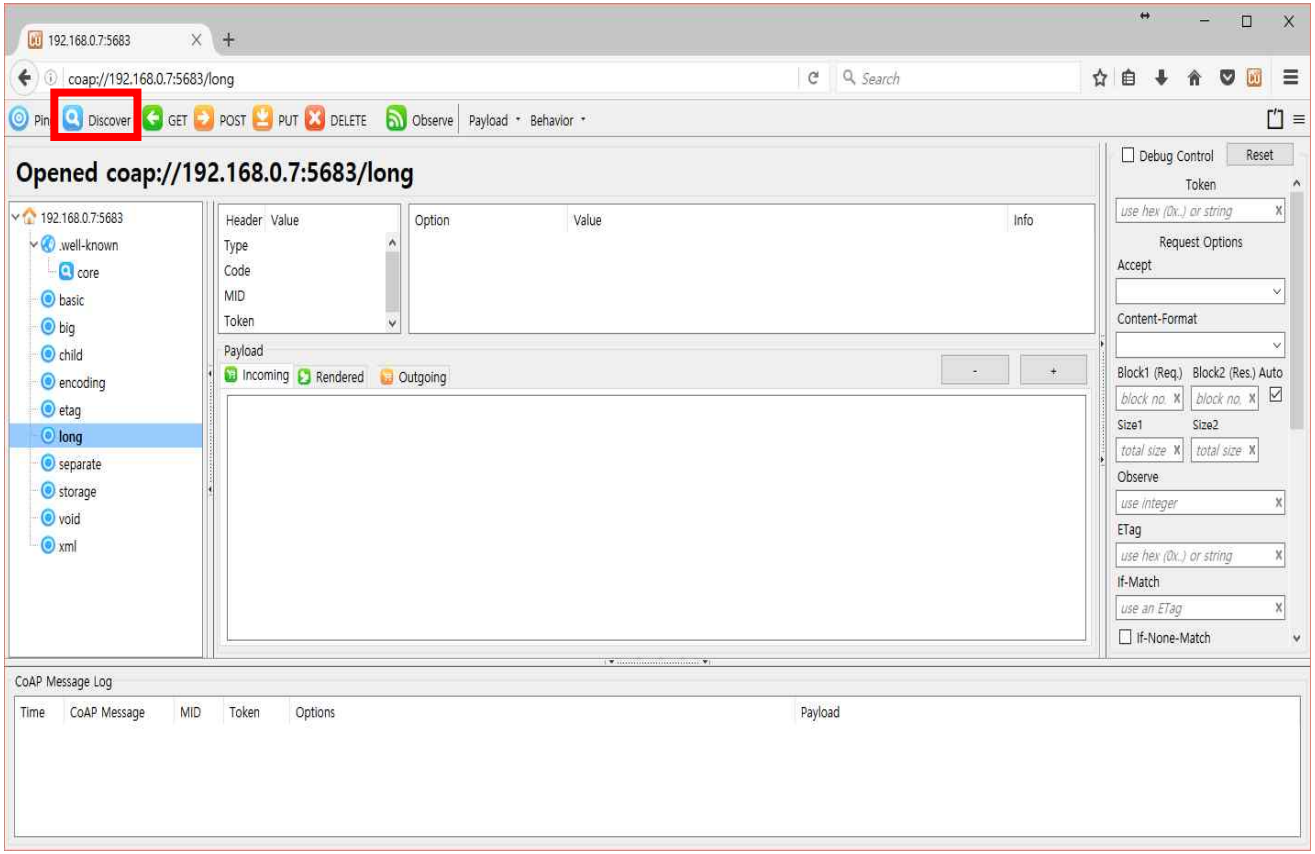
- 윈도우 CoAP 클라이언트 사용 (1/5)
  - GET : 리소스 목록에서 선택 후 GET 버튼으로 리소스 조회



6.4 CoAP 실습

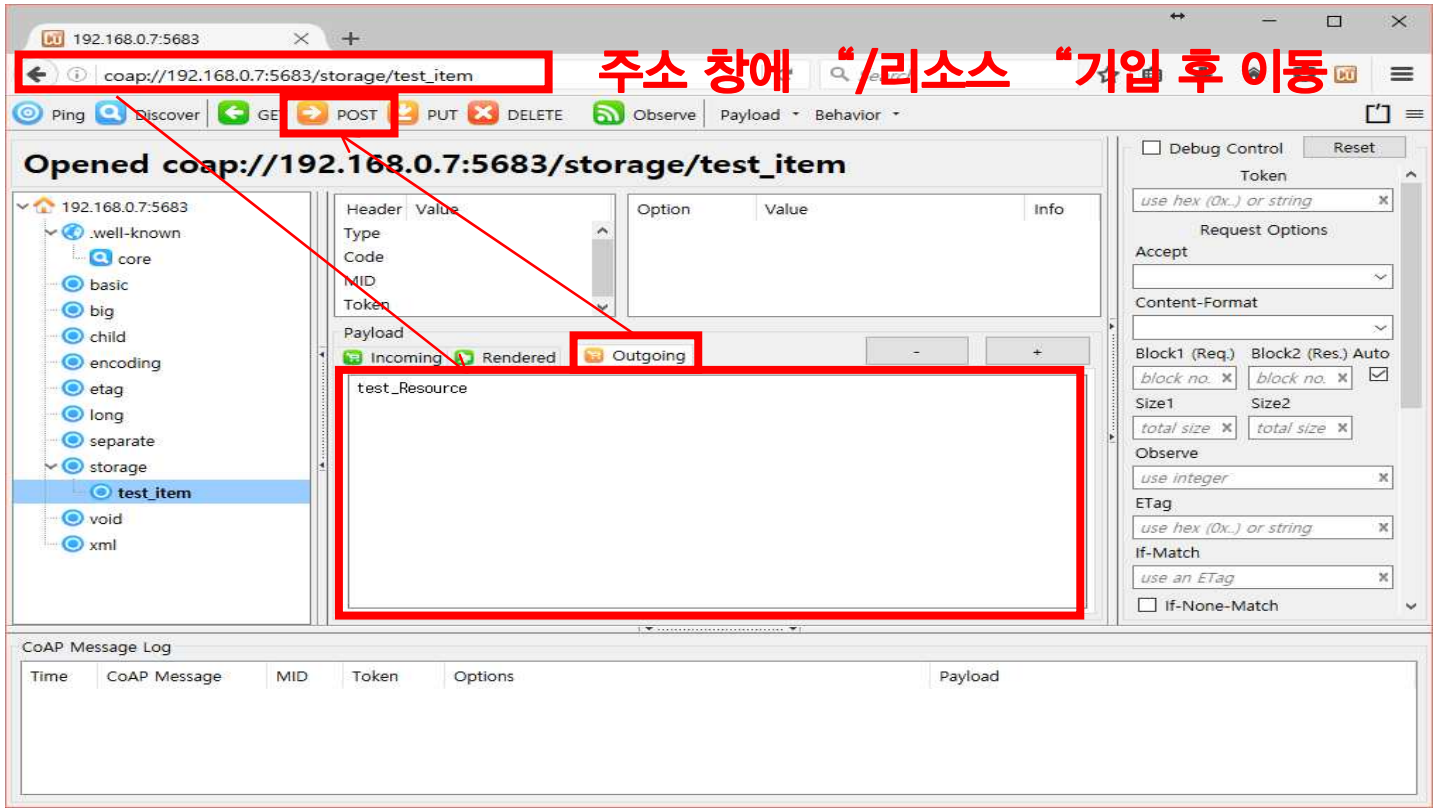
▪ 윈도우 CoAP 클라이언트 사용 (2/5)

▪ Discover 버튼을 사용하여 Resource list를 갱신



6.4 CoAP 실습

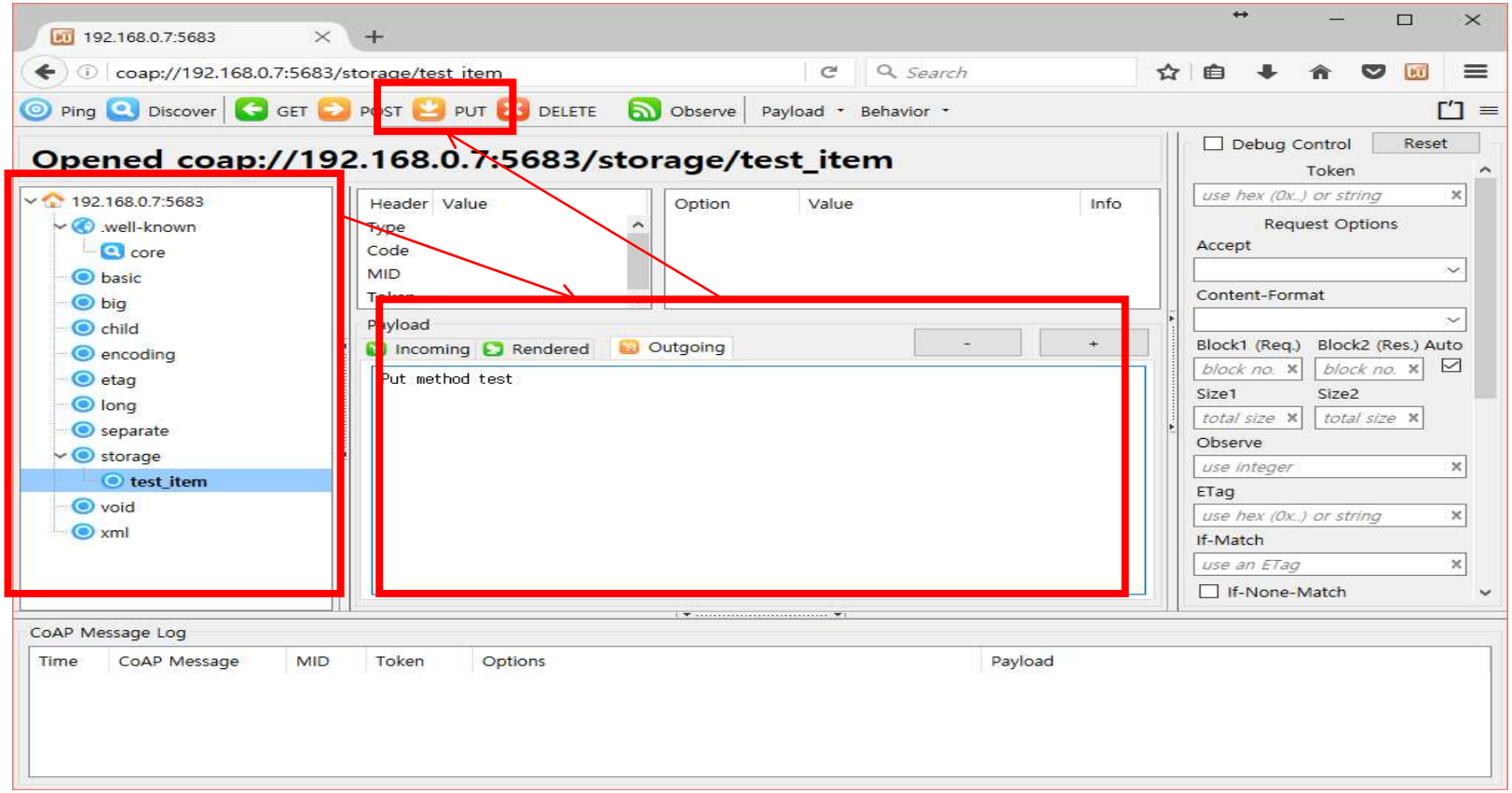
- 윈도우 CoAP 클라이언트 사용 (3/5)
  - POST : URI에 리소스 갱신 후 outgoing 탭을 통해 페이로드를 적고 POST 버튼을 통해 리소스 저장



6.4 CoAP 실습

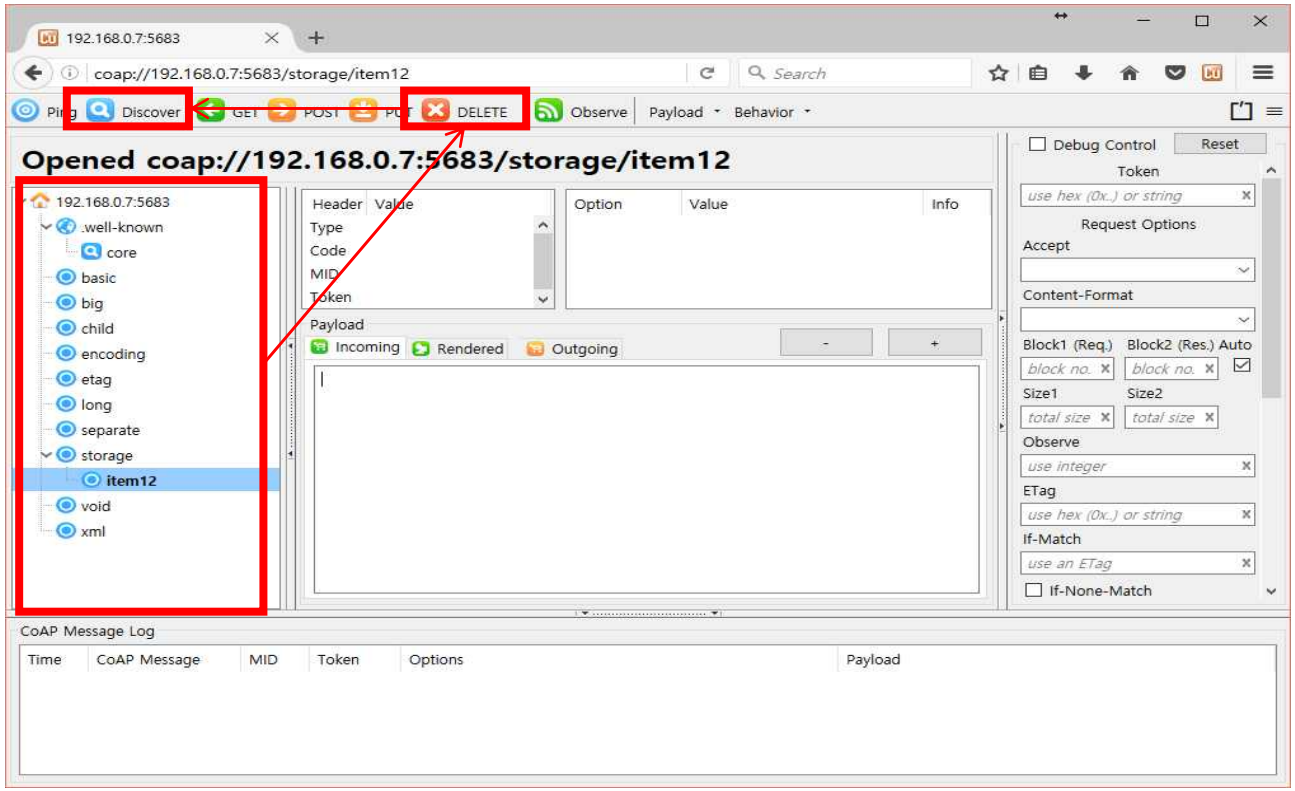
▪ 윈도우 CoAP 클라이언트 사용 (4/5)

▪ PUT : 리소스 선택 후 Outgoing 탭에 수정 사항 입력 후 PUT 버튼을 선택하려 리소스 수정



6.4 CoAP 실습

- 윈도우 CoAP 클라이언트 사용 (5/5)
  - DELETE : 리소스 선택 후 DELETE 버튼을 사용하여 삭제 후 , Discover 버튼을 사용하여 리소스 리스트 갱신



Assignment

- Assignment 6

실습을 통해 설치한 Firefox의 Add-On인 Copper와 같은 기능을 하는 프로그램을 작성해 보시오. (프로그래밍 언어에 상관없이 GET, POST, PUT, DELETE 기능을 CoAP 프로토콜을 이용하여 구현)

Reference

[6.2] Web Socket

Web Socket 메시지 포맷

그림 출처 : https://tools.ietf.org/html/rfc6455

7장. oneM2M 기술 및 플랫폼 소개

Chapter 7. oneM2M Technology and Introduction of Platform

7.1 Organization
7.2 Specifications
7.3 Market Adoption
7.4 Service Platform
7.5 RESTful API

- Assignment
- Reference

강의 개요

▪ 강의 목표

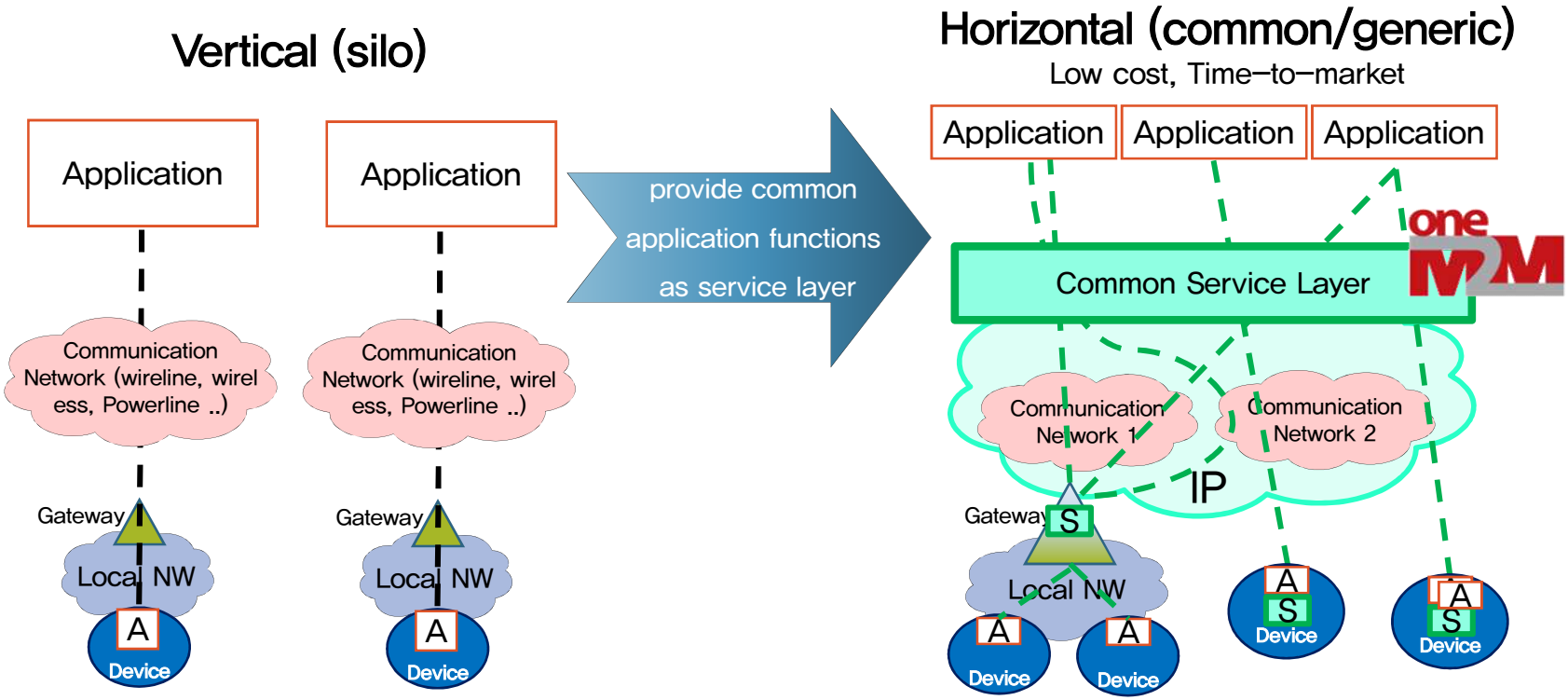
ononeM2M 기술의 기본 개념과 서비스 플랫폼으로서의 특징을 학습한다.

▪ 강의 내용

- oneM2M 기술의 기본 개념
- oneM2M 표준의 구성
- 서비스 플랫폼의 정의
- RESTful 구조와 oneM2M

7.1 Organization

- Goal
  - To have one common/standard platform
    - that is generic to different IoT service domains
    - so that the IoT market can be expanded

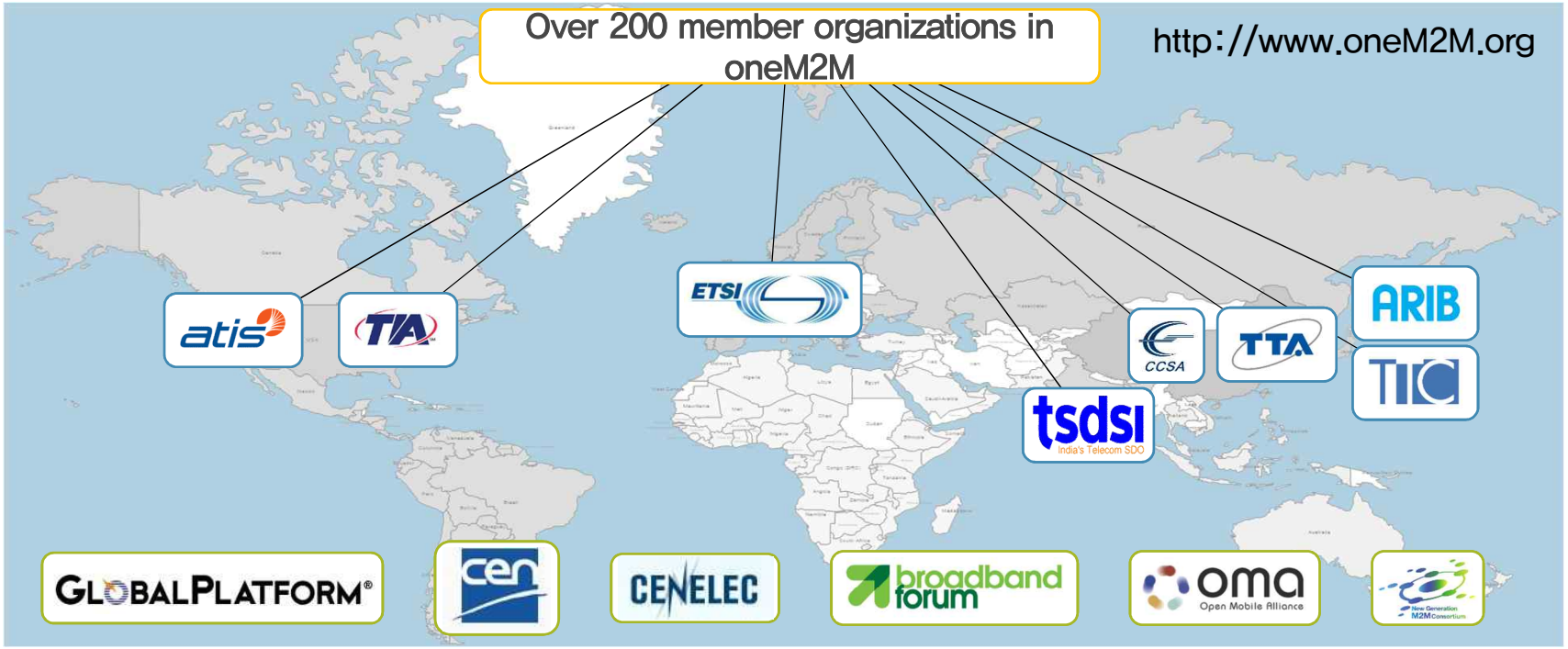


7.1 Organization

▪ Aim to Global Standard

▪ Partnership project among regional SDO1)s

- SDOs of North America, Europe, Korea, Japan, China and India
- not to develop different/fragmented IoT platform standards



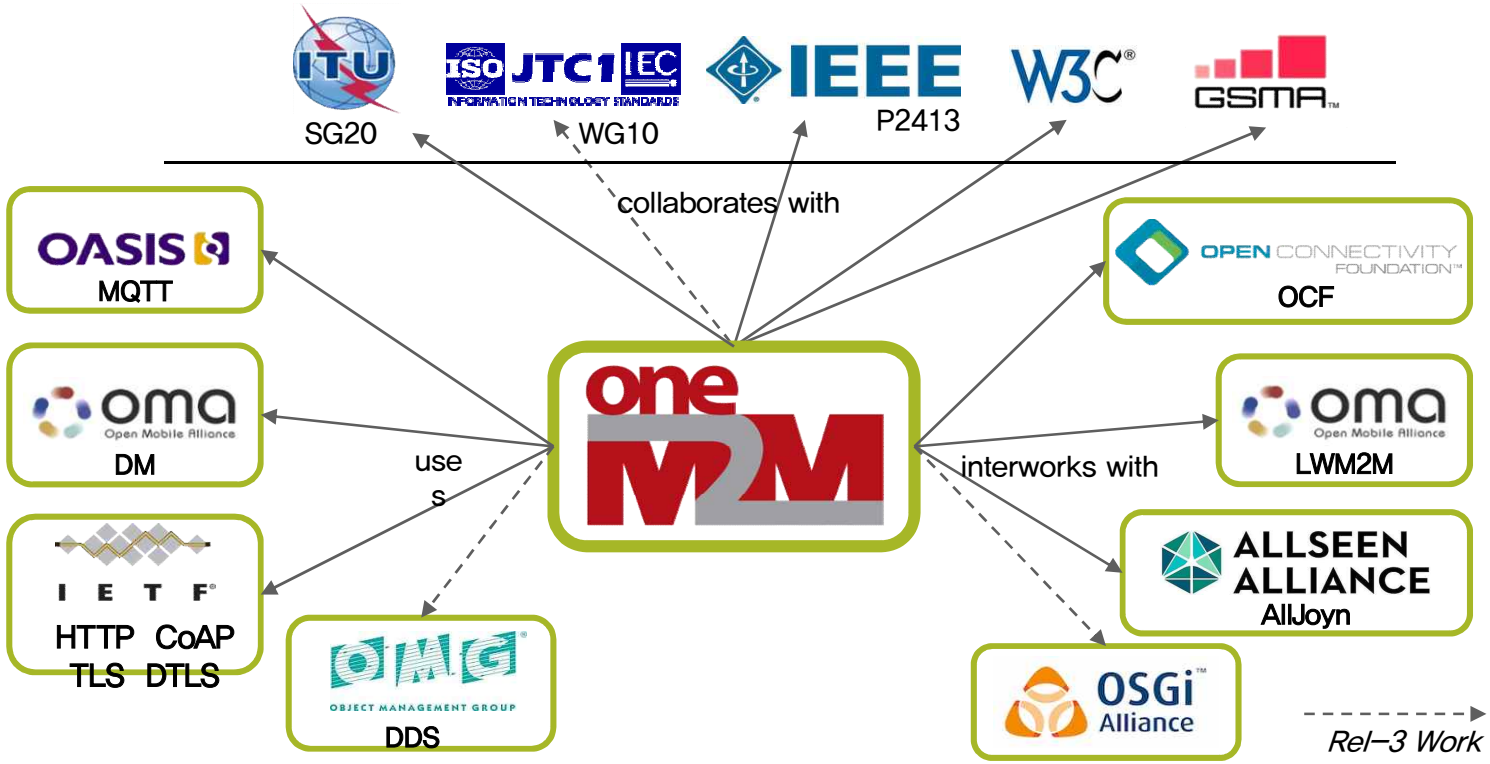
1) Standard Development Organization

7.1 Organization

▪ Collaboration

▪ Collaboration with global SDOs

• Interworking with existing systems and utilization of existing protocols



7.1 Organization

▪ Members

▪ IoT ecosystem stakeholders

- Operators, Venders, Solution Providers and Institutes
- 237 members (Aug–2016)

http://www.oneM2M.org

7.1 Organization

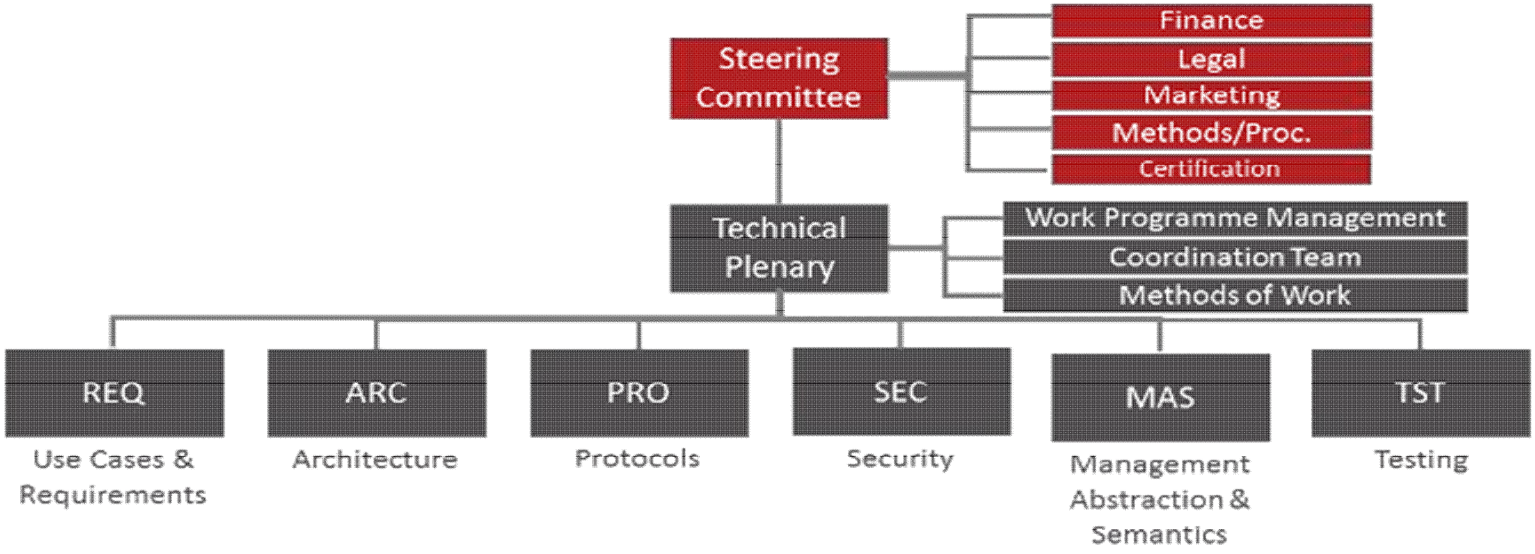
Structure

Technical Plenary

- Working Group(WG)으로 구성되며 표준 규격 개발 관리

Steering Committee

- 법률, 회계, 마케팅 등 기관 운영 관리



http://member.oneM2M.org

7.2 Specifications

Deliverables (1/2)

WG1(Requirements)

- Use Case TR1), Requirements TS2), Terminology TS

WG2(Architecture)

- Functional Architecture TS,
- LWM2M Interworking TS, AllJoyn Interworking TS, OCF Interworking TS

WG3(Protocol)

- Core Protocol TS
- HTTP Binding TS, CoAP Binding TS, MQTT Binding TS, WebSocket Binding TS

7.2 Specifications

Deliverables (2/2)

WG4(Security)

- Security Solutions TS, Secure Environment Abstraction TS

WG5(MAS)

- OMA Management Enablement TS, BBF Management Enablement TS,
- Home Appliances Information Model and Mapping TS, Base Ontology TS

WG5(Testing)

- Testing Framework TS
- Interoperability Testing TS
- Implementation Conformance Statements TS, Test Suite Structure & Test Purposes TS, Abstract Test Suite & Implementation eXtra information for Test TS

7.2 Specifications

▪ Snapshot of Releases

Minimum Deployable Solution
- common service functions (Data mgmt., Device mgmt., Group, etc.)
- security
- protocol binding (HTTP, CoAP, MQTT)

Interworking
- AllJoyn
- OCF
- OMA LWM2M
- 3GPP Rel-13

Enhanced Security
- end-to-end security
- dynamic authorization

Semantic Support
- base ontology
- semantic discovery

Protocol Binding
- WebSocket

Information Model
- Home Appliances Model

Market Adoption
- Developer Guide Series
- Product Profiles
- Platform Feature Catalog

Interworking
- 3GPP Rel-13/14
- Proximal IoT
- OPC-UA

Enhancements
- Distributed Authorization
- Semantic Support
- LWM2M Interworking

Testing Spec.
- security conformance
- Rel-2 interoperability

7.3 Market Adoption

Commercialization

Domestic

- Operator IoT platforms deployed oneM2M
  - SKT – ThingPlug(Smart home and factory)
  - KT – IoTMakers
  - LG – U+(Kid’s band 로직만을 포함하는 제한된 기능을 가지는 제한적 디바이스

Abroad

- Solution vendor is selling oneM2M solution
  - InterDigital
- Server and chipset vendors deployed oneM2M
  - HP, Qualcomm
- And more are coming at this moment

7.3 Market Adoption

▪ Open Sources

▪ Available 0pen sources (Jan-2016)

	OM2M Connecting things	OPEN DAYLIGHT IotDM	OCEAN
Lead	LAAS-CNRS	CISCO	KETI
Homepage	http://www.eclipse.org/om2m	https://wiki.opendaylight.org/view/IoTDM:Main	http://www.iotocean.org/
License	Eclipse (EPL 1.0)	Eclipse 1.0	3 Clause BSD
Version	Rel-1 and Rel-2(partial)	Rel-1 and Rel-2(partial)	Rel-1 and Rel-2(partial)
Binding	HTTP, CoAP(unstable)	HTTP, CoAP	HTTP, MQTT
Format	XML, JSON(unstable)	JSON	XML, JSON
Language	Java	Java	Java, Node.js
Interworking	(Prototype) KNX, ZIgBee, HUE, LoRa, SigFox, etc	ZigBee	AllJoyn, OCF, Nest, ZigBee, FIWARE, Jawbone

7.3 Market Adoption

Pilot Projects

Domestic

- Busan smart city[1)]
  - Parking, street light, crosswalk, building, city monitoring, marine safety, big data
- Daegu smart healthcare
- Goyang smart city (coming)

Abroad

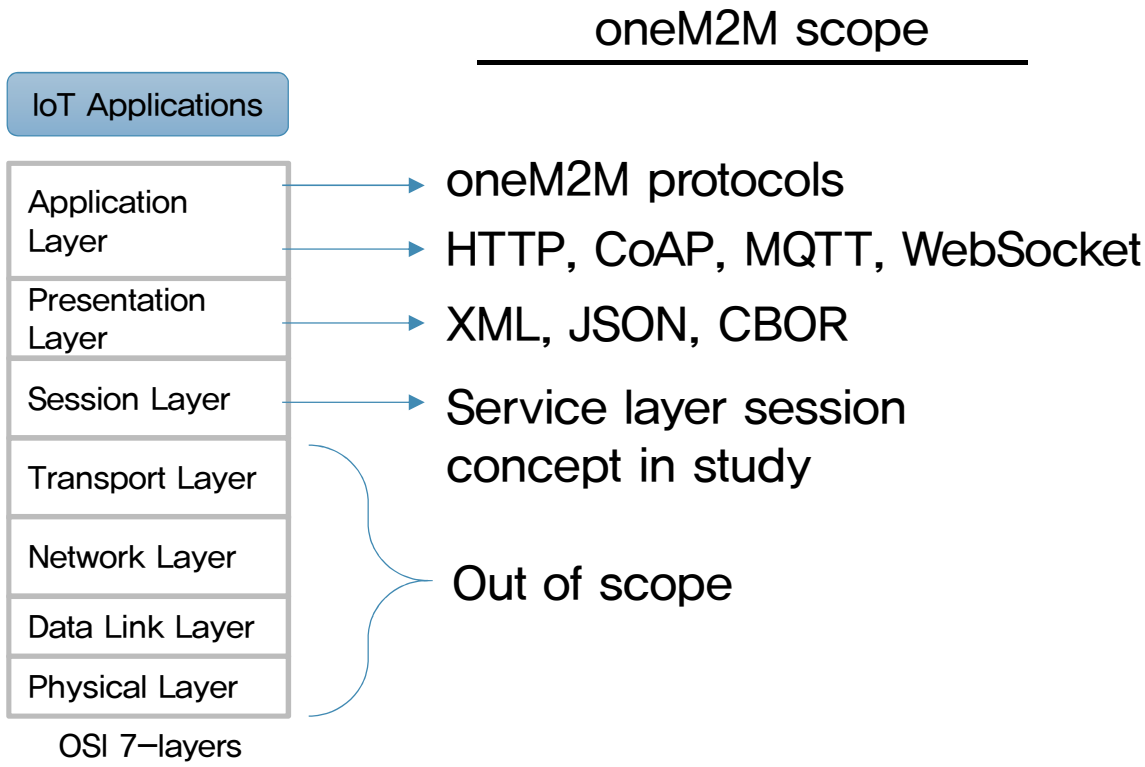
- UK oneTransport
  - Traffic management system

7.4 Service Platform

▪ Definition (1/2)

▪ Difficult to explain in OSI 7-layer model

• It focuses on traditional communication protocol stack concept

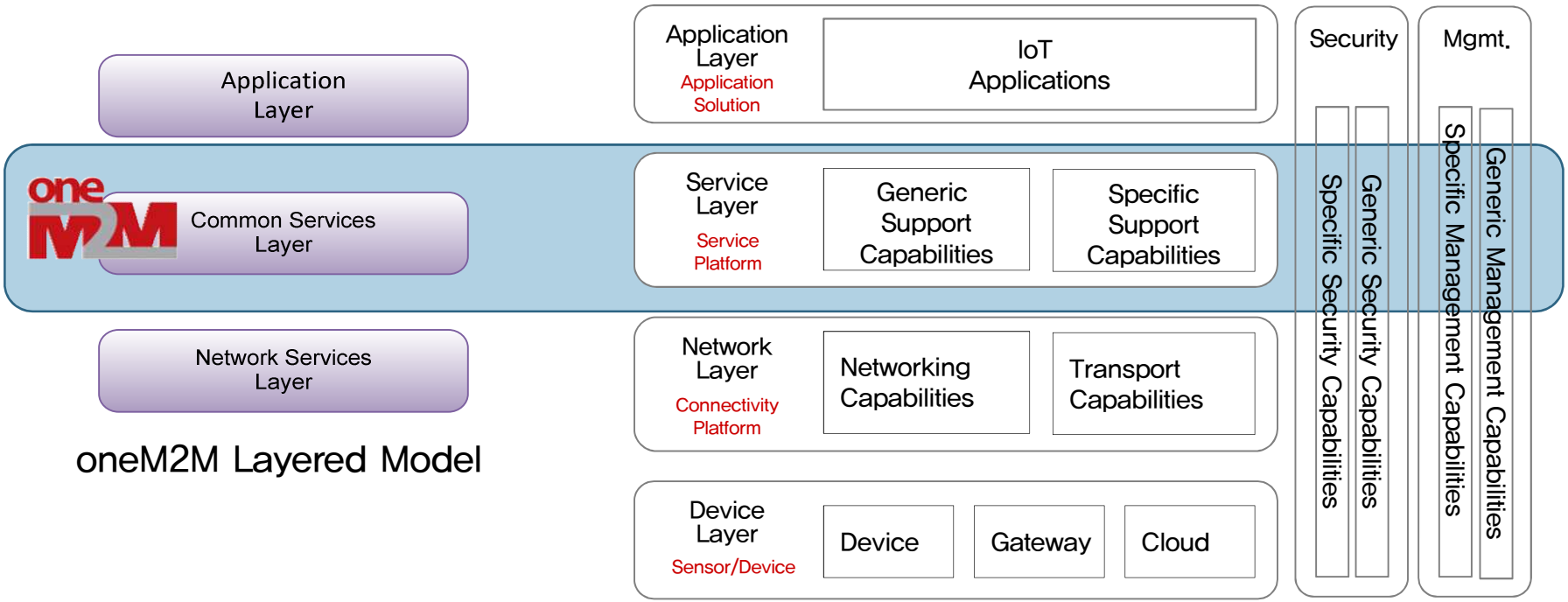


7.4 Service Platform

▪ Definition (2/2)

▪ Well explained in ITU-T IoT Reference Model

• oneM2M platform as a service platform provides common services, supporting capabilities to IoT applications

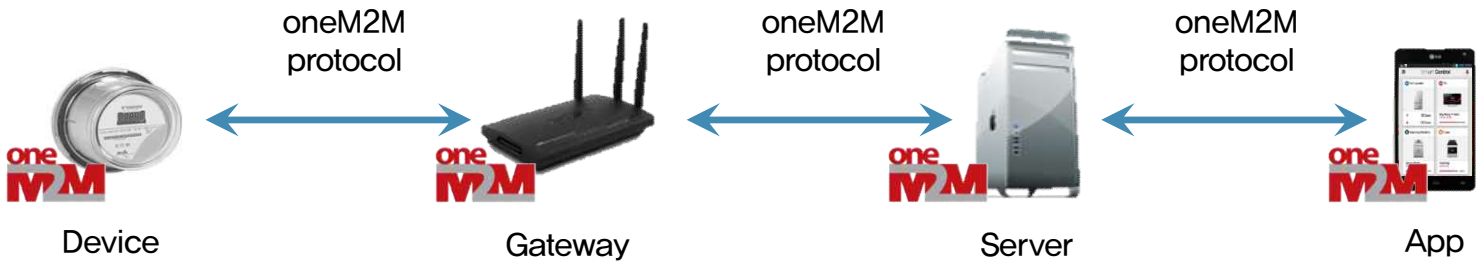


7.4 Service Platform

▪ Device and Server Platform

▪ oneM2M is generic service platform

- that applies to device, gateway and server
  - server specific functions are defined
- that provides standard/common APIs to applications
- stores data that is consumed by application
- is transport agnostic
  - no dependency to any specific messaging protocol (e.g. CoAP)
  - support both TCP and UDP
  - basically over IP

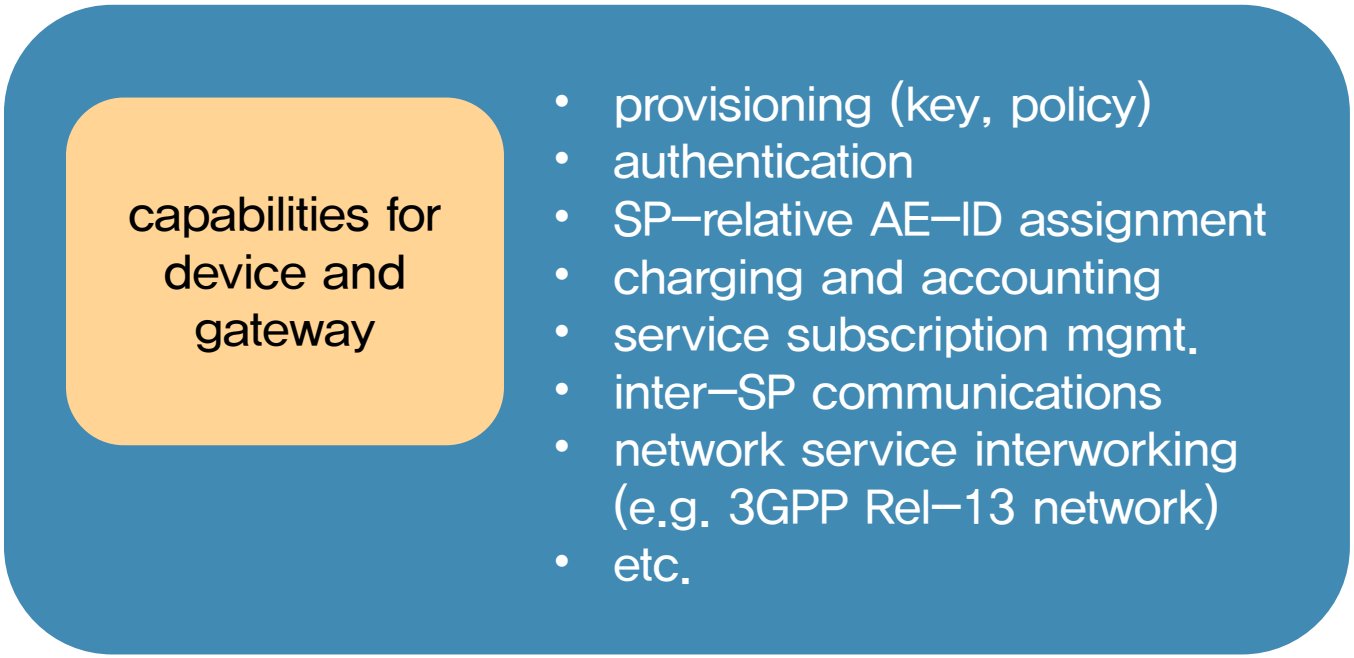


7.4 Service Platform

▪ Server Platform

▪ oneM2M server capabilities

- extend device/gateway capabilities to have server specific ones
- that are needed by an M2M/IoT Service Provider(SP) to run their systems



7.5 RESTful API

- RESTful
  - Web is RESTful
    - GET http://www.google.com HTTP/1.1
    - oneM2M follows the basic principles of RESTful

RESTful API		oneM2M API	
Resource	• Anything that can be identified by URI	Resource	• Resource as a service/function/data
+		+	
URI	• Unique identifier of a resource	Resource Address	• Resource addresses in different formats and different addressing schemes
+		+	
Method	• resource access (CRUD)	Operation	• resource access(CRUD) and Notify

7.5 RESTful API

oneM2M Resource

Resource

- stores data or represents a function/service
- consists of attributes and child resources
- each attribute represents data or configuration of function/service

Resource type vs. resource (instance)

- Each resource is instantiated in a resource type, which is resource creation
- Resource type is a definition in the spec, while resource is in memory

(Normal) Resource vs. virtual resource

- virtual resource has no attributes, no child resources
- virtual resource represents a pre-defined function that is written in the spec.

Resource hierarchy

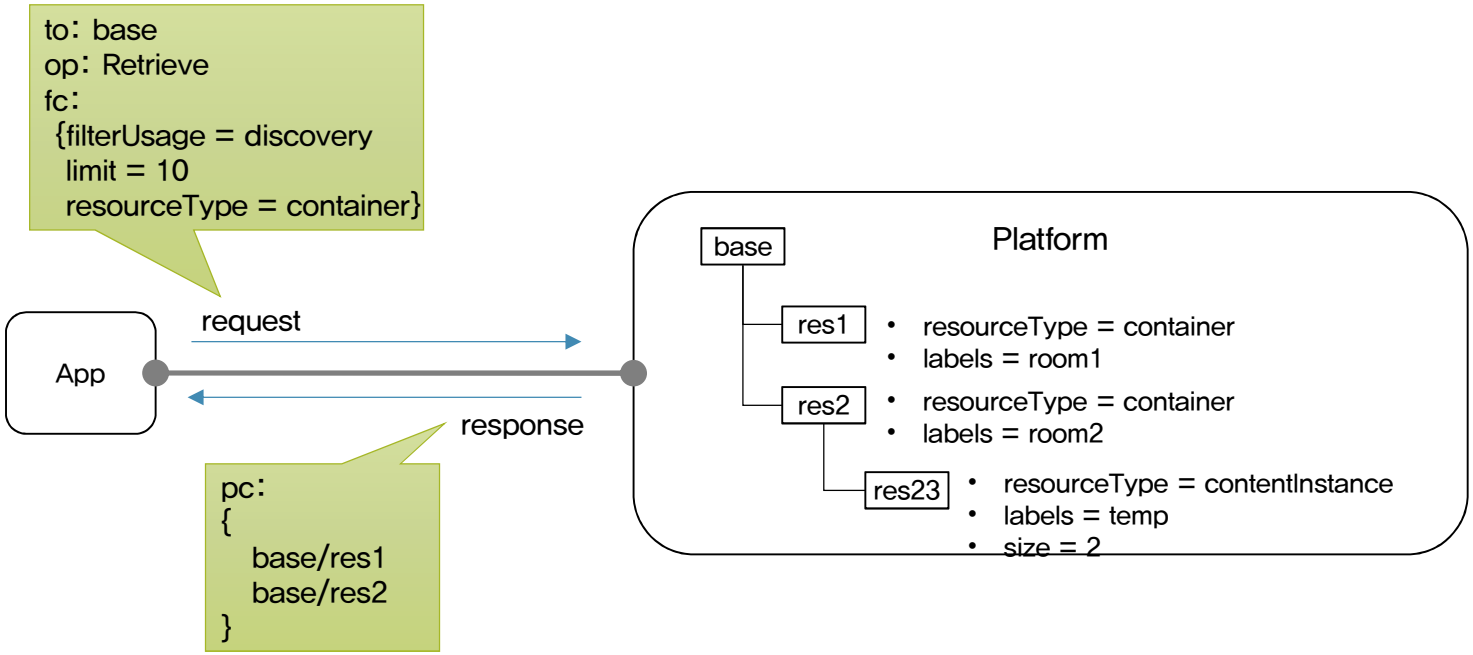
- Resources have parent-child relationship with other resources
- This looks like a resource tree in file systems
- Each platform(CSE) has a resource tree

7.5 RESTful API

▪ oneM2M RESTful API

▪ Platform provides APIs to applications

- Apps consist of
  - oneM2M API call
  - application/service logic



Assignment

- Assignment 7

RESTful 개념을 가진 기술을 조사하고 각각의 특징을 분석해 보시오.

Reference

[7.1] Organization

Aim to Global Standard

그림 출처 : http://www.oneM2M.org

Members

그림 출처 : http://www.oneM2M.org

Structure

그림 출처 : http://www.oneM2M.org

8장. oneM2M 플랫폼 아키텍처와 기능

Chapter 8. oneM2M Platform Architecture and Functions

8.1 Architecture
8.2 Identifiers and Addressing
8.3 Communication Scheme
8.4 Common Services Function
8.5 Parameter Specific Function
8.6 Resource Type Specific Function

- Assignment
- Reference

강의 개요

- **강의 목표**

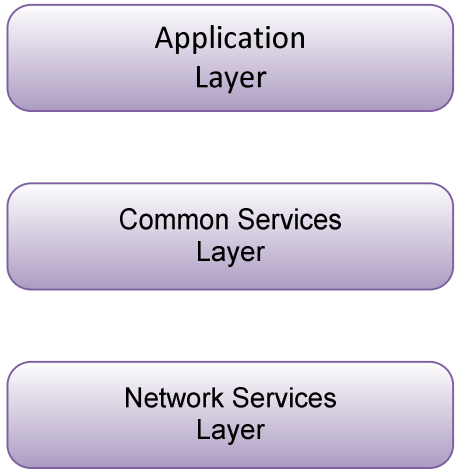
oneM2M 아키텍처의 기본 개념과
oneM2M 플랫폼의 기능을 학습한다.

- **강의 내용**

  - oneM2M 아키텍처
  - oneM2M 식별체계
  - Common Services Function (CSF)
  - Parameter Specific Function
  - Resource Type Specific Function

8.1 Architecture

- oneM2M platform is common services layer
  - The platform provides common functions that needed for IoT applications
  - Application developers do not need to implement network services APIs by themselves



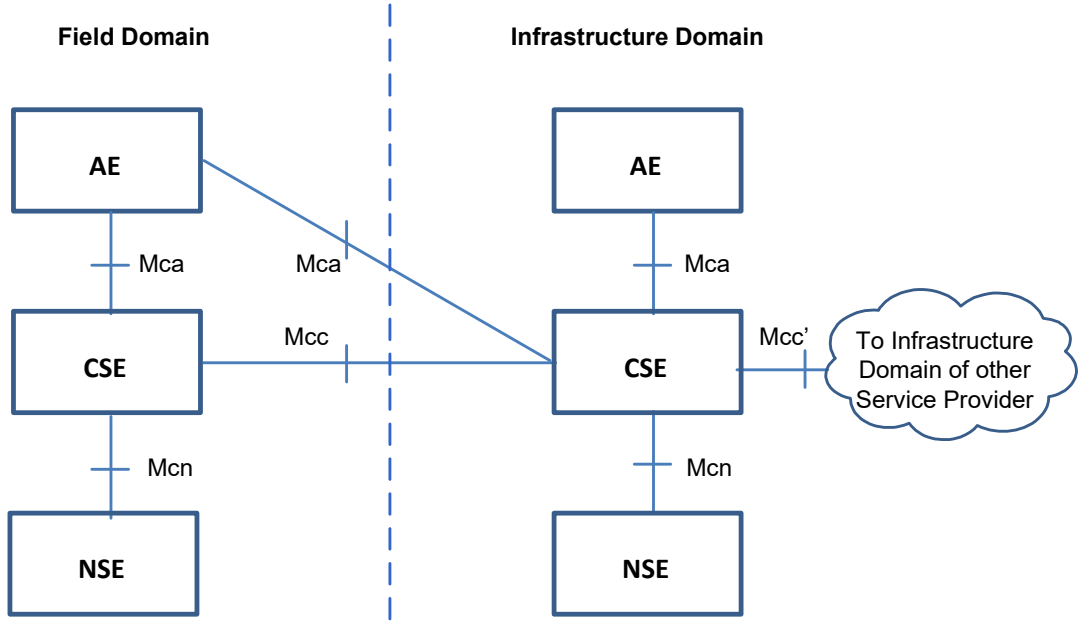
oneM2M Layered Model

8.1 Architecture

▪ Functional Architecture

▪ Two domains in the oneM2M system

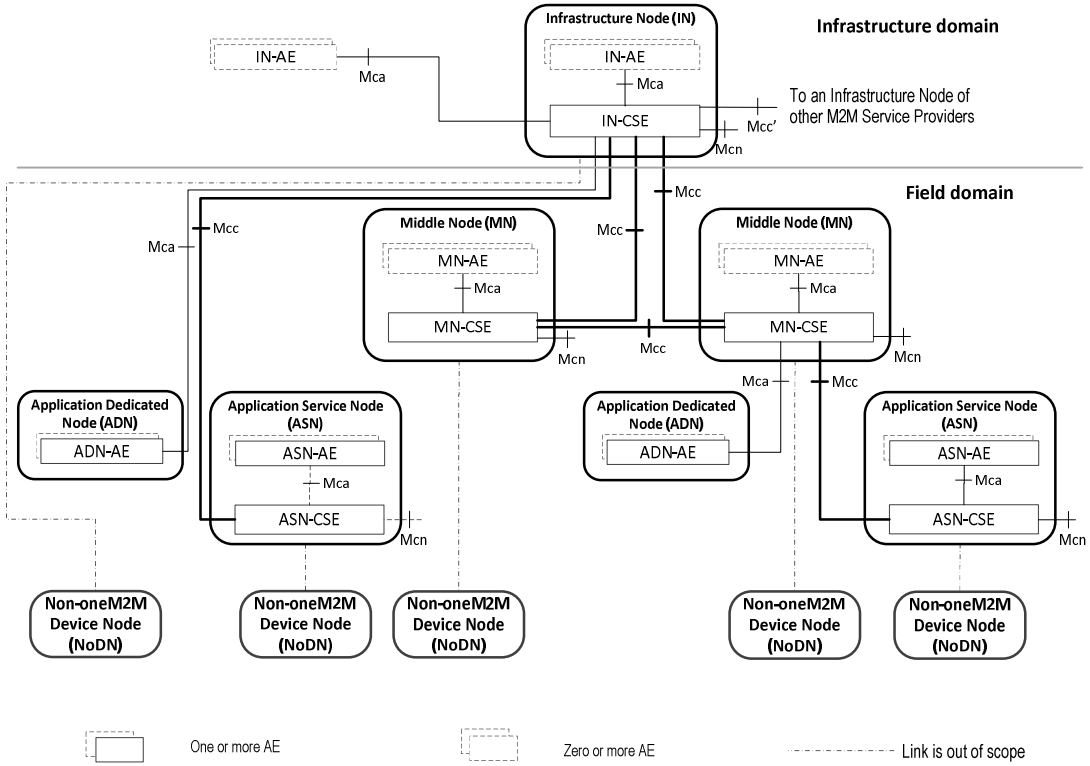
- Infrastructure domain: oneM2M server and applications
- Field domain: oneM2M device, gateways and their applications



oneM2M Functional Architecture

8.1 Architecture

- System Configurations
  - Infrastructure Node
  - Middle Node
  - Application Service Node
  - Application Dedicated Node



Configurations supported by oneM2M Architecture

8.2 Identifiers and Addressing

- oneM2M IDs
  - Entity related
    - M2M Service Provider Identifier
    - Application Entity Identifier
    - Application Identifier
    - CSE Identifier
    - M2M Node Identifier
    - External Identifier
    - Underlying Network Identifier
    - Trigger Recipient Identifier
  - Service related
    - M2M Subscription Identifier
    - M2M Service Profile Identifier
  - Etc.

8.3 Communication Scheme

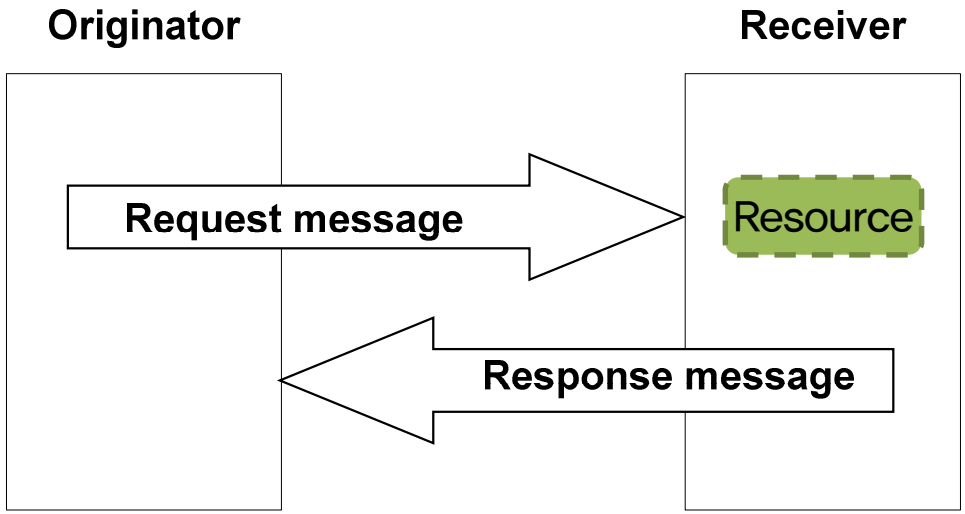
General Flow

Request/Response pair

- Originator sends a request targeting a resource (normally)
- Receiver always sends back a response

Originator requests to

- Create, Retrieve, Update or Delete a resource
- Notify information



General Flow

8.3 Communication Scheme

Entity ID

Originator information of a request

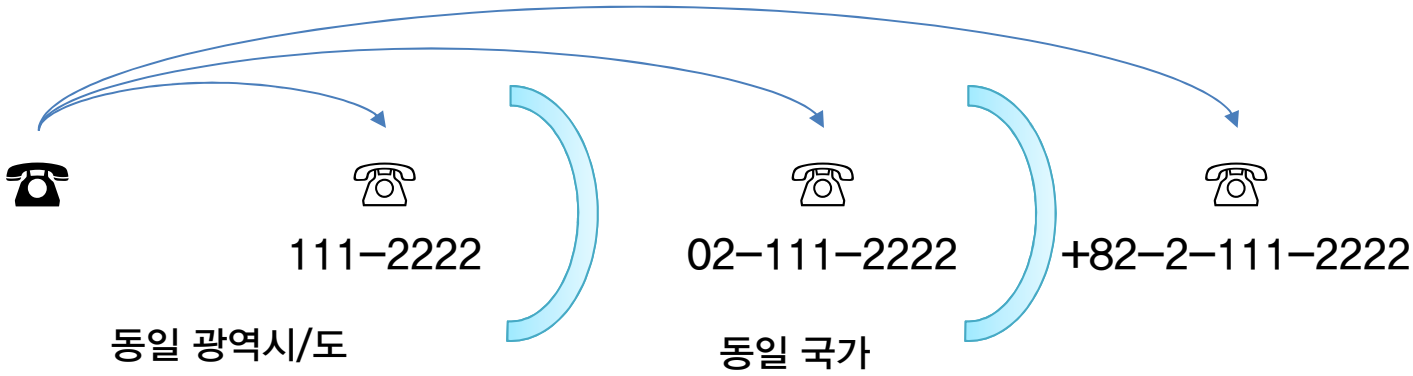
- is included in the From parameter
- is an AE-ID or CSE-ID
  - AE or CSE can be an Originator
- is in Absolute, SP-relative or CSE-relative identifier format
  - the same information but can be in different formats
  - absolute format contains SP-ID (e.g. //telecomABC/CSE12/AE34)
  - SP-relative format contains CSE-ID (e.g. /CSE12/AE34)

8.3 Communication Scheme

Resource ID

Target information of a request

- is included in the To parameter
- is in Absolute, SP-relative or CSE-relative identifier format
- Target is a resource in nature of RESTful API
- A resource is hosted on a CSE, which is a Hosting CSE



8.3 Communication Scheme

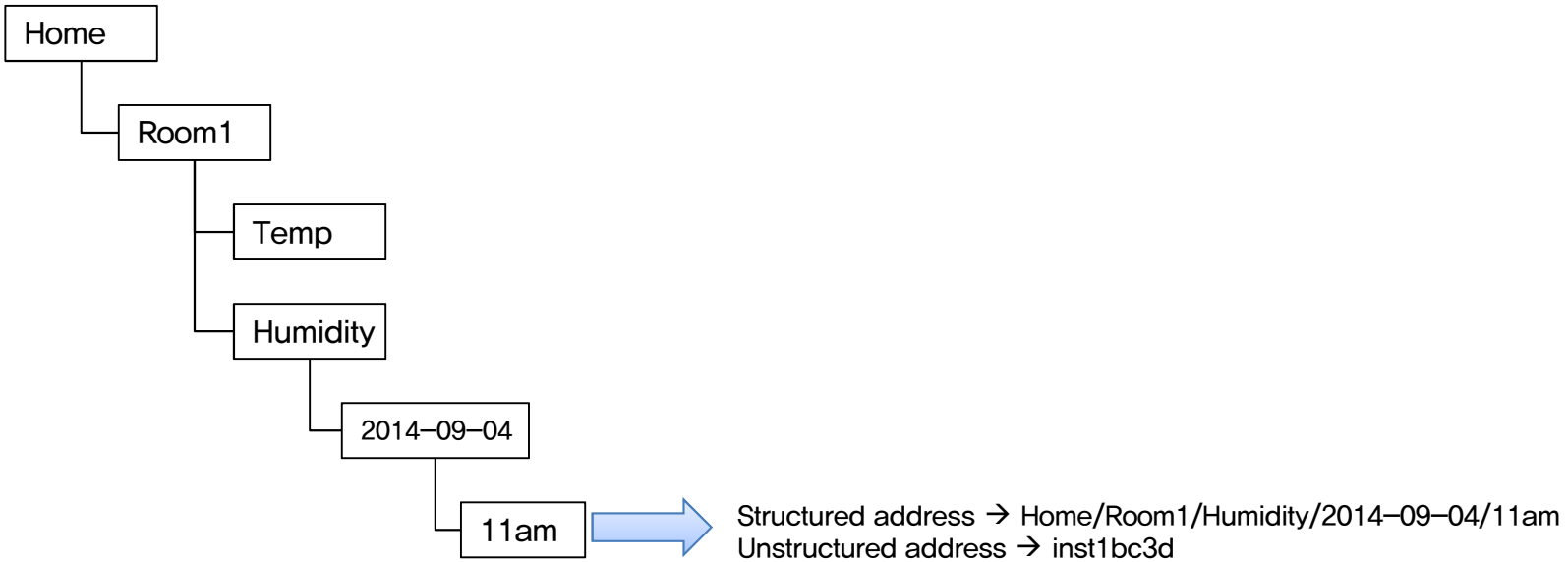
▪ Addressing Schemes

▪ Hierarchical addressing method

- Structured resource address

▪ Non-hierarchical addressing method

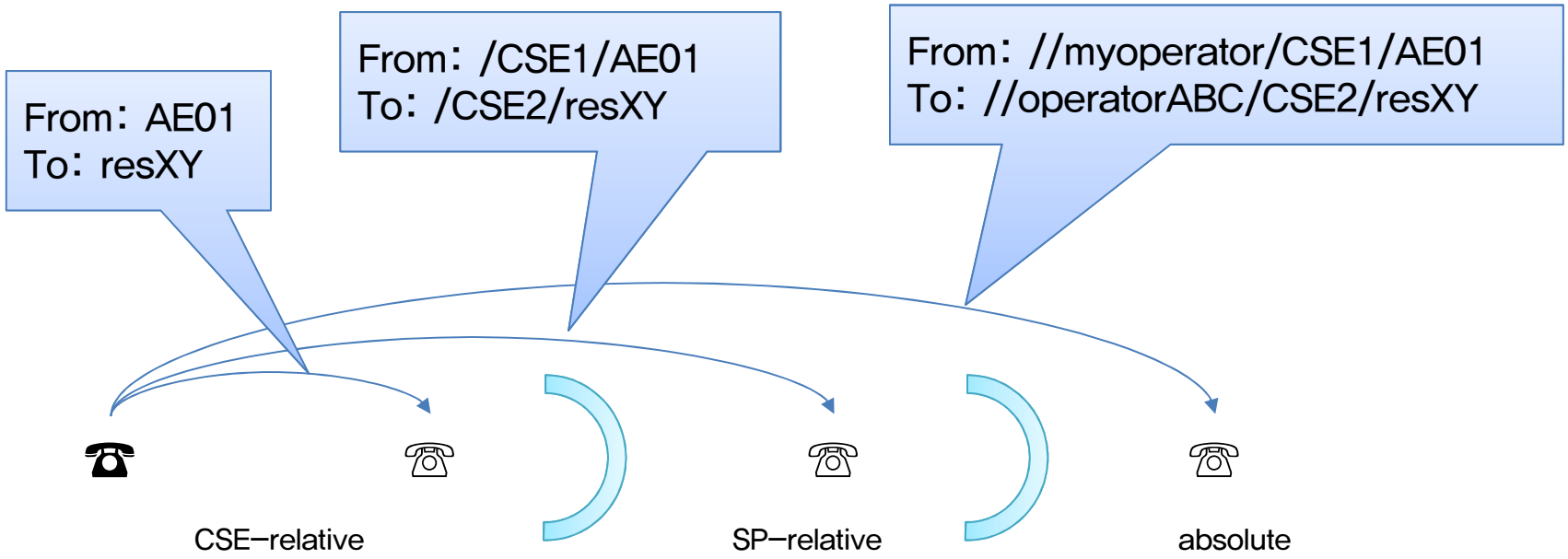
- Unstructured resource address



8.3 Communication Scheme

ID Format Selection

- Use of a format depends on where the Originator and the Receiver locate in
  - They’re Registree and Registrar → CSE-relative format
  - They’re in the same SP domain → SP-relative format
  - They’re not in the same SP domain → absolute format



8.3 Communication Scheme

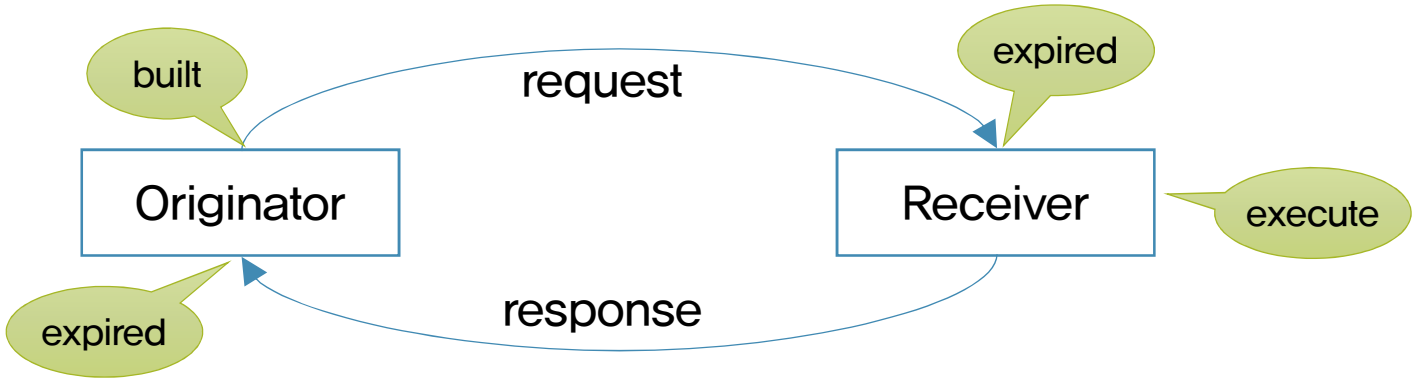
Request Parameters (1/3)

- From
- To
- Operation
  - indicates whether it is Create, Retrieve, Update, Delete or Notify
- Request Identifier
  - unique identifier of a request message
- Content
  - indicates whether it is Create, Retrieve, Update, Delete or Notify
- Resource Type
  - resource type to be created
- Response Type
  - indicates communication type: blocking, non-blocking sync or async

8.3 Communication Scheme

Request Parameters (2/3)

- **Originating Timestamp**
  - indicates the time that the message built
- **Request Expiration Timestamp**
  - indicates when the request gets expired
- **Result Expiration Timestamp**
  - indicates when the response gets expired
- **Operational Execution Time**
  - indicates when the request gets executed by Hosting CSE



8.3 Communication Scheme

Request Parameters (3/3)

Filter Criteria

- is used in Discovery as discovery filter
  - only matching resources are returned
- Retrieve, Update, Delete as conditional
  - only when the filter matches, receiver performs the request
- has filter conditions
  - createdBefore, createdAfter, modifiedSince, unmodifiedSince, stateTagSmaller, stateTagBigger, expireBefore, expireAfter, sizeAbove, sizeBelow, labels, resourceType, contentType, attribute, filterUsage, filterOperation, limit, level, offset, semanticsFilter, contentFilterSyntax, contentFilterQuery

Result Content

- indicates which type of request result will be contained in the Content parameter of the response message

8.3 Communication Scheme

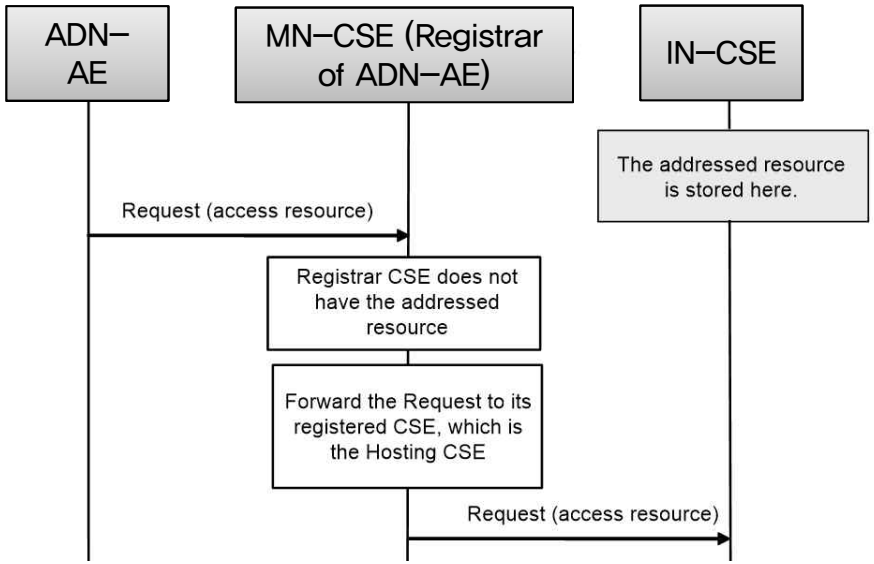
Request Parameter Handling

Handling by Transit CSEs

- request forwarding, expiration handling

Handling by the Hosting CSE

- requested operation handling (CRUDN)



AE/CSE accesses a resource at the Hosting CSE (One Hop)

8.3 Communication Scheme

Response Parameters

Response Status Code

- indicates whether the request handling was successful, unsuccessful, or acknowledged
  - when it is unsuccessful it indicates a specific error code
  - acknowledgement is used for non-blocking sync or async mode

Request Identifier

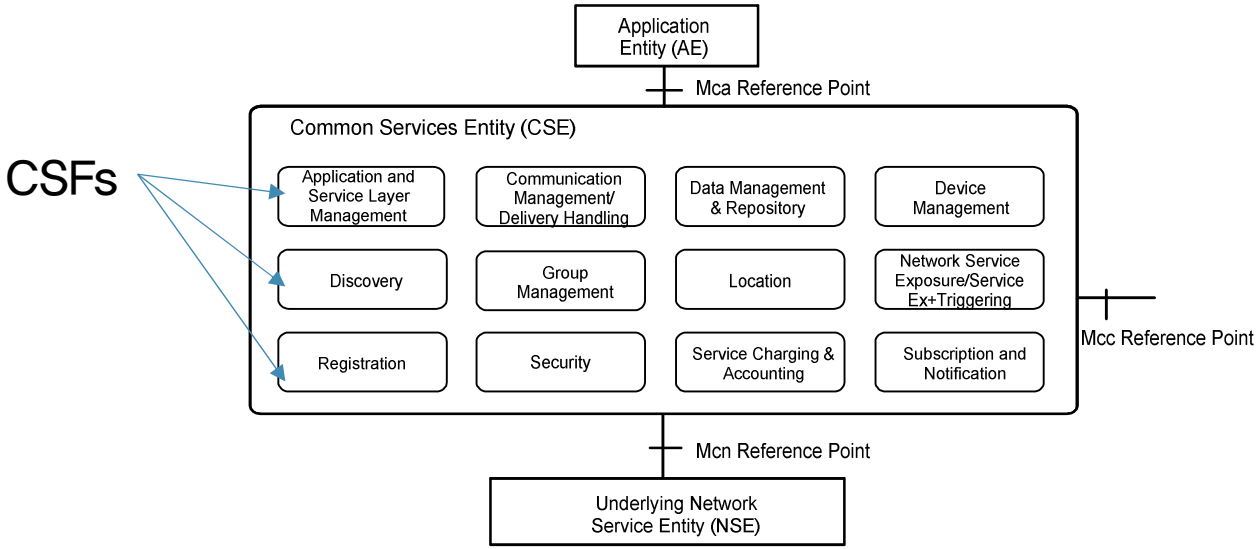
- the same identifier copied from the request message

Content

- response message payload/body

8.4 Common Services Function

- oneM2M platform functions as CSFs
  - Commonly used functions for IoT services by applications
  - application developers don’ t have to implement them by themselves but use the oneM2M common functions(APIs)
  - applications get interoperable by using the same common functions



Common Services Functions

8.4 Common Services Function

Descriptions of CSFs

Registration

- Mandatory/initial procedure to use the platform

Data Management and Repository

- Data storing and sharing

Device Management(DM)

- Device management utilizing existing DM protocols (e.g. OMA DM)

Application and Service Layer Management

- Software management and configuration

Communication Management and Delivery Handling

- Network management and message delivery handling between oneM2M entities

8.4 Common Services Function

- List of CSFs (contd.)
  - Discovery
    - Resource discovery within a CSE
  - Location
    - Location information generation and sharing
  - Group Management
    - Group management in terms of resources(data, services), fan-out and aggregation
  - Subscription/Notification
    - Resource subscription and event notification(as push) configuration
  - Security
    - Provisioning, authentication and authorization
  - Network Service Exposure
    - Network service utilization (e.g. 3GPP network traffic pattern configuration)
  - Service Charging and Accounting
    - Service usage logging and charging

8.5 Parameter Specific Function

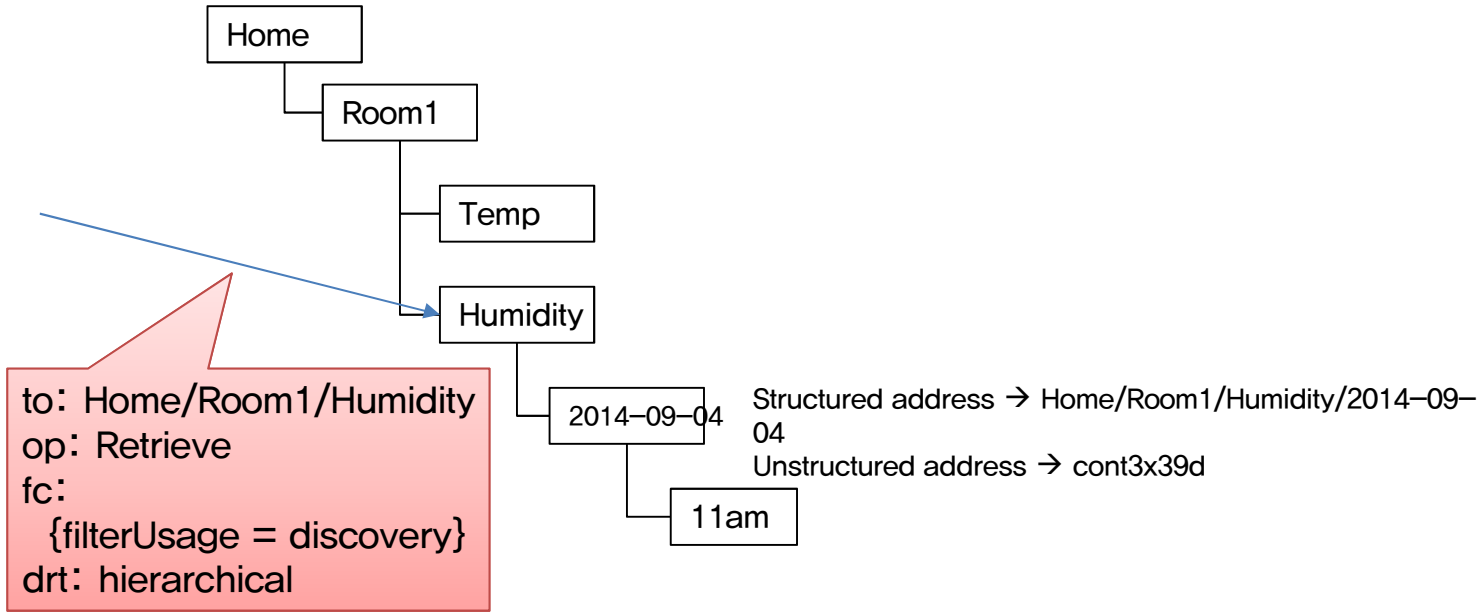
Discovery

Discovery Result Type Parameter

- Indicates whether the result contains structured or unstructured addresses

Scope of Discovery

- Sub(child/descendent)–resources of the target resource
  - Target resource is not included in the result

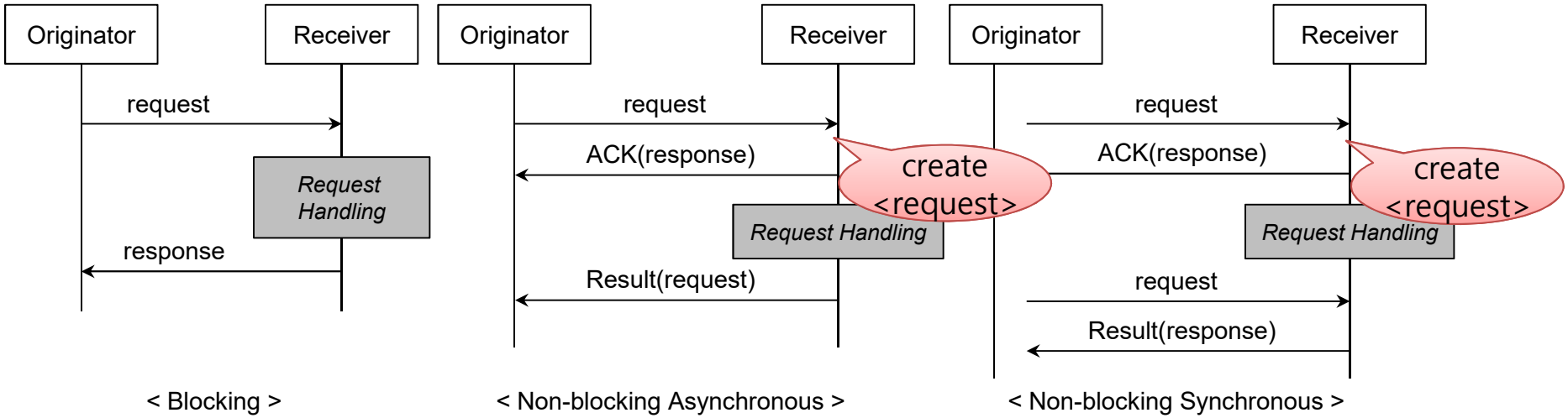


8.5 Parameter Specific Function

▪ Communication Modes

▪ Response Type parameter

- Originator suggests communication modes
  - –Blocking, Non–blocking Synchronous, Non–blocking Asynchronous, Flex–blocking Mode
  - –In case of non–blocking, 〈request〉 resource is created
  - –In case of flex–blocking, the Receiver chooses communication mode



8.5 Parameter Specific Function

- Result Content parameter
  - Nothing → CUDN
  - Attributes → CRUD
  - hierarchical-address
  - hierarchical-address+attributes
  - attributes+child-resources
  - child-resources
  - attributes+child-resource-references
  - child-resource-references
  - original-resource

Create only

Retrieve only

Value	Create	Retrieve	Update	Delete	Notify
attributes	default	default	default	default	n/a
hierarchical-address	valid	n/a	n/a	n/a	n/a
hierarchical-address+attributes	valid	n/a	n/a	n/a	n/a
attributes+child-resources	n/a	valid	n/a	n/a	n/a
child-resources	n/a	valid	n/a	n/a	n/a
attributes+child-resource-references	n/a	valid	n/a	n/a	n/a
child-resource-references	n/a	valid	n/a	n/a	n/a
nothing	valid	n/a	valid	valid	valid
original-resource	n/a	valid	n/a	n/a	n/a

Summary of Result Content Values

8.5 Parameter Specific Function

Content in Response

- In case of a Create request, if the Result Content is “attributes”, then the non-hierarchical address is included in the response
  - To get the hierarchical address one of the following is used
    - hierarchical-address
    - hierarchical-address+attributes
- In case of a Retrieve request, multiple resource representation and/or reference can be returned
  - To indicate the type of information for target resource and/or child(descendent) resources, one of the following is used
    - attributes+child-resources
    - child-resources
    - attributes+child-resource-references
    - child-resource-references
  - “attributes” means attributes(representation) of the target resource
  - “child-resource” means attributes of child/descendent resource
  - “reference” means meta information including resource address

8.5 Parameter Specific Function

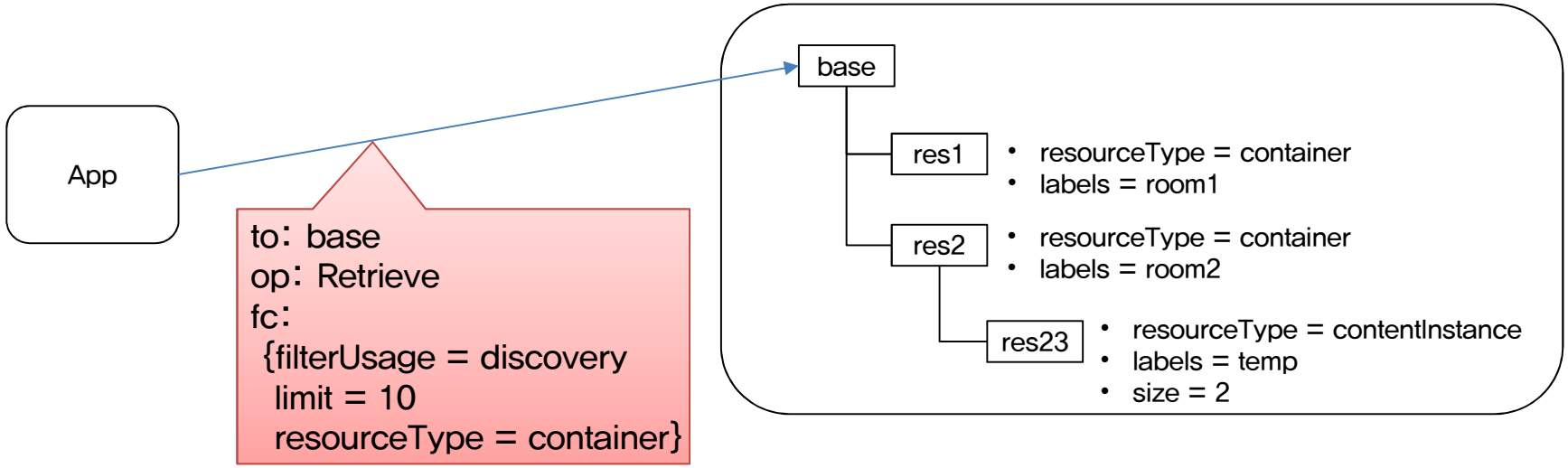
▪ Resource Filtering

▪ Filter Criteria

- is applied to match a specific resource attribute value
  - creationTime, lastModifiedTime, expirationTime, stateTag, label, attribute, resourceType, size, contentType

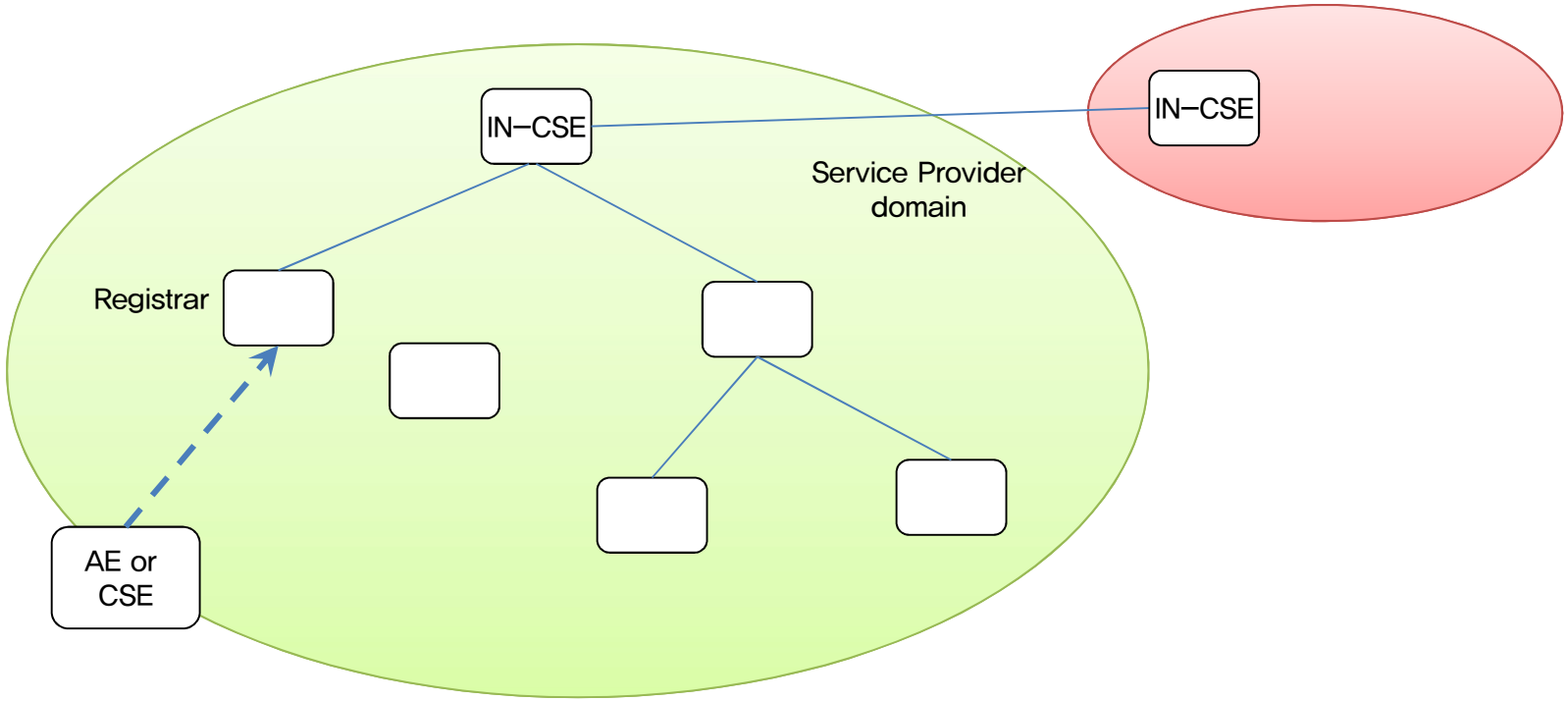
▪ is applied to a filtering procedure, not to an attribute value

- filterUsage, filterOperation, limit, level, offset, semanticsFilter, contentFilterSyntax, contentFilterQuery



8.6 Resource Type Specific Function

- Registration
  - is a prerequisite condition to use the oneM2M system that confirms an entity identifier with authentication check
  - A Registree requests registration to its pre-assigned Registrar (CSE)



8.6 Resource Type Specific Function

Registration

AE registration

- AE should be authenticated using security credentials
- AE-ID is usually assigned by Registrar CSE or IN-CSE
- Relevant APIs
  - 〈AE〉 resource: Create/Retrieve/Update/Delete

CSE registration

- CSE is trusted by the system more than AE
- CSE-ID is always pre-provisioned
- Relevant APIs
  - 〈remoteCSE〉 resource: Create/Retrieve/Update/Delete
  - 〈CSEBase〉 resource: Retrieve

8.6 Resource Type Specific Function

- Data Management : container and contentInstance resource type
  - container contains sub-containers to represent data hierarchy
  - container contains data, which are content instances
    - has meta data of all instances in it
      - max # of instances, current # of instances, max byte size, current byte size, max instance age
  - contentInstance
    - has meta data of each instance as well as data itself
      - media type, encoding, size, creator
  - latest and oldest
    - are pointers to latest and oldest 〈contentInstance〉 resource, respectively
  - Relevant APIs
    - 〈container〉 resource: CRUD
    - 〈contentInstance〉 resource: CRD
    - 〈latest〉, 〈oldest〉 resource: RD

8.6 Resource Type Specific Function

- Data Management : flexContainer resource type
  - is variant of ⟨container⟩, ⟨contentInstance⟩ resource
  - is a customizable container that may have custom attributes
    - so that there are some flexContainer specializations
  - is a container and also a data instance
  - Relevant APIs
    - ⟨flexContainer⟩ resource: CRUD
- timeSeries and timeSeriesInstance resource type
  - are variant of ⟨container⟩, ⟨contentInstance⟩ resource
  - are recording/handling data generation time, not the stored time
    - dataGenerationTime vs. creationTime
  - Relevant APIs
    - ⟨timeSeries⟩ resource: CRUD
    - ⟨timeSeriesInstance⟩ resource: CRD

8.6 Resource Type Specific Function

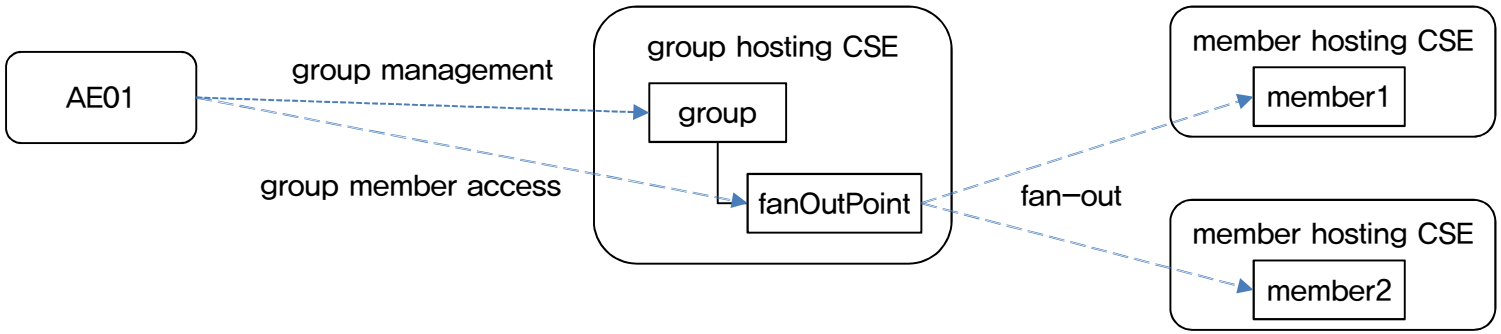
- Access Control : privilege control or authorization
  - Decision making whether a CSE as a receiver allows an Originator's request to access (CRUDN) a resource on the CSE or not
  - Three information is checked as mandatory
    - Who: Originator's Identification (From parameter or others)
    - What: Privilege(rights) to access the target resource (To parameter)
    - How: Operation (CRUDN) to access a resource (Operation parameter)
  - This means a set of access privileges is pre-configured in the system and then used for decision making later when a CSE gets a request

8.6 Resource Type Specific Function

- Access Control : Different mechanisms for access control
  - Access Control Policy
    - Basic mechanism supported in Rel-1 as well as the other releases
    - 〈accessControlPolicy〉 resource and accessControlPolicyIDs attribute
  - Dynamic Authorization Consultation
    - 〈dynamicAuthorizationConsultation〉 resource and dynamicAuthorizationConsultationIDs attribute
  - Role
    - 〈role〉 resource and Role IDs parameter
  - Token
    - 〈token〉 resource and Token Request Indicator / Tokens / Token IDs parameters

8.6 Resource Type Specific Function

- group and fanOutPoint resource type
  - group function consists of
    - group members management
    - request fan-out and response aggregation
  - 〈group〉 resource contains
    - a list of members and access control policy links for accessing members, not the group resource
    - fan-out point to receive a request that will be sent to all members
  - Relevant APIs
    - 〈group〉, 〈fanOutPoint〉 resource: CRUD

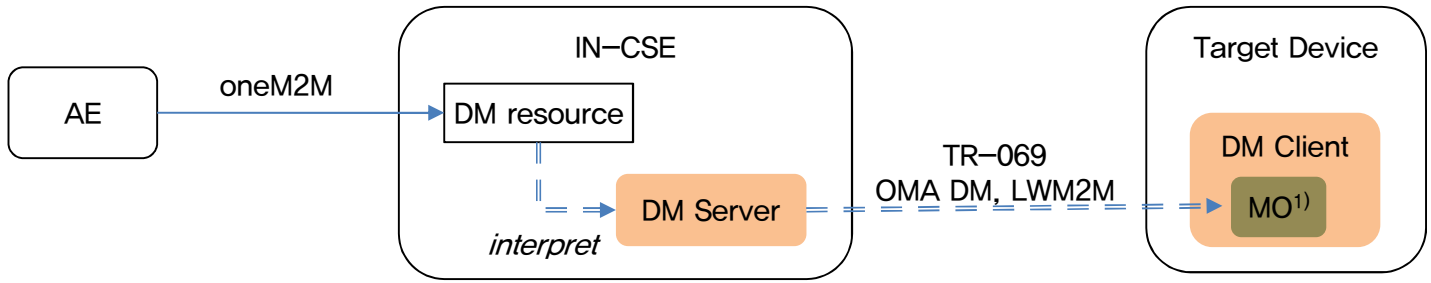


8.6 Resource Type Specific Function

▪ Device Management(DM)

▪ mgmtObj, mgmtCmd and execInstance resource type

- oneM2M DM function utilizes external DM protocols
  - –BBF TR–069, OMA DM and Lightweight M2M
- and interprets oneM2M protocol to those
  - –resource manipulation triggers DM API call
- Relevant APIs
  - –〈mgmtObj〉, 〈mgmtCmd〉 resource: CRUD
  - –〈execInstance〉 resource: RUD

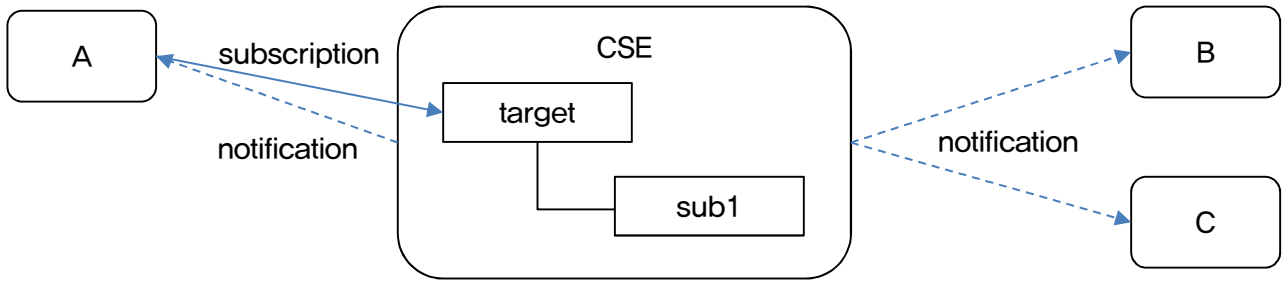


8.6 Resource Type Specific Function

- locationPolicy resource type
  - CSE provides the entity location with the following mechanisms
    - Device-based (e.g. GPS)
    - Network-based (e.g. operator's location server)
    - Sharing-based
  - a location policy stores configuration to get location information
    - Update period, target ID, location server, target area, etc.
  - when a policy is set, data container(〈container〉 resource) is created and they're linked each other
  - then a CSE puts location info. as 〈contentInstance〉 resource and apps get them
  - Relevant APIs
    - 〈locationPolicy〉 resource: CRUD

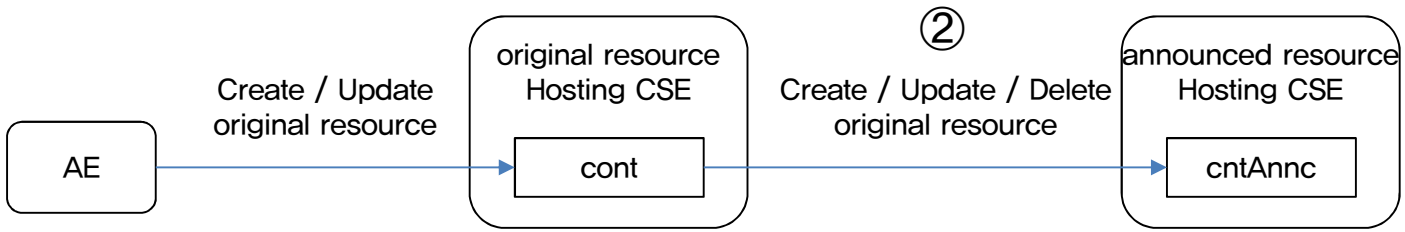
8.6 Resource Type Specific Function

- subscription resource type
  - 〈subscription〉 resource represents configurations for event definition and notification message/transmission
  - Not all resource types are subscribe-able
  - Event monitoring target is subscribed-to resource and its children and events can be filtered
  - Notification target can be multiple and can be different from a resource subscriber
  - Relevant APIs
    - 〈subscription〉 resource: CRUD



8.6 Resource Type Specific Function

- xxAnnc1) resource type
  - Announcement in oneM2M is creating a partial copy of an original resource to a remote CSE
    - Apps can choose attributes to be announced
    - An announced resource has a link to the original one
  - Announcement can ease resource discovery
  - Announced resource cannot be created directly by apps, apps request CSEs to Create/Update/Delete that resource
  - Relevant API
    - ⟨announcableResourceType⟩ resource: CU



1) E.g. containerAnnc

8.6 Resource Type Specific Function

Request Handling

request resource type

- Stores request handling status information in case of non-blocking communication
- No separate APIs, but uses the Response Type parameter

delivery resource type

- Deliver multiple request to the same CSE at once
- No separate APIs, but uses the Delivery Aggregation parameter

pollingChannel and pollingChannelURI resource type

- Request message polling when an entity cannot get push message
- Relevant APIs
  - 〈pollingChannel〉 resource: CRUD
  - 〈pollingChannelURI〉 resource: R

Assignment

- Assignment 08

oneM2M 플랫폼 기능들의 개념과 특징을 분석하고
활용 사례를 조사해 보시오.

Reference

[8.1] Architecture

Functional Architecture

그림 출처: http://www.oneM2M.org/technical/published-documents

System Configurations

그림 출처: http://www.oneM2M.org/technical/published-documents

[8.3] Communication Scheme

Request Parameter Handling

그림 출처: http://www.oneM2M.org/technical/published-documents

[8.4] Common Services Function

oneM2M platform functions as CSFs

그림 출처: http://www.oneM2M.org/technical/published-documents

9장. oneM2M 프로토콜 및 시험인증

Chapter 9. oneM2M Protocol and Test

9.1 Procedures
9.2 Data Types and Serializations
9.3 Protocol Bindings
9.4 Testing Framework
9.5 Interoperability Test
9.6 Conformance Test

- Assignment
- Reference

강의 개요

▪ 강의 목표

oneM2M 프로토콜의 정의와 과정을 학습하고 시험인증 수행절차를 이해한다.

▪ 강의 내용

- one M2M프로토콜 수행 과정
- oneM2M 데이터 구성
- MQTT/CoAP Binding
- oneM2M 시험인증 절차

9.1 Procedures

▪ Overview of Core Protocol

▪ Primitives definition

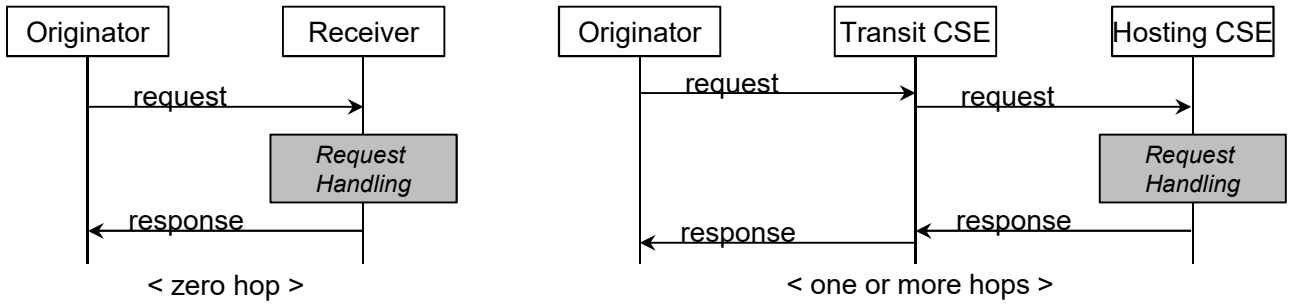
- Request/response primitives (= messages) having the parameters and data types for each parameter and resource attribute

▪ Procedures definition

- Originator procedures
- Receiver(Transit CSE + Hosting CSE) procedures

▪ Serializations definition

- that is generic to different IoT service domains
- so that the IoT market can be expanded

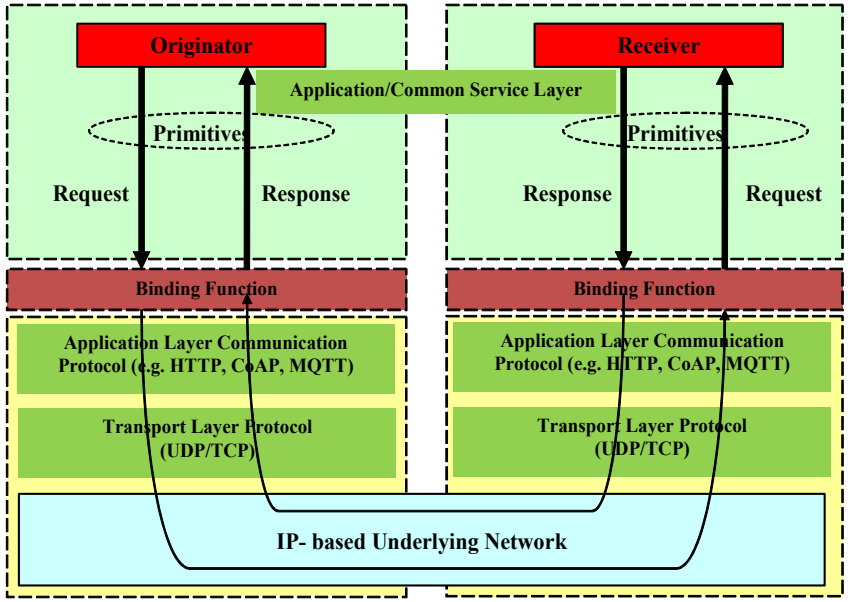


9.1 Procedures

▪ Overview of Protocol Binding

▪ oneM2M primitive transport

- uses messaging protocols HTTP/CoAP/MQTT, and this means
  - primitive parameters are mapped to binding message existing/new headers and body, vice versa
  - there are some binding protocol specific handlings



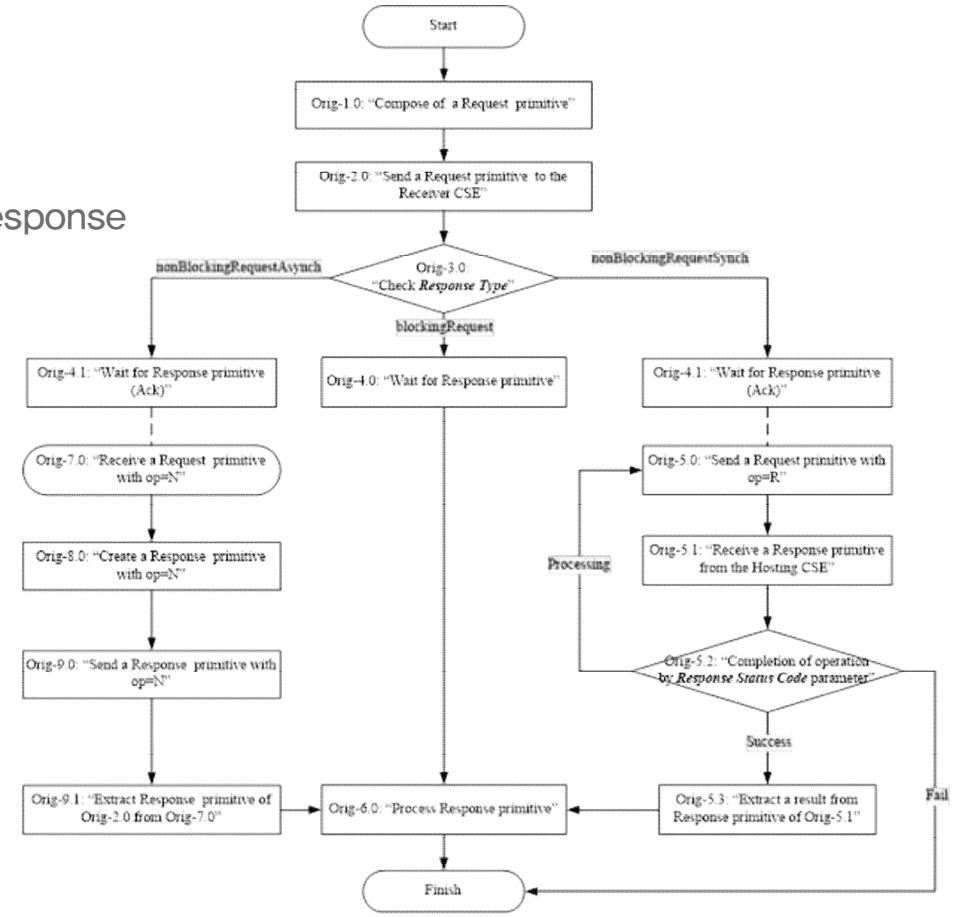
Communication model using Request and Response primitives

9.1 Procedures

▪ Generic Procedures (1/3)

▪ Originator procedure

- In blocking mode
  - it's simply send a request and gets a response
- In non-blocking sync mode,
  - it initiates another Retrieve request
- In non-blocking async mode
  - it handles following notifications



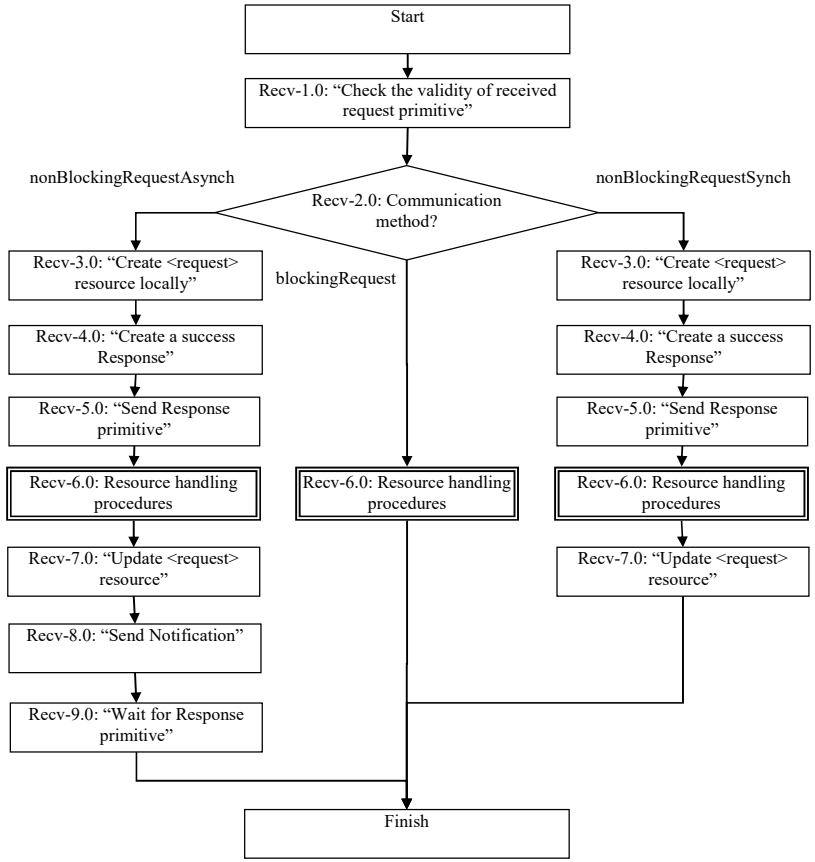
[그림] Generic procedure of Originator

9.1 Procedures

▪ Generic Procedures (2/3)

▪ Receiver procedure

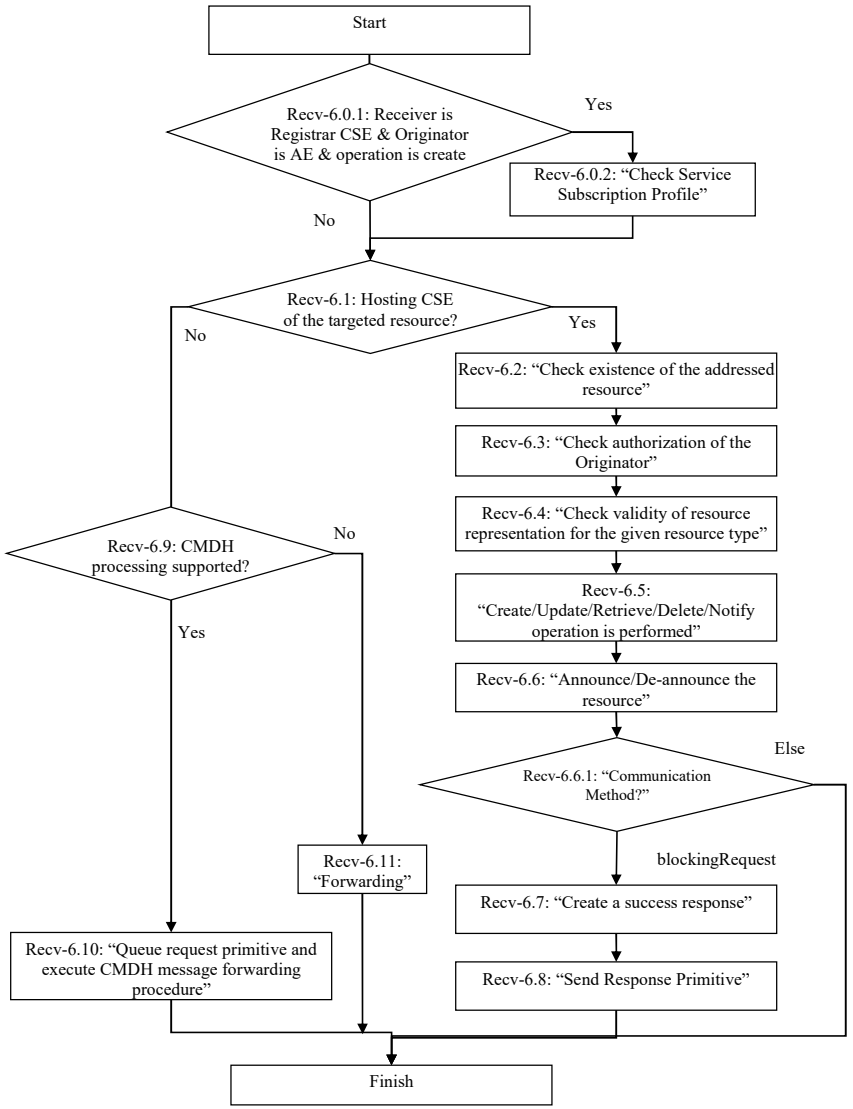
- is applied to Transit CSE as well as Hosting CSE
- both Transit CSE and Hosting CSE perform
  - –blocking and non–blocking communication handling
  - – message validity checking
    - –syntax checking
    - –parameter value checking
- resource handling procedures
  - –is described in further details



[그림] Generic procedure of Receiver

9.1 Procedures

- Generic Procedures (3/3)
  - Resource handling procedure
    - if it is the Hosting CSE
      - check existence of the target
      - authorization
      - perform operation
      - announcement (optional)
      - send a response
    - if it is the Transit CSE
      - forward the request
      - CMDH handling (optional)



[그림] Resource handling procedure

9.1 Procedures

▪ Resource Specific Procedures (1/3)

▪ Each resource type defines

- corresponding XSD file for message validation
- per attribute, it defines
  - request optionality (c.f. multiplicity defined in the TS-0001)
  - data type
  - default value (if any) and constraints
- applicable CRUDN procedures
  - deviating from generic Originator and Receiver procedures

9.1 Procedures

▪ Resource Specific Procedures (2/3)

▪ An example of attribute definition

- request optionality information for CU is given, except RD because
- per attribute, it defines
  - –every attribute is optional in R, and
  - –attribute deletion is done by Update operation with NULL value
    - –this is only applicable for optional attribute
    - –attribute deletion vs. resource deletion
- Data type and constraints for common/universal attributes are defined in the other place (see the Table 6.3.6–1), note that they don’ t have default values

Table 7.4.6.1–3: Resource Specific Attributes of 〈container〉 resource

Attribute Name	Request Optionality		Data Type	Default Value and Constraints
	Create	Update		
maxNrOfInstances	O	O	xs:nonNegativeInteger	No default
maxByteSize	O	O	xs:nonNegativeInteger	No default
maxInstanceAge	O	O	xs:nonNegativeInteger	No default
currentNrOfInstances	NP	NP	xs:nonNegativeInteger	No default (This is generated by the hosting CSE and limited by the maxNrOfInstances)
currentByteSize	NP	NP	xs:nonNegativeInteger	No default (This is generated by the hosting CSE and limited by the maxByteSize)

9.1 Procedures

▪ Resource Specific Procedures (3/3)

▪ Examples of resource specific procedure

7.4.7.2.1 Create 〈contentInstance〉

***Originator**:*

No change from the generic procedures in clause 7.2.2.1.

***Receiver**:*

Primitive specific operation on Recv-6.5 "Create/Update/Retrieve/Delete/Notify operation is performed":

1) The hosting CSE shall set the *contentSize* to the size in bytes of the content attribute.

2) *currentNrOfInstances* and *currentByteSize* of direct parent <container> resorce shall be updated. If currentNrOfInstances and/or currentByteSize exceeds *maxNrOfInstances* and/or maxByteSize of direct parent <container> resource respectively, the hosting CSE shall return the response primitive with a ***Response Status Code*** indicating " NOT_ACCEPTABLE" error.

3) *stateTag* attribute of direct parent <container> resource shall be incremented and copied into the *stateTag* of this <contentInstance> resource.

No other changes from the generic procedures in clause 7.2.2.2. The Originator may omit the name of the <contentInstance> resource unless the Originator need to refer specific content later.

7.4.6.2.1 Create 〈container〉

Originator:

No change from the generic procedures in clause 7.2.2.1.

Receiver:

No change from the generic procedures in clause 7.2.2.2.

9.2 Data Types and Serializations

Data Types

List of oneM2M data types

- Simple types defined in XML schema1)
  - E.g. xs:string, xs:integer, xs:boolean
- Simple types
  - E.g. m2m:ID, m2m:labels, m2m:timestamp
  - Defined in CDT-commonTypes-v2_x_y.xsd[2)]
- Enumeration types represented in integer
  - E.g. m2m:operation, m2m:resourceType
  - defined in CDT-enumerationTypes-v2_x_y.xsd[2)]
- Complex types consist of multiple simple and/or enumeration types
  - E.g. m2m:filterCriteria, m2m:notification
  - Defined in CDT-commonTypes-v2_x_y.xsd[2)]
- Enumeration types defined for home domain information model
  - E.g. hd:deviceType, hd:doorState

1) http://www.w3.org/2001/XMLSchema, 2) http://www.oneM2M.org/technical/xml-schemas

9.2 Data Types and Serializations

Serializations

List of supported message formats

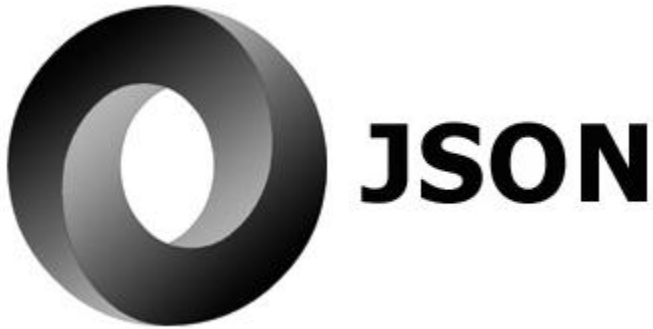
- XML and JSON from Rel-1
- CBOR from Rel-2

Short names

- (Long) Names are used in the documents for human's readability
- However, in the primitives and the binding messages short names are used for efficiency
- Short names are defined for
  - Primitive parameters
    - E.g. Request Identifier → rqi
  - Resource attributes including resource types
    - E.g. container → cnt
  - Complex data type member
    - E.g. createdBefore (of Filter Criteria) → crb

9.2 Data Types and Serializations

- JSON ?
  - JavaScript Object Notation (JSON) is an open standard format that uses human-readable
  - JSON is designed to make data more simple than xml
  - JSON is more useful for mobile because it is faster and easier to parse because it has less functionality than XML.



  - Learn more about JSON
    - http://www.json.org/json-ko.html
    - https://opentutorials.org/course/1375/6844

9.2 Data Types and Serializations

- XML
  - A primitive is serialized in XML
    - <contentInstance> resource creation request example
      - content attribute = “OFF”
      - contentInfo attribute = {text/plain, no encoding}
      - Result Content parameter = “nothing”

```
<?xml version="1.0" encoding="UTF-8"?>
<m2m:rqp xmlns:m2m="http://www.oneM2M.org/xml/protocols">
    <op>1</op>
    <to>/mn-cse/home_gateway/light_ae1/light</to>
    <fr>/mn-cse/Clight_ae1</fr>
    <rqi>mncse/24345</rqi>
    <ty>4</ty>
    <rcn>0</rcn>
    <pc>
        <m2m:cin>
            <con>OFF</con>
            <cnf>text/plain:0</cnf>
        </m2m:cin>
    </pc>
</m2m:rsp>
```

9.2 Data Types and Serializations

▪ JSON

▪ A primitive is serialized in XML

• 〈contentInstance〉 resource creation request example

– content attribute = “OFF”

– contentInfo attribute = {text/plain, no encoding}

– Result Content parameter = “nothing”

```
{
  "op": "1",
  "to": "/mn-cse/home_gateway/light_ae1/light",
  "fr": "/mn-cse/Clight_ae1",
  "rqi": "mncse/24345",
  "ty": 4,
  "rcn": 0,
  "m2m:cin":
  {
    "con": "OFF",
    "cnf": "text/plains:0"
  }
}
```

9.3 Protocol Bindings

▪ Protocol Bindings

▪ List of supported protocol bindings

- HTTP, CoAP and MQTT from Rel-1
- WebSocket from Rel-2
- Possibly DDS1) and OSGi2) from Rel-3
- Possibly Service Layer API from Rel-3
  - APIs in a library for an AE that located in the same device as CSE

	Server/Client	Publish/Subscribe
TCP	HTTP, WebSocket	MQTT
UDP	CoAP	DDS

1) Data Distribution Service, 2) Open Service Gateway initiative

9.3 Protocol Bindings

▪ HTTP Binding (1/2)

▪ Header mappings

oneM2M Parameter	HTTP Header/component
To	path component
From	X-M2M-Origin
Operation	Method
Filter Criteria	query component
Response Status Code	Status-Code
pointOfAddress (attribute)	Host
Resource Type	Content-Type
Request Identifier	X-M2M-RI
notificationURI (element) of *Response Type*	X-M2M-RTU
Response Status Code	X-M2M-RSC
Content	Message-Body

N to 1 mapping

1 to 1 mapping

9.3 Protocol Bindings

▪ HTTP Binding

▪ 〈contentInstance〉 resource creation example

- The Content parameter is carried in the body part
- The other parameters are carried in the header part

```
HTTP Request:

POST /~/mn-cse/home_gateway/light_ae1/light?rcn=0 HTTP/1.1
Host: http://mn.provider.com:8080
X-M2M-Origin: /mn-cse/Clight_ae1
Content-Type: application/vnd.oneM2M-res+xml;ty=4
X-M2M-RI: mncse/24345

〈m2m:cin xmlns:m2m="http://www.oneM2M.org/xml/protocols"〉
    〈con〉OFF〈/con〉
    〈cnf〉text/plain:0〈/cnf〉
〈/m2m:cin〉

HTTP Response:

201 Created
X-M2M-RSC: 2001
X-M2M-RI: mncse/24345
Content-Location: /mn-cse/cin-394798749
Content-Type: application/vnd.oneM2M-res+xml
```

9.3 Protocol Bindings

▪ CoAP Binding

▪ Header mappings

oneM2M Parameter	CoAP Options
To	URI-Path
From	oneM2M-FR
Operation	Method
Filter Criteria	Uri-Query
Response Status Code	Response Code
pointOfAddress (attribute)	Uri-Host, Uri-Port
Resource Type	oneM2M-TY
Request Identifier	oneM2M-RQI
Response Type	Uri-Query
Response Status Code	oneM2M-RSC
Content	*payload*

9.3 Protocol Bindings

▪ MQTT Binding

▪ Header mappings

- MQTT has no header concept
  - The whole request/response primitive representation in XML or JSON is carried in the message body
- However, some parameters are carried in a MQTT topic
  - E.g. "/oneM2M/req/〈originator〉/〈receiver〉"

▪ MQTT topics

- Registration topic using credential
- Request/response topics using Originator ID and Receiver ID
  - To parameter, which is the target resource address, is carried in the body, while the Receiver's entity ID is included in the topics

▪ A message example

- (omitted)

9.4 Testing Framework

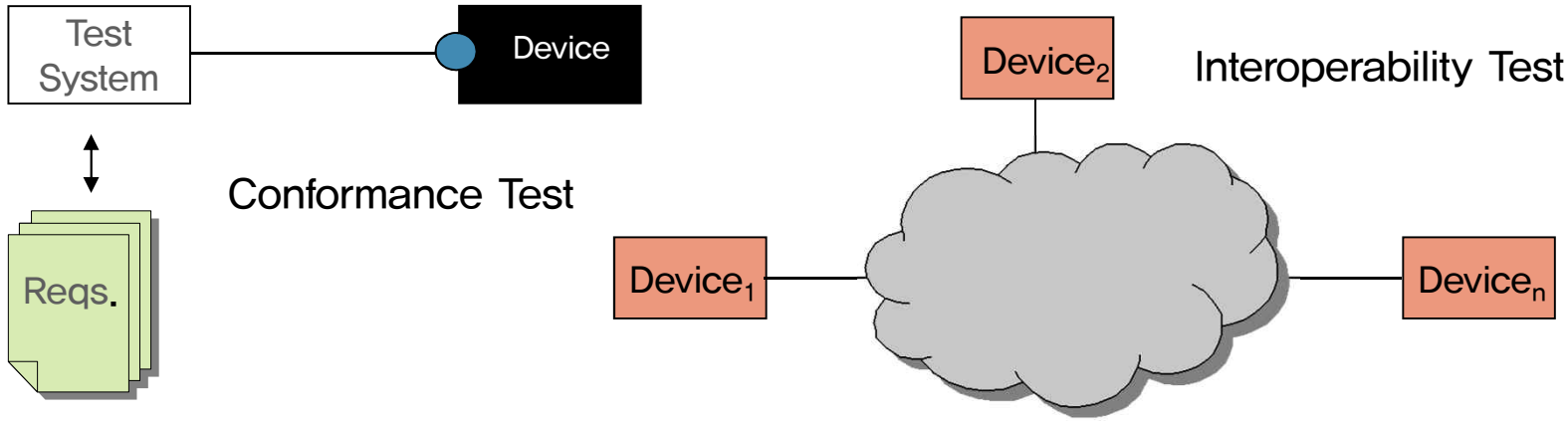
Introduction to Testing

Verifies interoperability of different products

- Products are implementing standards for the interoperability
- To guarantee that, those need to be tested in terms of interoperability and conformance
- Tests provide feedback to enhance standard specifications

Interoperability test and conformance test

- IOP does not guarantee conformance, vice versa
- They're complementary so the both should be tested



9.4 Testing Framework

- IOP vs. Conformance
  - Interoperability test
    - is end-to-end manner
    - tests a complete device or set of devices
    - shows a function is working, but don't know how
    - involves interfaces of different products
    - is manually done
    - is wider scope but less thorough
  - Conformance test
    - is layer-by-layer manner
    - tests a system component
    - can test error behaviors
    - involves test systems
    - can automate and repeat executions
    - is thorough and accurate, but scope is limited

9.5 Interoperability Test

▪ Test Specifications (1/2)

▪ TS-0018, Interoperability Testing

- in version 1
  - Configurations for zero hop and one hop
  - Tested functions
    - Registration
    - Data management (container, contentInstance)
    - Discovery
    - Subscription/notification
    - Device management
    - Request polling
    - Group management and fan-out
    - Access control policy enforcement
    - Non-blocking communication
- in version 2
  - Tested functions to be added (TBD)
    - Service subscription management
    - Interworking with AllJoyn, OCF, LWM2M
    - Etc.

9.5 Interoperability Test

▪ Test Specifications (2/2)

▪ An example of interoperability test description

Interoperability Test Description			
Identifier:			TD_M2M_NH_10
Objective:			AE creates a container resource in registrar CSE via a container Create Request
Configuration:			M2M_CFG_01
References:			TS-0001 [1], clause 10.2.4.1 TS-0004 [2], clause 7.3.5.2.1
Pre-test conditions:			• AE has created an application resource <AE> on registrar CSE
Test Sequence			
Step	**RP**	**Type**	**Description**
1		Stimulus	AE sends a request to create a <container>
2	Mca	PRO Check Primitive	• op = 1 (Create) • to = {CSEBaseName}/URI of <AE> resource • fr = AE-ID • rqi = (token-string) • ty = 3 (Container) • pc = Serialized representation of <container> resource
		PRO Check HTTP	Sent request contains • Request method = POST • Request-Target:{CSEBaseName}/URI of <AE> resource • Host: IP address or the FQDN of Registrar CSE • X-M2M-RI: (token-string) • X-M2M-Origin: AE-ID • Content-Type: application/vnd.onem2m-res+xml; ty=3 or application/vnd.onem2m-res+json; ty=3 • Message-body: Serialized representation of <container> resource
		PRO Check CoAP	Sent request contains • Method: 0.02 (POST) • Uri-Host: IP address or the FQDN of Registrar CSE • Uri-Path: {CSEBaseName}/URI of <AE> resource • Content-type: application/vnd.onem2m-res+xml or application/vnd.onem2m res+json • oneM2M-TY: 3 • oneM2M-FR: AE-ID • oneM2M-RQI: (token-string) • Payload: Serialized representation of <container> resource
		PRO Check MQTT	Sent MQTT PUBLISH message: Topic: "/oneM2M/req/< AE-ID>/<Registrar CSE-ID>" Payload: • op = 1 (Create) • to = {CSEBaseName}/URI of <AE> resource • fr = AE-ID • rqi = (token-string) • ty = 3 (Container) • pc = Serialized representation of <container> resource
3		IOP Check	Check if possible that the <container> resource is created in registrar CSE.
4	Mca	PRO Check Primitive	• rsc = 2001 (CREATED) • rqi = (token-string) same as received in request message • pc = Serialized representation of <container> resource
		PRO Check HTTP	Registrar CSE sends response containing: • Status Code = 201 (Created) • X-M2M-RSC: 2001 • X-M2M-RI: (token-string) same as received in request message • Content-Location: URI of the created resource. • Content-Type: application/vnd.onem2m-res+xml or application/vnd.onem2m-res+json • Message-body: Serialized representation of <container> resource
		PRO Check CoAP	Registrar sends response containing: • Response Code = 2.01 • oneM2M-RSC: 2001 • oneM2M-RQI: (token-string) same as received in request message • Location-Path: URI of the created resource • Content-format: application/vnd.onem2m-res+xml or application/vnd.onem2m-res+json • Payload: Serialized representation of <container> resource

9.5 Interoperability Test

IOP Test Events

oneM2M IOP test events

- 1st test event in France (Sep. 2015)
  - TS-00131) (Rel-1)
  - 26 companies, 64 test descriptions
- 2nd test event in Korea (May 2016)
  - TS-0013 (Rel-1)
  - 20 companies, 70 test descriptions
- 3rd test event in Japan (Nov. 2016)
  - TS-0013 (Rel-2)

Feedbacks to WG2 and WG3

- bug-fix and clarifications had been made
  - E.g. serialization rules, ambiguous behavior descriptions

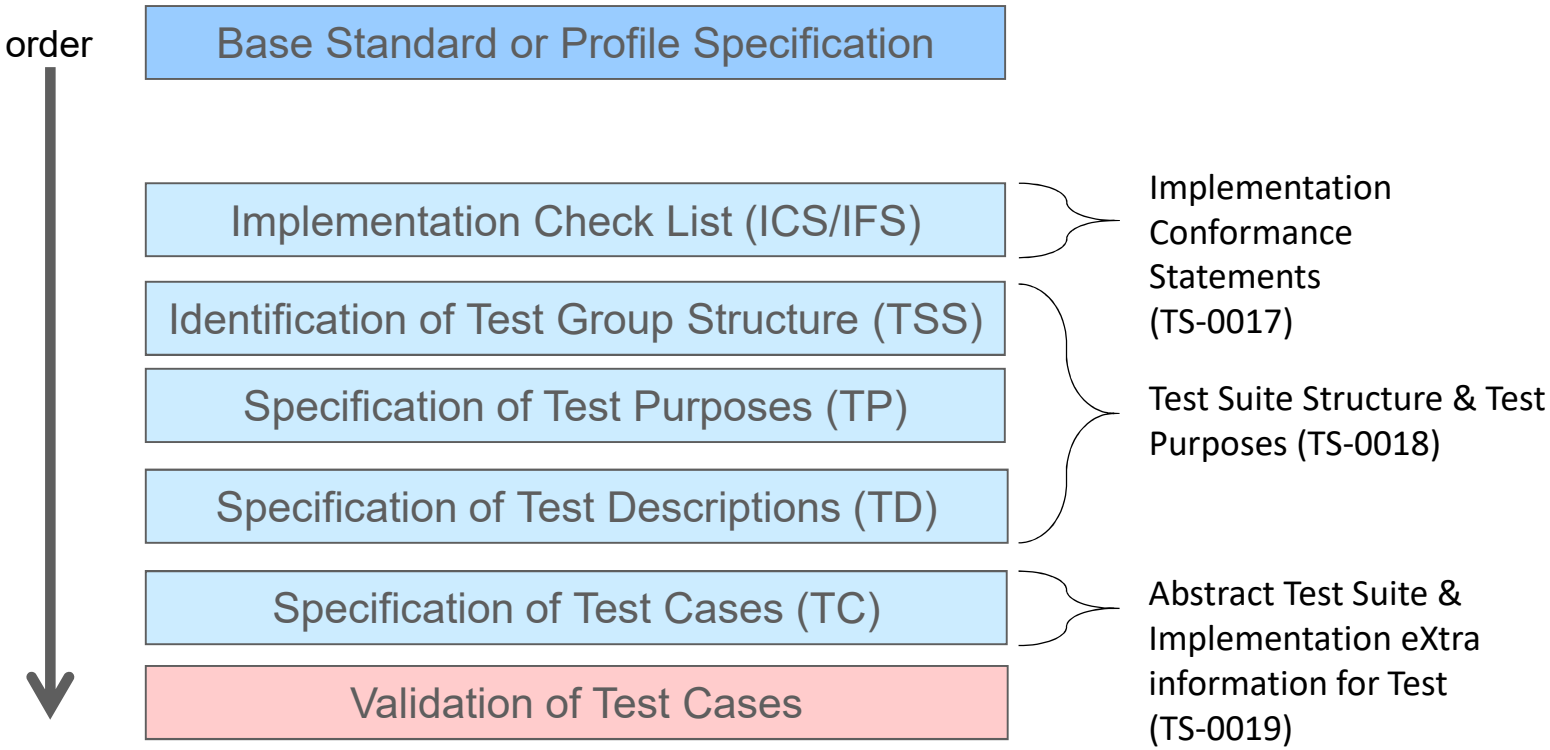
1) Interoperability Testing TS

9.6 Testing Framework

▪ Test Specifications (1/3)

▪ Development procedure

• A series of test specifications are developed in order



9.6 Testing Framework

▪ Test Specifications (2/3)

▪ Implementation Conformance Statement (ICS)

• Checklist of the capabilities supported by the Implementation Under Test (IUT)

– Allows quick & easy first good guess on potential interoperability between two or more implementations

Q#	ICS Question	Reference	Status	Support
Q1	Support of Feature F1	[x] Clause a	OPTIONAL	Yes/No
Q2	Support of Feature F2	[x] Clause b	MANDATORY	Yes/No
:	:	:	:	:
Qn	Support of Message M1	[y] Clause c	CONDITIONAL	Yes/No
:	:	:	:	:
Qk	Support of message Parameter P1	[y] Clause d	OPTIONAL	Yes/No

9.6 Testing Framework

▪ Test Specifications (3/3)

▪ An example of Test Purposes

• Detailed descriptions of test purposes for a particular requirement

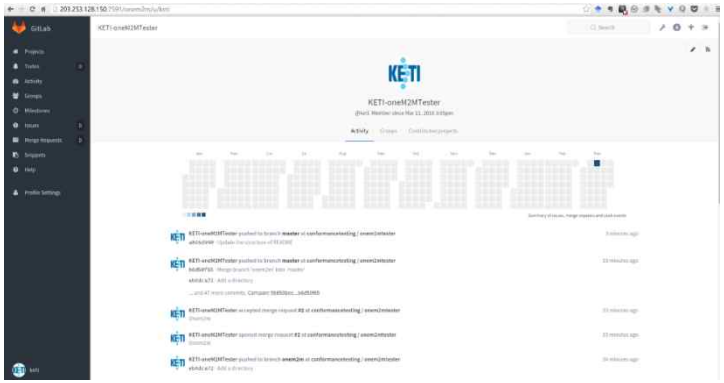
TP Id	TP/oneM2M/CSE/DMR/CRE/BV/001	
Test objective	Check that the IUT accepts the creation of a *RESOURCE_TYPE* resource with resource name not provided by AE	
Reference	TS-0001 10.1.1.1	
Config Id	CF01	
PICS Selection	PICS_CSE	
Initial conditions	**with {** the IUT **being** in the "initial state" **and** the IUT **having registered** the AE **and** the AE **having** privileges to perform CREATE operation on the resource TARGET_RESOURCE_ADDRESS }	
Expected behaviour	**Test events**	**Direction**
	when { the IUT **receives** a valid CREATE request **from** AE **containing** To **set to** AE_RESOURCE_ADDRESS **and** Resource-Type **set to** *RESOURCE_TYPE* **and** From **set to** AE-ID **and** **no** Name attribute **and** Content **containing** *RESOURCE_TYPE* resource **containing** **no** ResourceName attribute **}**	IUT ← AE
	then { the IUT **sends** a Response message **containing** Name **and** Response Status Code **set to** 2001 (CREATED) **and** Content **containing** *RESOURCE_TYPE* resource **containing** ResourceName attribute **}**	IUT → AE

9.6 Testing Framework

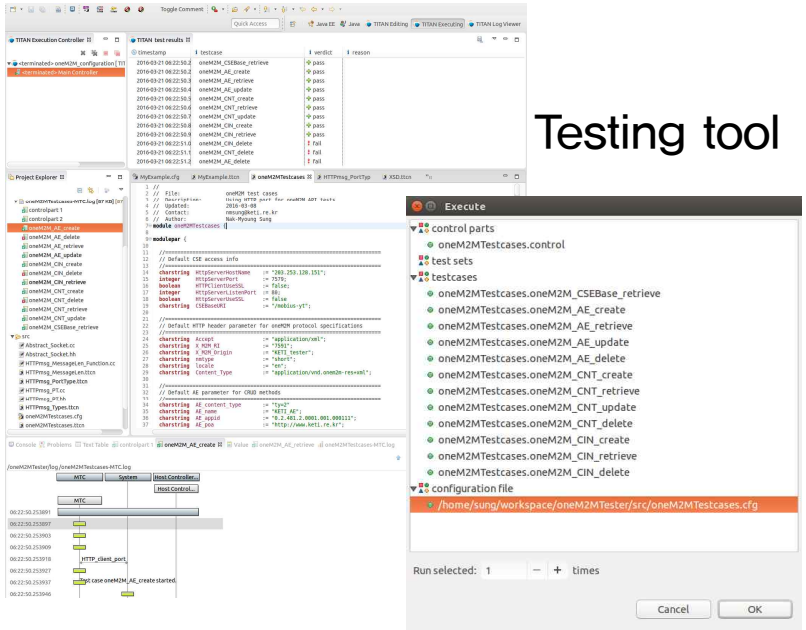
▪ Testing Tool

▪ oneM2M Tester is the name of open source project as well as the conformance test tool

- Project members are KETI, InterDigital, Ericsson, ETSI, TTA, etc.
- The tool was demonstrated during oneM2M 2nd IOP Test (`15.05)



Open source project webpage



Testing tool

9.6 Testing Framework

Certification Program

Phase-1 certification

- will be launched in 2016 per partner type 1 (regional SDO)
  - TTA runs the certification program extending TTA's oneM2M verified program in early 2016
- is compatible with oneM2M Rel-1 specifications
  - Test specifications for Rel-2 would be available in 2017

Phase-2 certification

- will be launched in 2017 globally
- The tool was demonstrated during oneM2M 2nd IOP Test ('15.05)

9.6 Testing Framework

- App Developer Guide
  - provides a set of call flows for use case scenario
  - provides XML and JSON messages for reference
  - provides example usages of oneM2M RESTful APIs
    - AE, container, contentInstance, latest, group, etc.

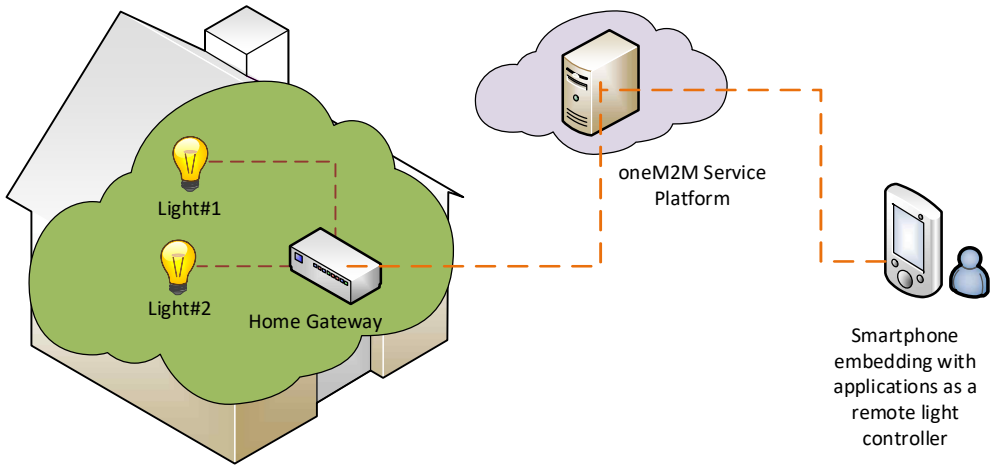


Figure 5.1: Overview of remote lights control use case [TR-0025]

Assignment

- Assignment 9

IoT 서비스에서 발생 할 수 있는 데이터를 가정하여 oneM2M표준에 따라 XML과 JSON 형식으로 작성해 보시오

Reference

[9.1] Procedures

Overview of Protocol Binding

그림 출처: http://www.oneM2M.org/technical/published-documents

Generic Procedures

그림 출처: http://www.oneM2M.org/technical/published-documents

[9.2] Interoperability Test

Test Specifications

그림 출처: http://www.oneM2M.org/technical/published-documents

[9.3] Testing Framework

App Developer Guide

그림 출처: http://www.oneM2M.org/technical/published-documents

10장. OCEAN Open Sources

Chapter 10. OCEAN Open Sources

10.1 OCEAN Open Sources
10.2 Mobius – &Cube
10.3 Utilization
10.4 Reference Hardware
10.5 Runtime Environment

- Assignment
- Reference

강의 개요

- **강의 목표**

  &Cube의 기능 및 버전을 학습하고 Raspberry pi 상에서
  동작시키기 위한 환경 구축을 목표로 한다.

- **강의 내용**

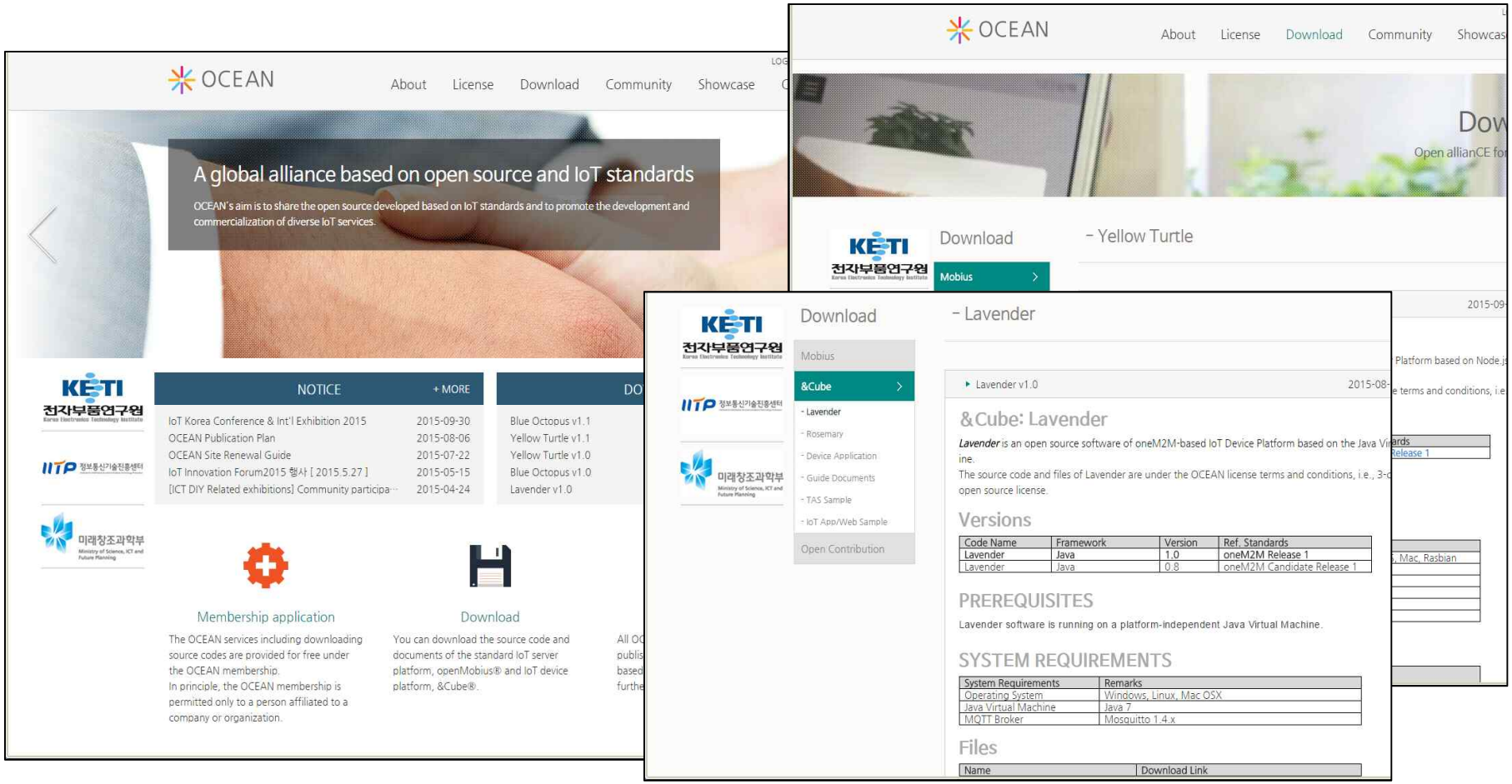
  - Ocean Open Source
  - Mobius – &Cube 의 버전 및 기능
  - & Cube 동작을 위한 raspberry pi 환경 구축
  - 센서 디바이스 연동

10.1 OCEAN Open Sources

- What is OCEAN
  - OCEAN (Open allianCE for iot stANdard) established in Jan. 6th, 2015 by KETI and Korea Government.
  - The objective of OCEAN is to share open sources based on IoT standards and to encourage co-working between its members
  - The OCEAN supports early commercialize and vitalized ecosystem for IoT
    - 데이터의 양은 크기를 말하는 것이지만 단순히 물리적인 크기가 아닌 데이터의 '속성' 이 더 중요하고 그것을 처리하는데 있어 어려움의 유무를 의미하는 것임
- License policy
  - 3-Clause BSD license policy
  - OCEAN adapts IPR policy of the standards referred by open
- Current Members
  - 464 members (Oct. 7, 2016)

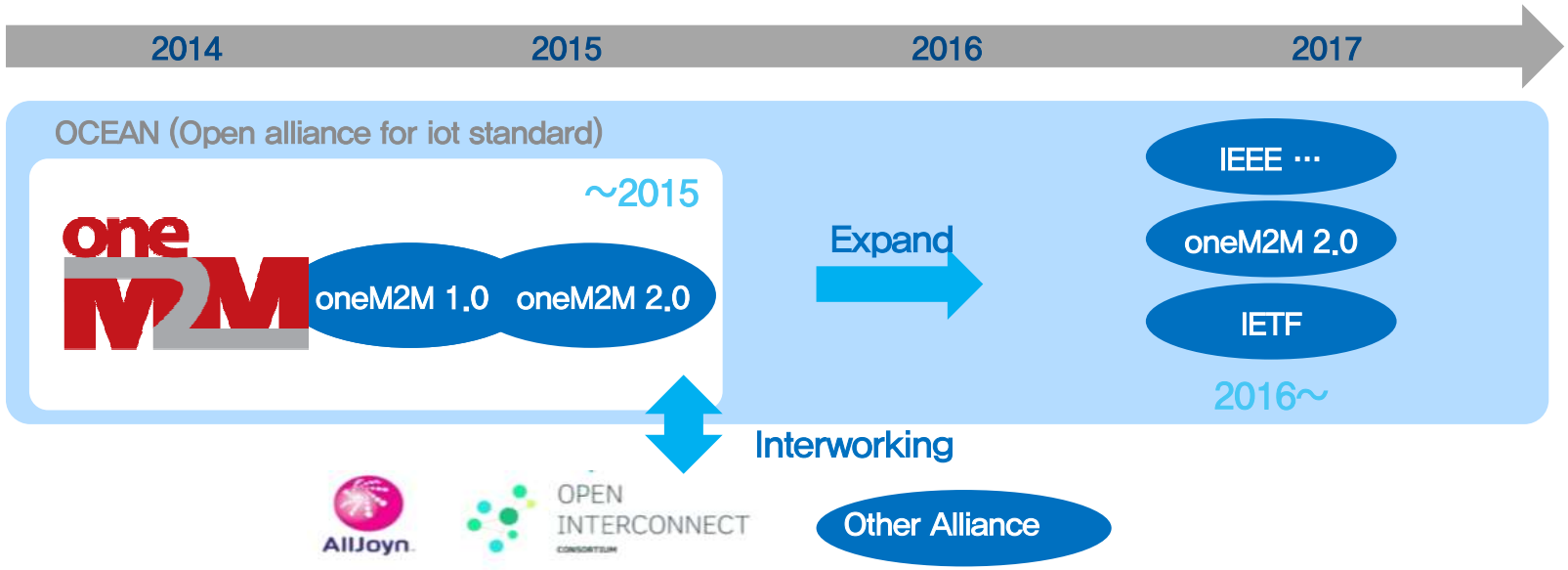
10.1 OCEAN Open Sources

- http://www.iotocean.org
- Have more platforms → become complicated



10.1 OCEAN Open Sources

- Open source
  - The OCEAN is now providing open sources of oneM2M-based IoT platform called "Mobius " and "&Cube ", and relevant documents.
  - For download of the open source, users should join to OCEAN web site (http://www.iotocean.org).



10.1 OCEAN Open Sources

▪ Platform Open Sources

Opened | Not yet

SW Name \ oneM2M Nodes		AE	CSE			Framework
			ASN	MN	IN	
Mobius	Blue Octopus				√	Spring
	Yellow Turtle				√	Node.js
&Cube	Rosemary			√		Java
				√		Node.js
	Lavender		√			Java
			√			Node.js
	Thyme	√				Java
		√				Node.js

10.1 OCEAN Open Sources

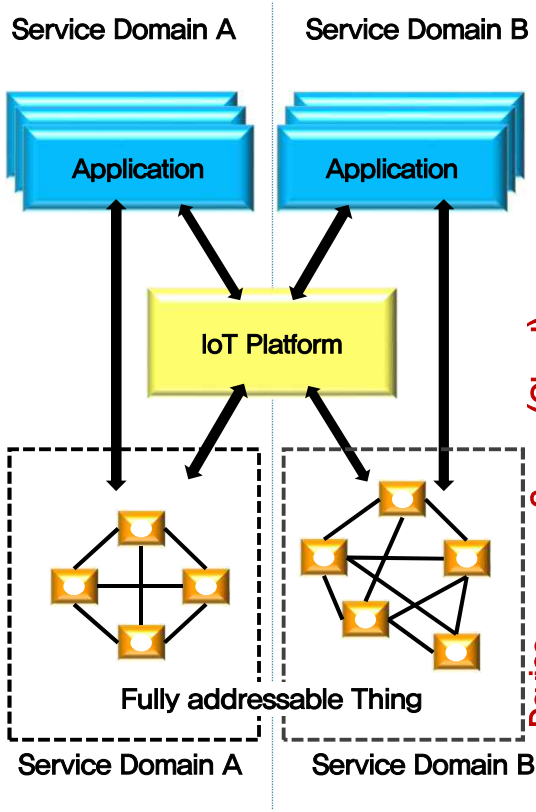
- Interworking and Tool Open Sources

	Opened	Not yet

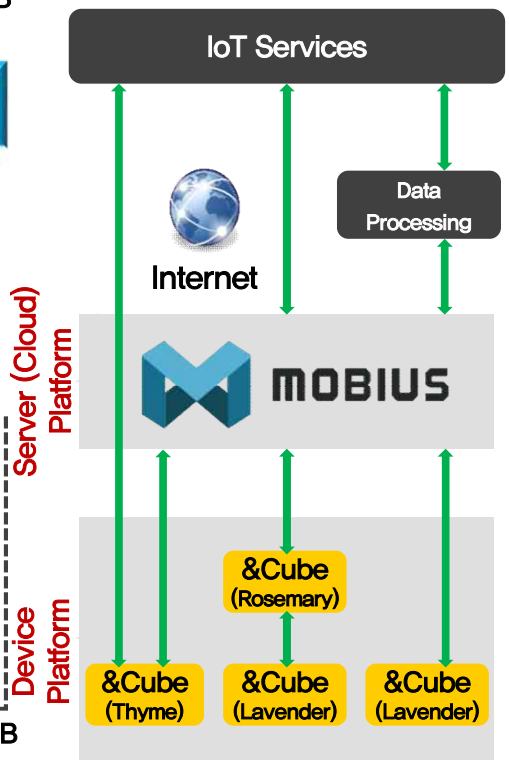
SW Name \ Interworking and Tools		Interworking SW	Testing Tool	Developer Supporting	Framework
Interworking	OCF IPE	√			IoTivity
	AllJoyn IPE	√			Java
	LWM2M IPE	√			Java
Tool	Resource Tree Viewer			√	C#
	oneM2MTester		√		Eclipse TITAN

10.2 Mobius – &Cube

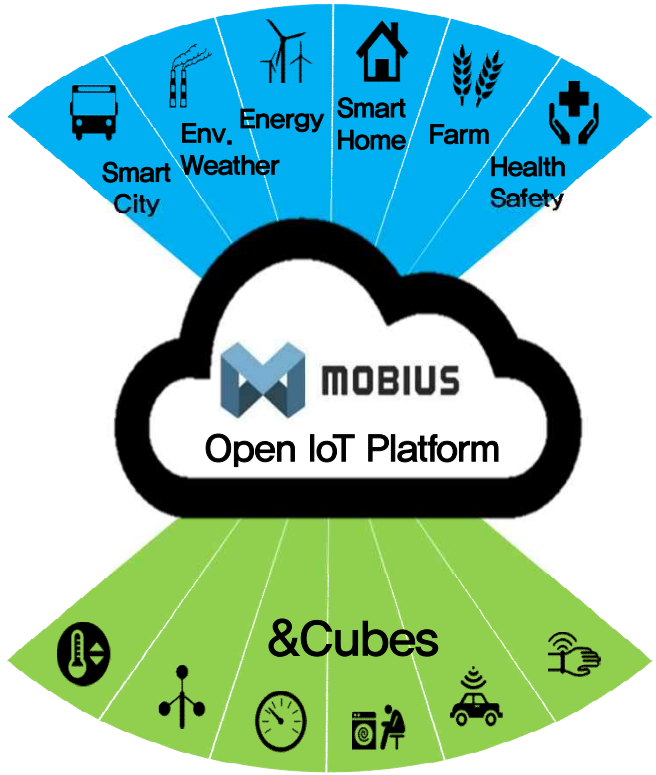
- Horizontal IoT Platform



Horizontal IoT Platform Model



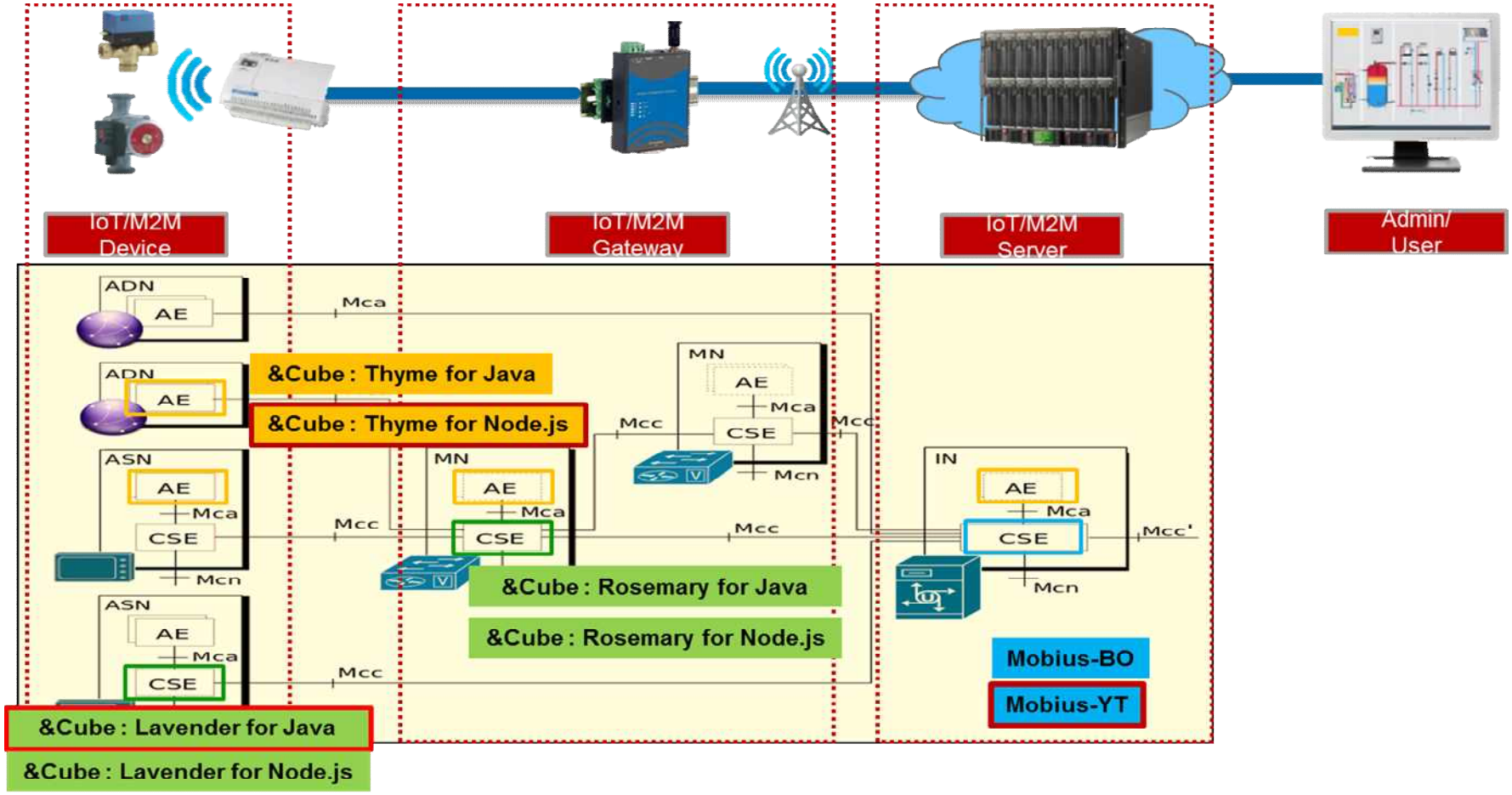
KETI's Open IoT Platforms (Mobius and &Cube)



IoT Application Domains

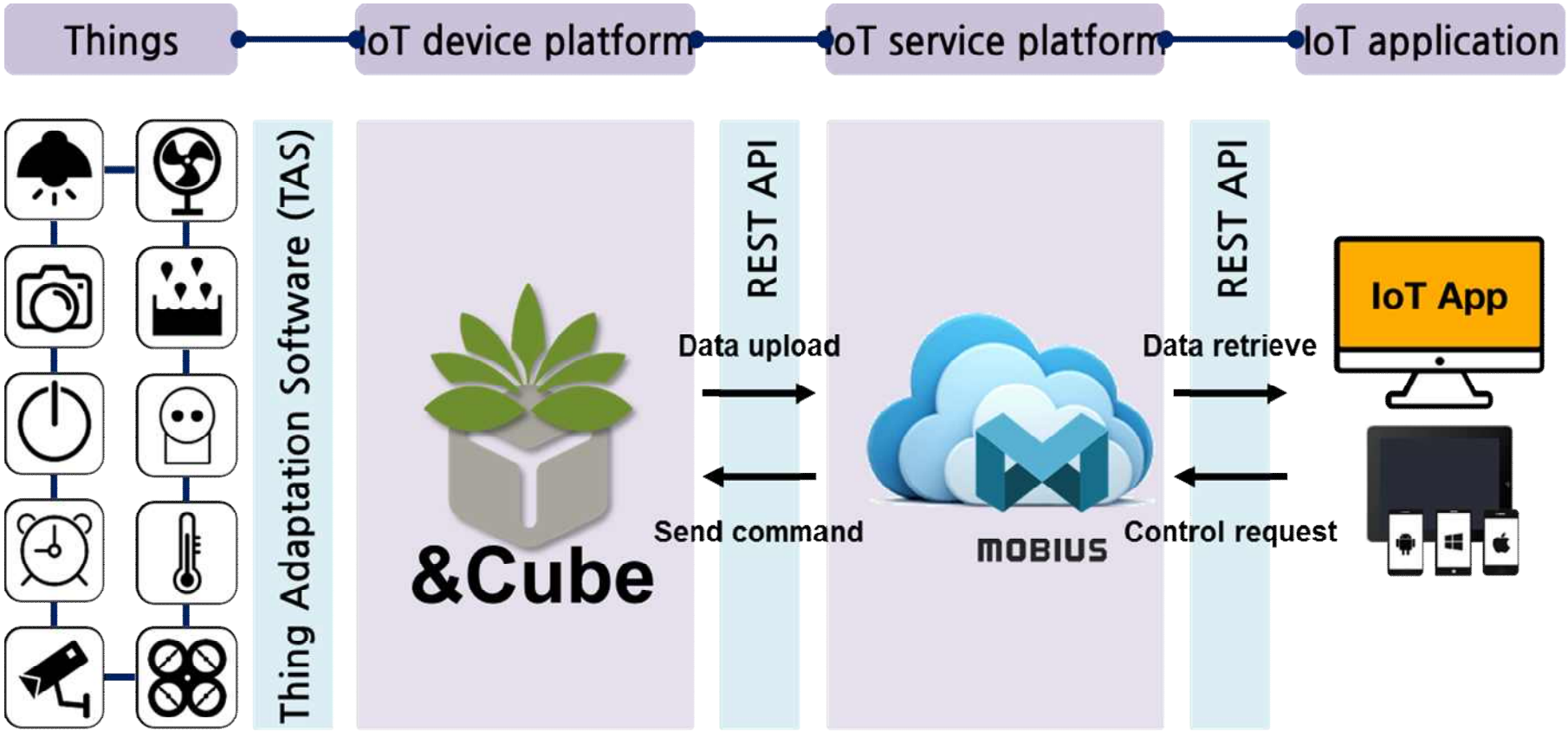
10.2 Mobius – &Cube

- Interconnection Structure



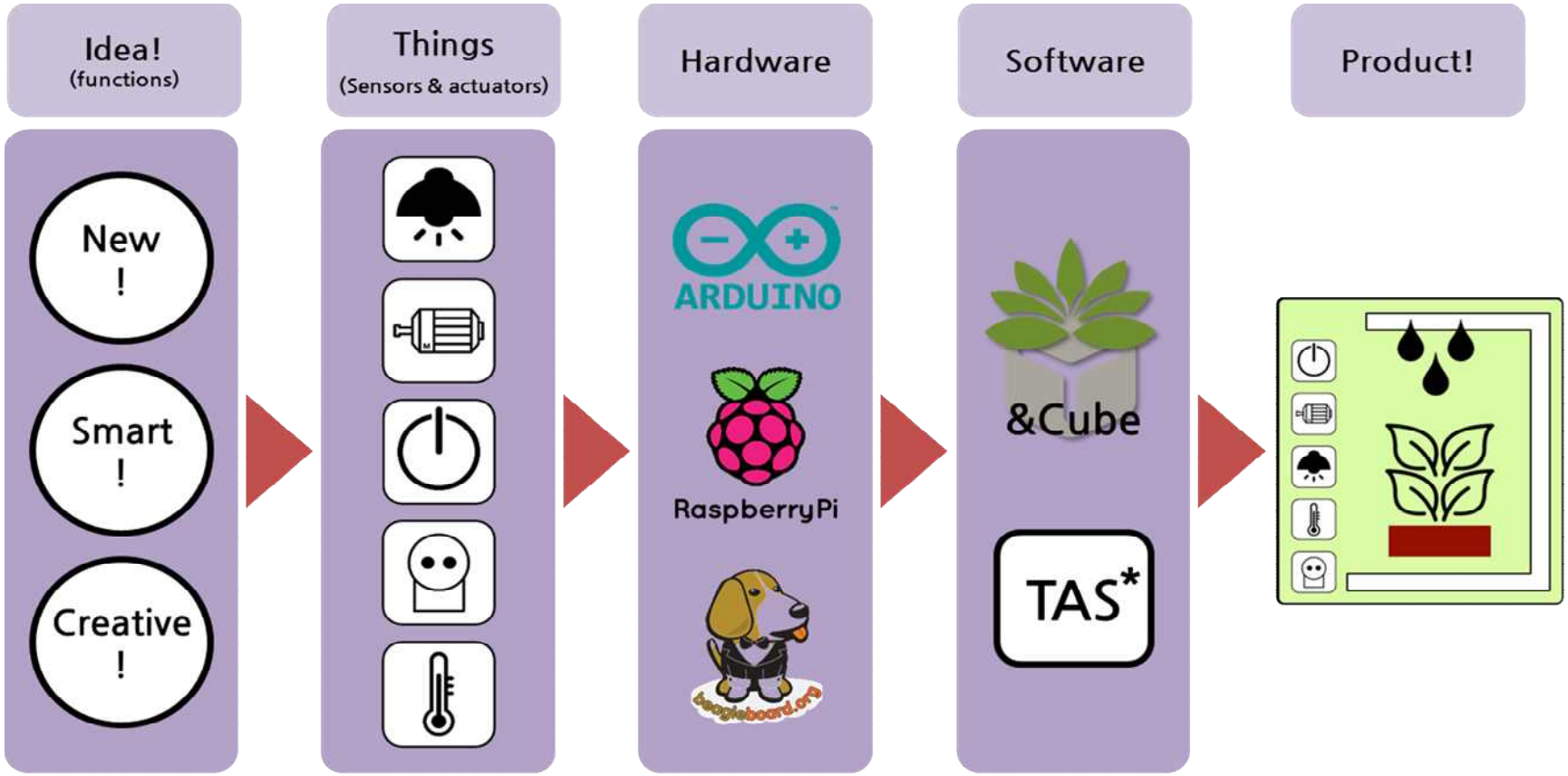
10.3 Utilization

- How IoT Devices Work



10.3 Utilization

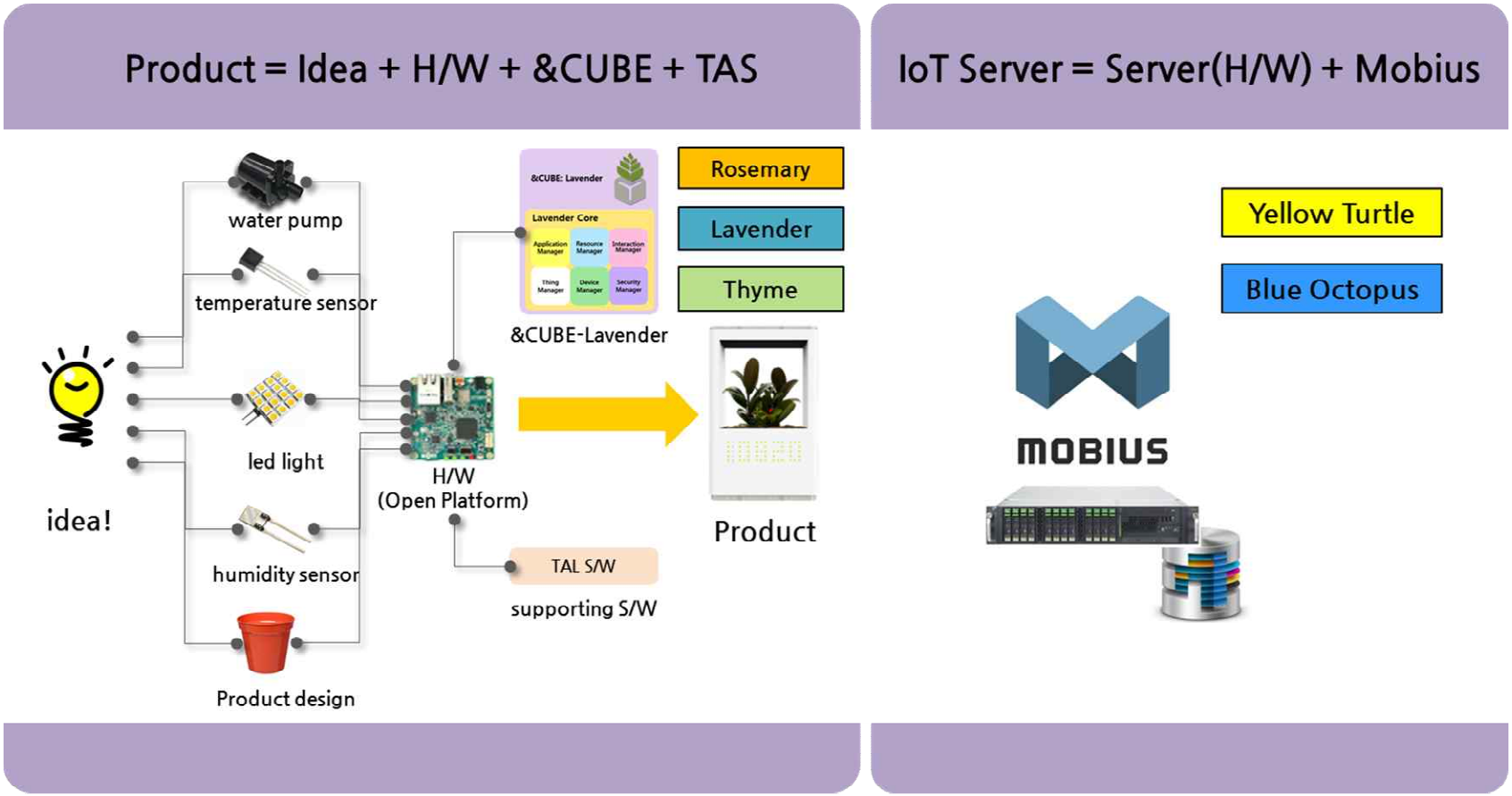
- Development Procedure



*TAS: Thing Adaptation Software

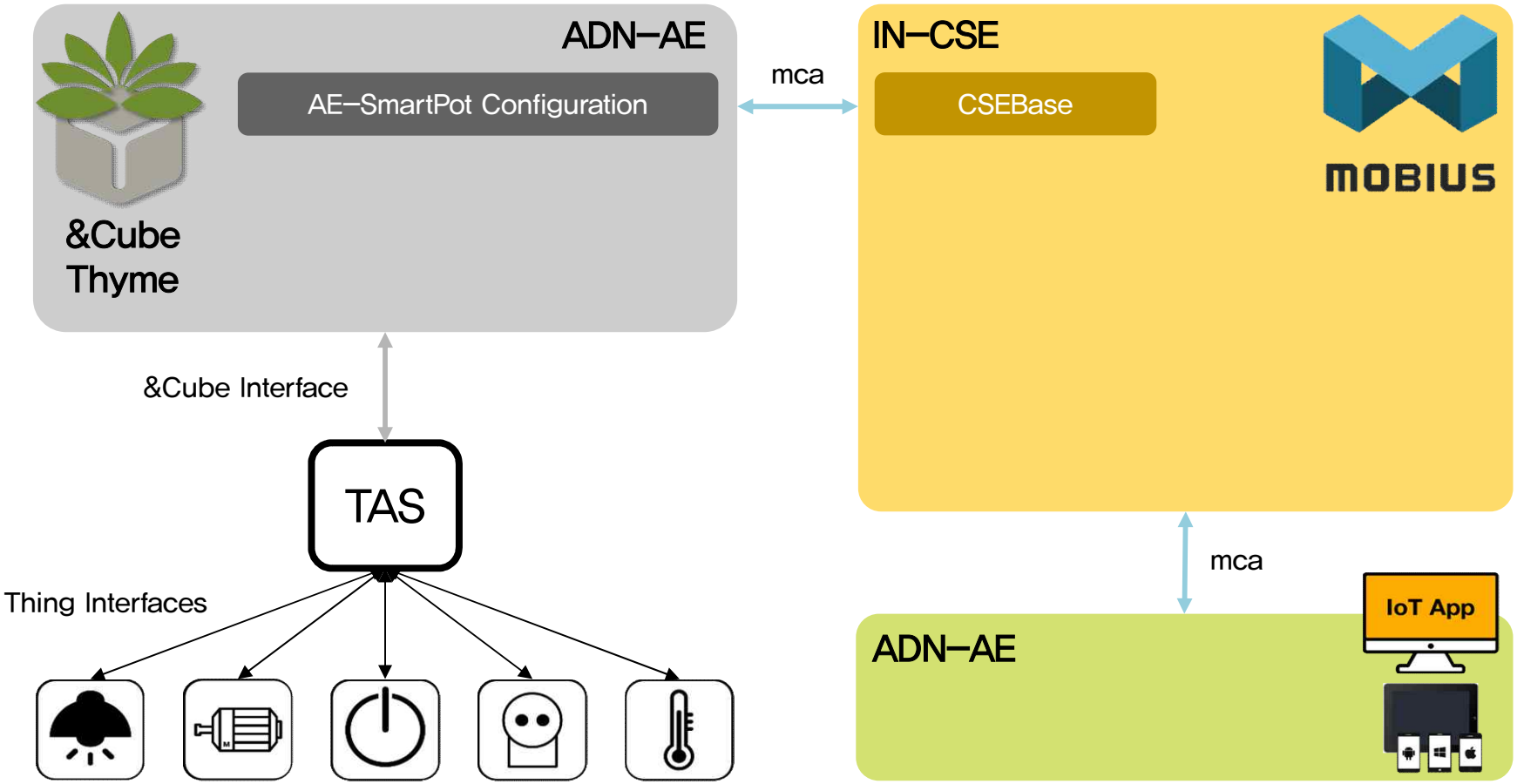
10.3 Utilization

- Development Method



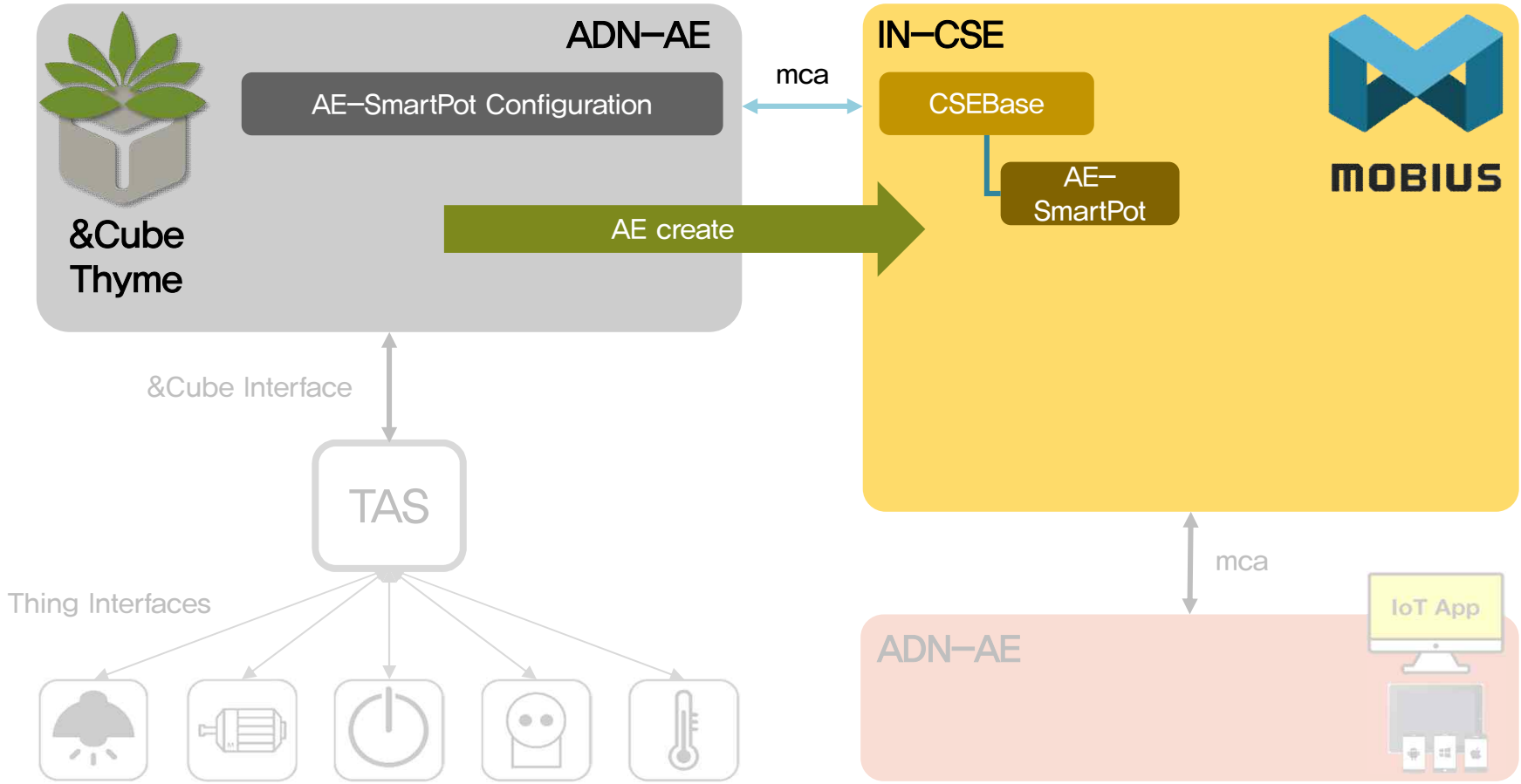
10.3 Utilization

- Basic State



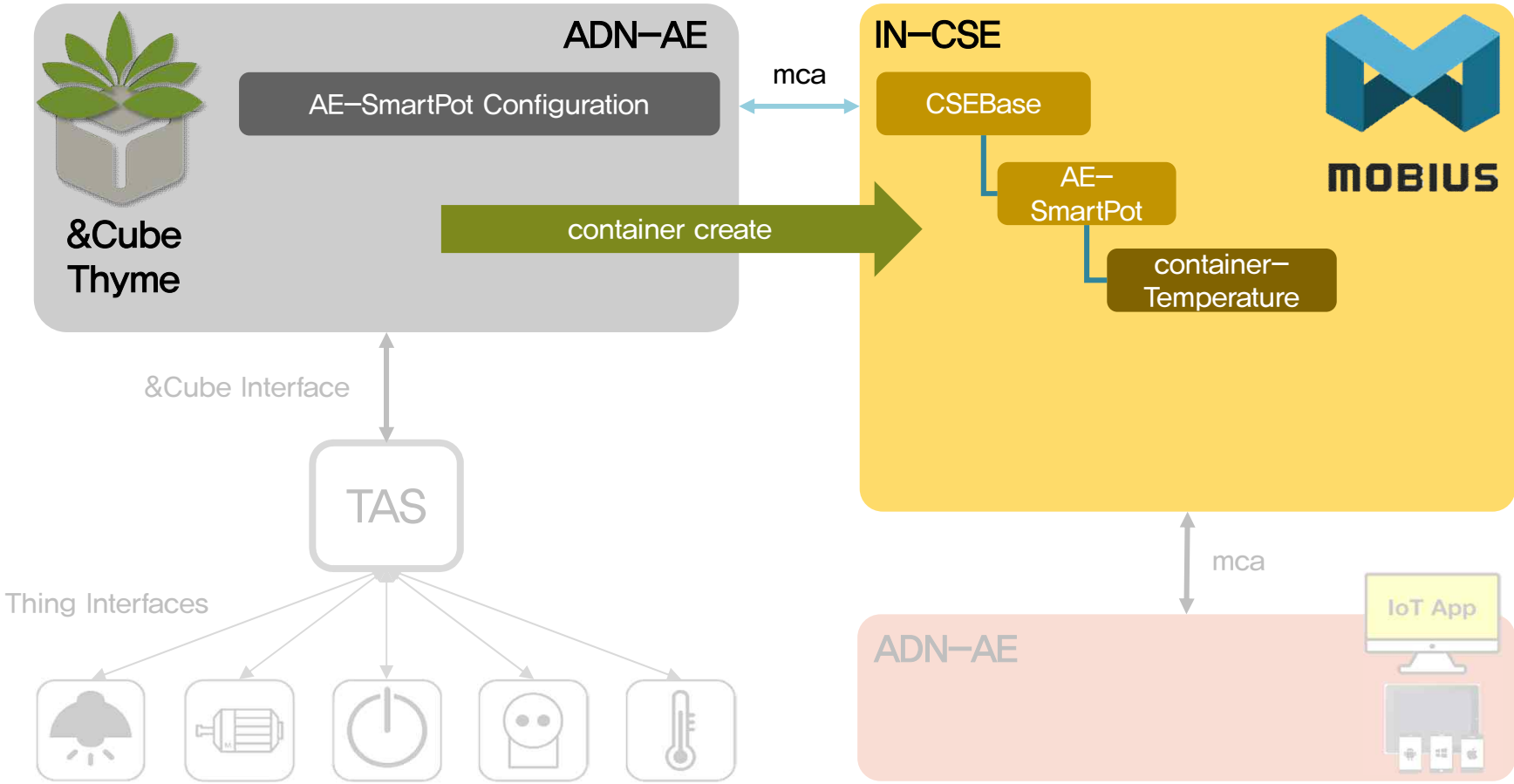
10.3 Utilization

- Device Registration



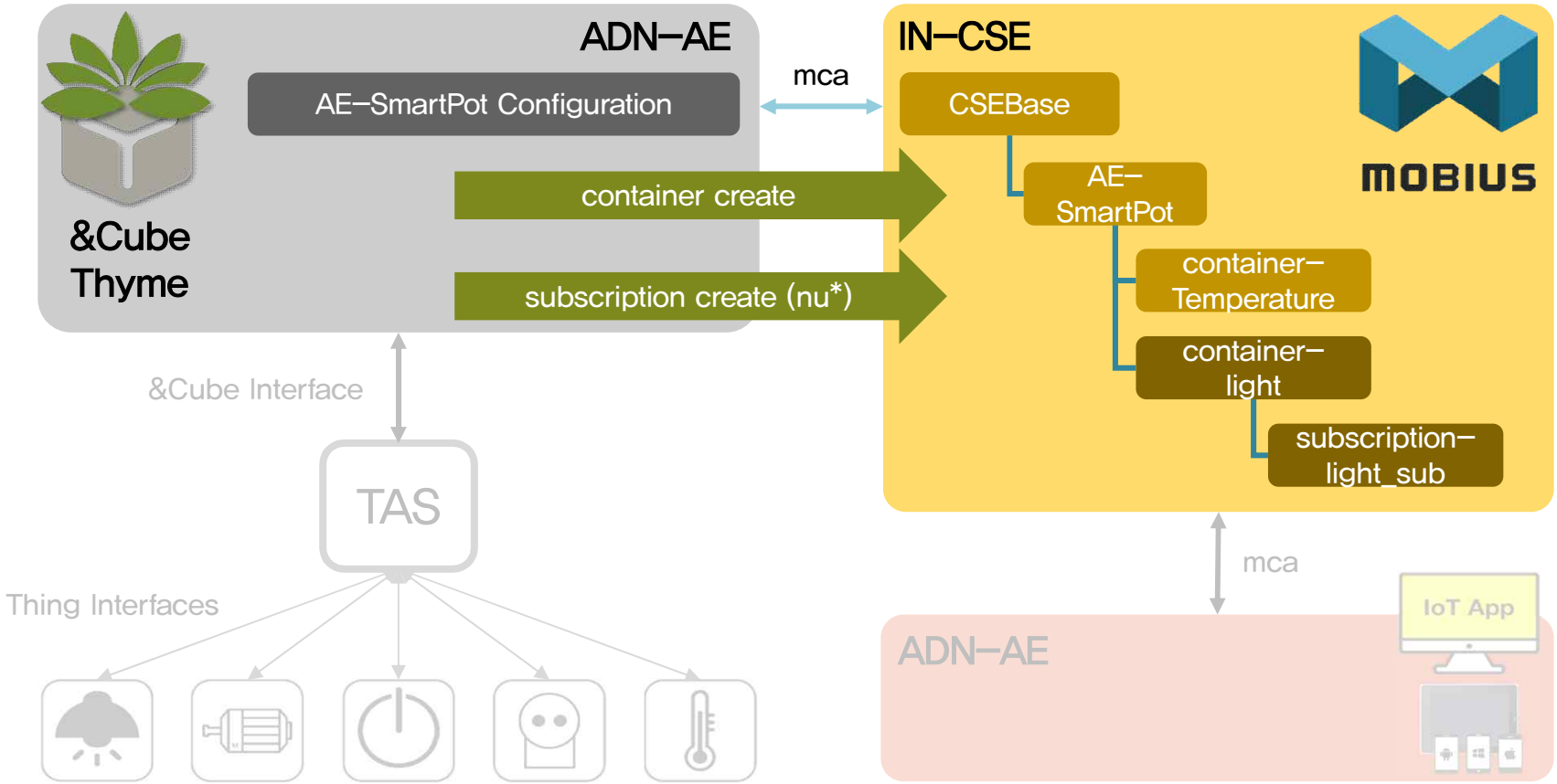
10.3 Utilization

- Thing(sensor) Registration



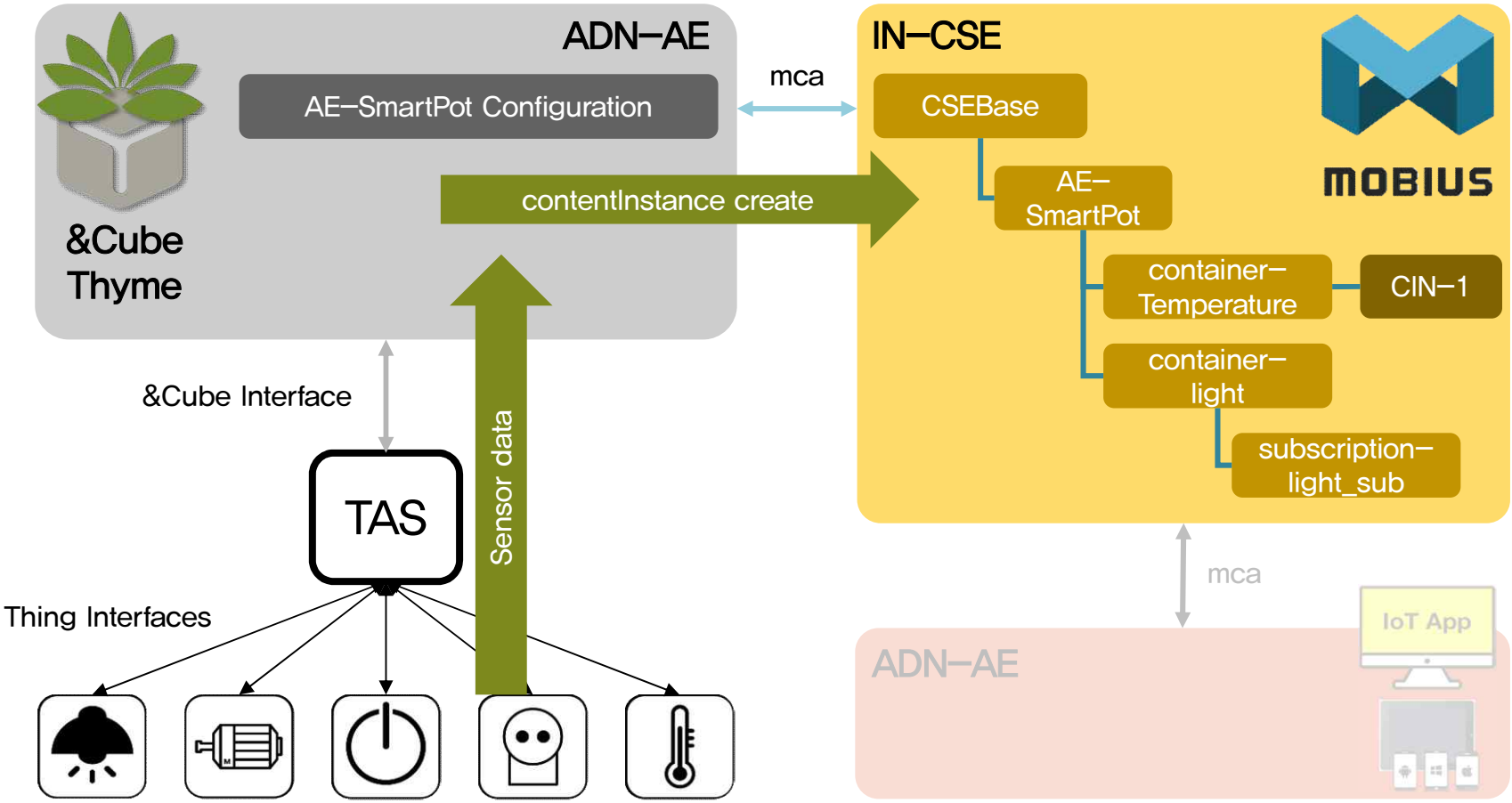
10.3 Utilization

- Thing(actuator) Registration



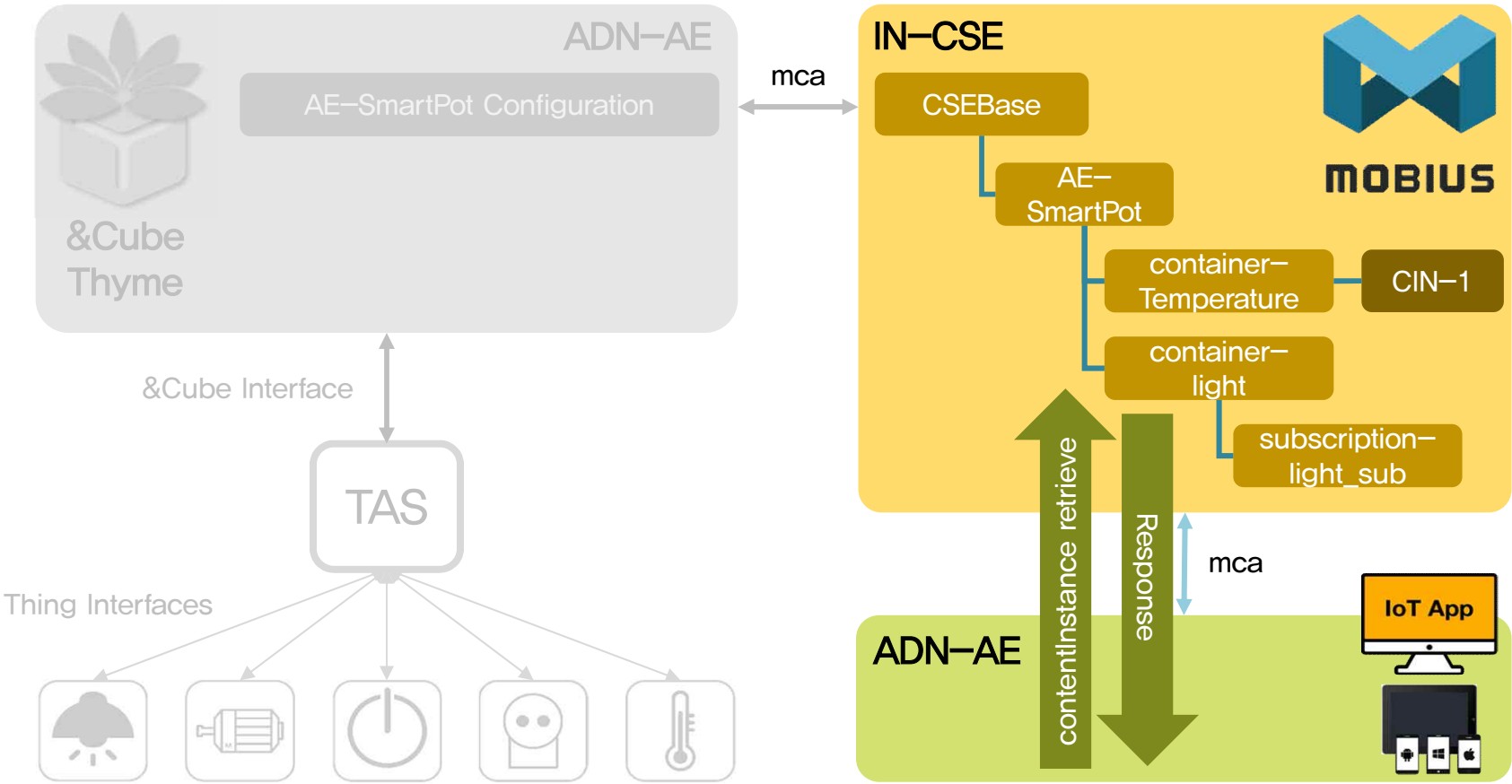
10.3 Utilization

- Thing(sensor) Data Upload



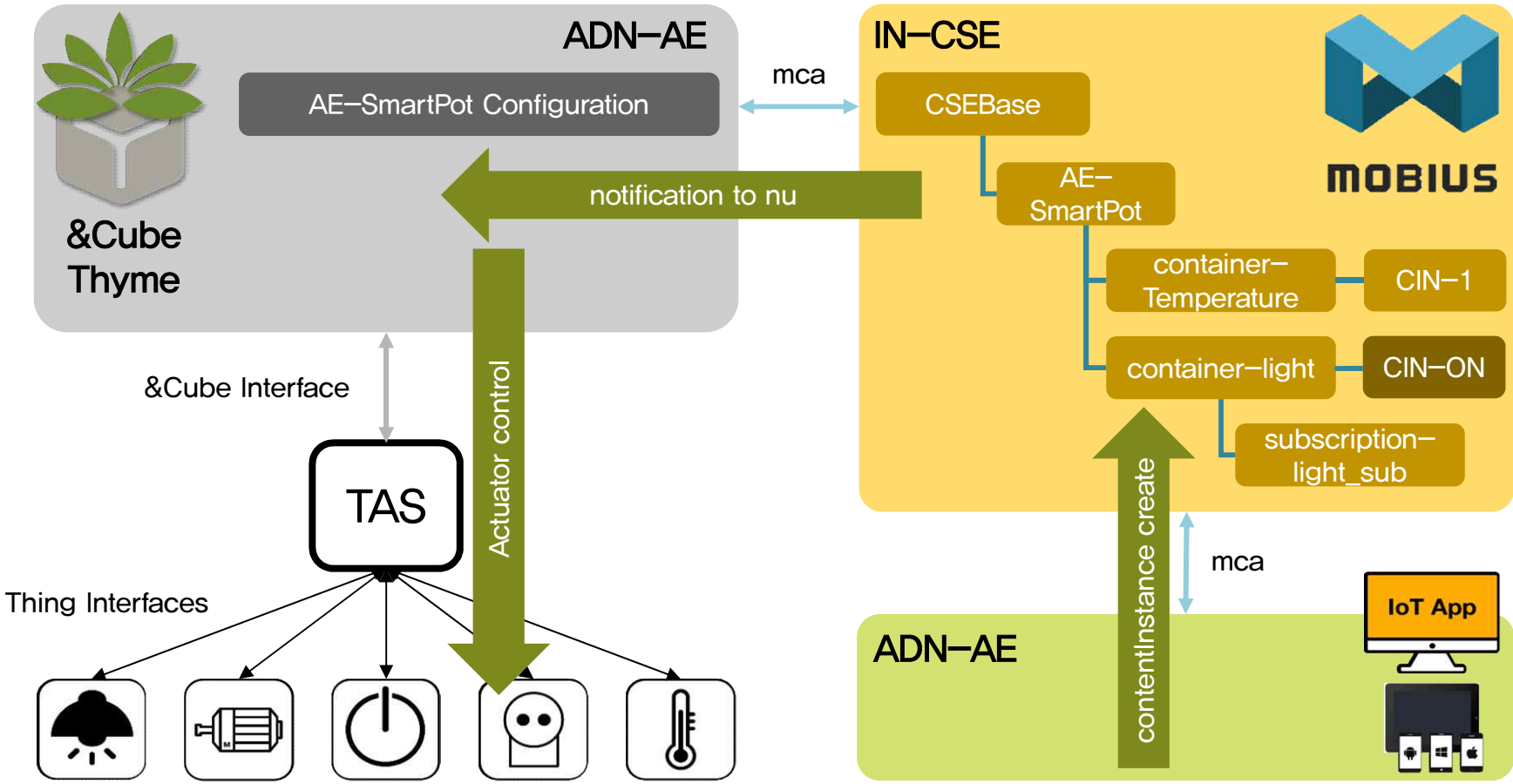
10.3 Utilization

- Thing(sensor) Data Retrieve



10.3 Utilization

- Thing(sensor) Control



10.4 Reference Hardware

▪ Raspberry Pi 3 Model B

- The Raspberry Pi is a series of credit card–sized single–board computers developed in the United Kingdom by the Raspberry Pi Foundation
- Using Raspbian (Debian Linux) operating system(OS)
- Other OS support
- Reference : http://www.raspberrypi.org/

10.4 Reference Hardware

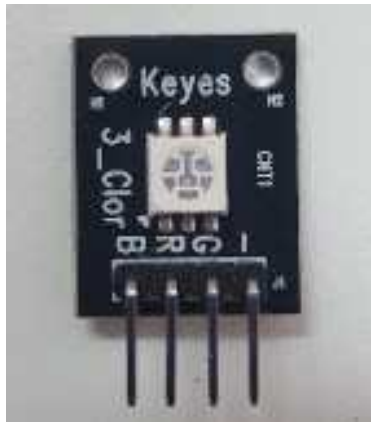
▪ CM1106 CO2 Sensor

- CM1106 is miniature infrared CO2 sensor module for detecting CO2 concentration
- 1. Measurement range: 0–2000ppm, 0–5000ppm
  2. Power supply: DC5V±5%, Pmax0.6W
  3. Dimension: 33*19.7*8.9mm
  4. Resolution:1ppm
  5. Accuracy: ± (50ppm+5% of reading)
  6. Sampling way: diffusion
  7. Response time: 60s
  8. Working condition: 0–50℃,0–95%RH
  9. Signal output: UART , PWM
  10. Life span: 10 years

10.4 Reference Hardware

▪ SMD RGB LED Module (SZH-EK045)

- RGB LED module consists of a full-color SMD LED made by R , G , B three-pin PWM input voltage can be adjusted in three primary colors (red / blue / green) strength in order to achieve full color mixing effect
- 1. using 5050 full-color LED
  2. RGB tricolor limiting resistor to prevent burn
  3. by PWM regulator may be obtained by mixing the three primary colors in different colors
  4. can be an interface with a variety of microcontroller
  5. Operating voltage: 5V
  6. LED drive modes: common cathode drive



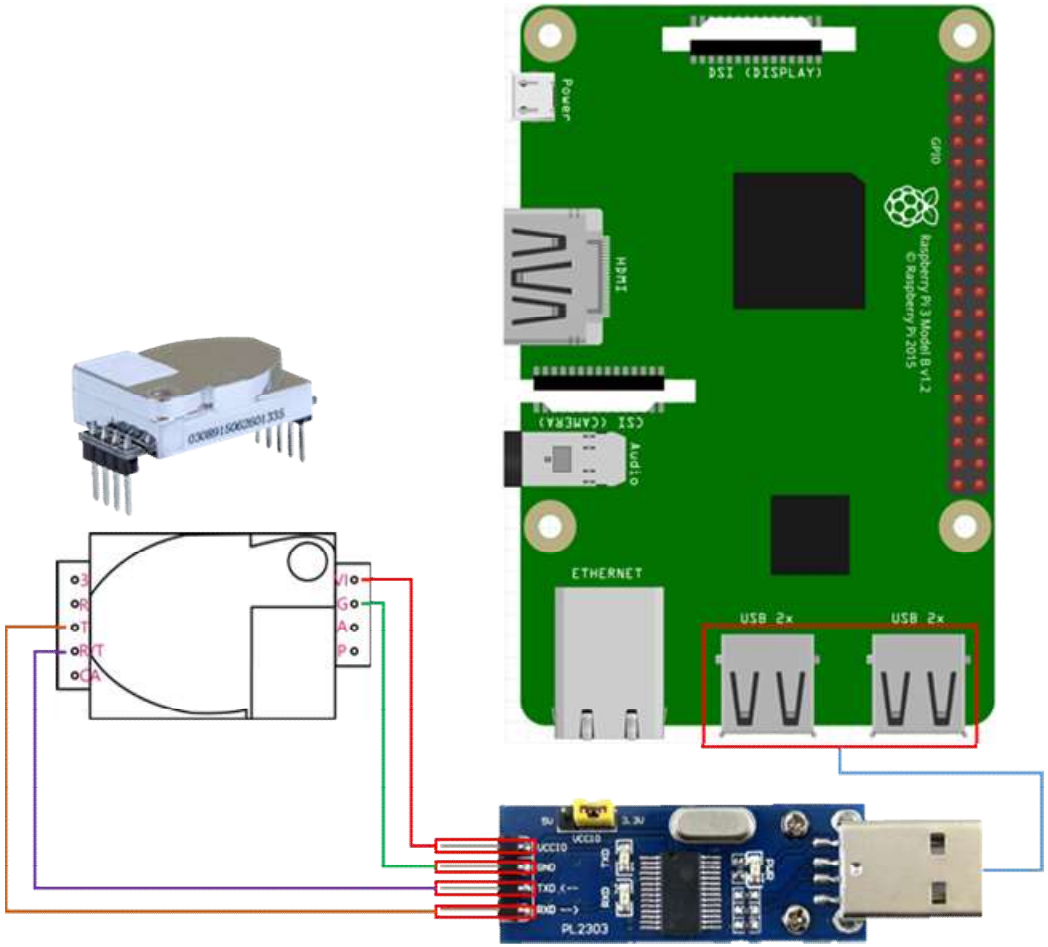
10.4 Reference Hardware

- PL2303 USB UART Board (type A)

  - USB TO UART solution with USB Type A connector
  - 1. PL2303TA onboard (the latest device, compatible with PL2303HXA)
    2. Supports windows XP/7/8/8.1/10/...
    3 power mode : 5V output, 3.3V output, or powered by target board (3.3V–5V)
    4. 3 LEDs: TXD LED, RXD LED, POWER LED

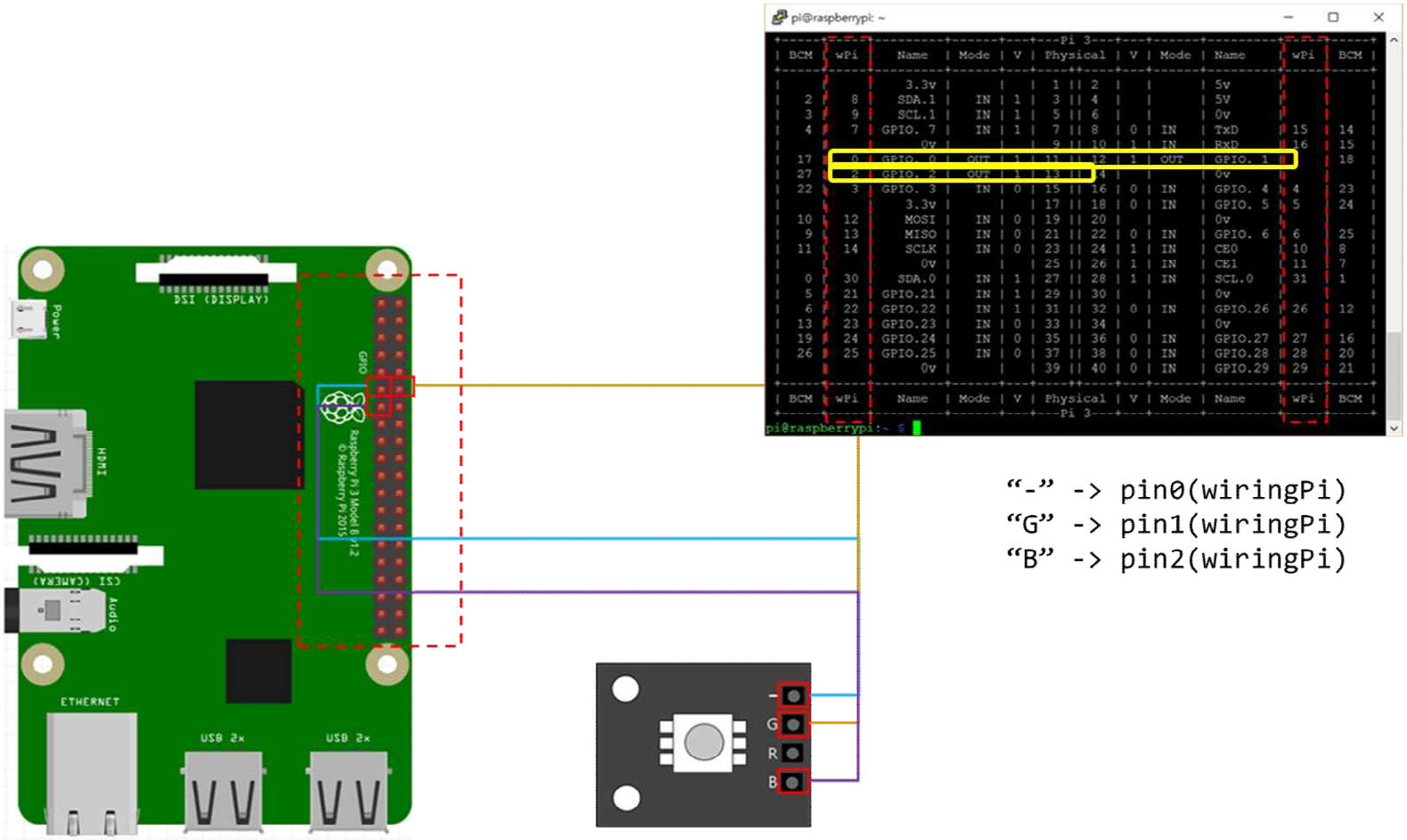
10.5 Runtime Environment

- CO2 sensor setup



10.5 Runtime Environment

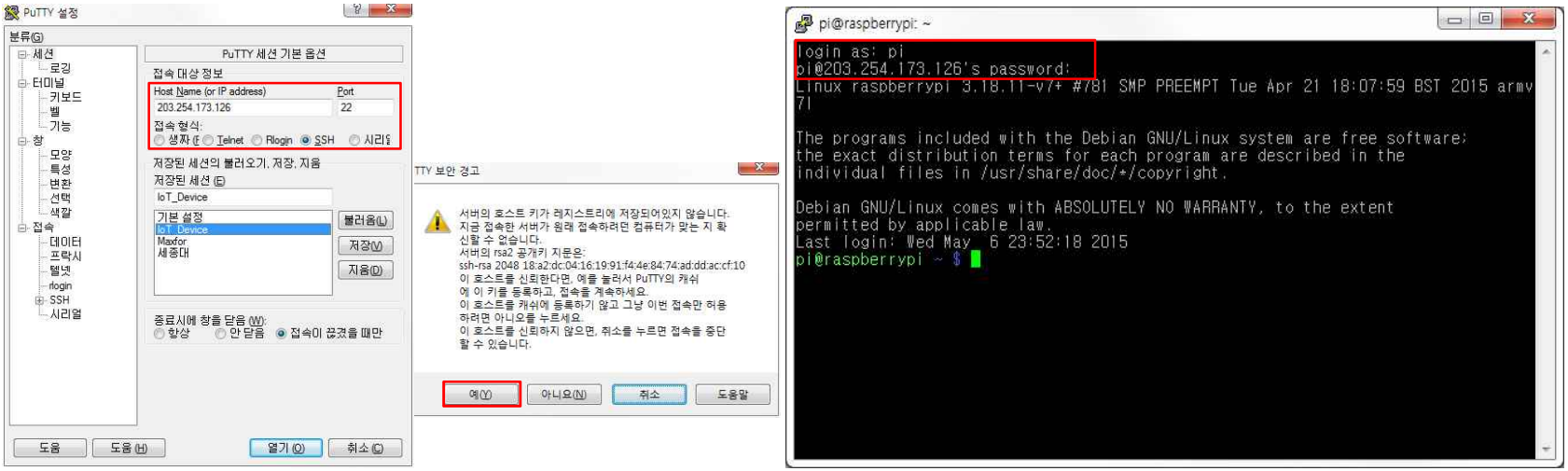
- RGB LED setup




```
"-" -> pin0(wiringPi)
"G" -> pin1(wiringPi)
"B" -> pin2(wiringPi)
```

10.5 Runtime Environment

- SSH client setup in Windows
  - SSH client (PuTTY) download and installation
    http://www.chiark.greenend.org.uk/~sgtatham/putty/
  - PuTTY configuration and connection
    1. Run → Host Name (Raspberry-pi IP address) → Open
    2. Host Key confirm (Yes)
    3. Connected → ID : pi, PW : raspberry (Raspberry-pi default)



10.5 Runtime Environment

▪ Samba file server setup in Raspberry-Pi

▪ Repository Update

```
pi@raspberrypi ~ $ sudo apt-get update
……
Reading package lists… Done
```

▪ Add user

```
pi@raspberrypi ~ $ sudo apt-get install samba samba-common-bin
……
Do you want to continue [Y/n]? Y
```

▪ Samba installation

```
pi@raspberrypi ~ $ sudo smbpasswd -a pi
New SMB password: (Input user password)
Retype new SMB password: (Input user password)
Added user pi.
```

▪ Samba user configuration

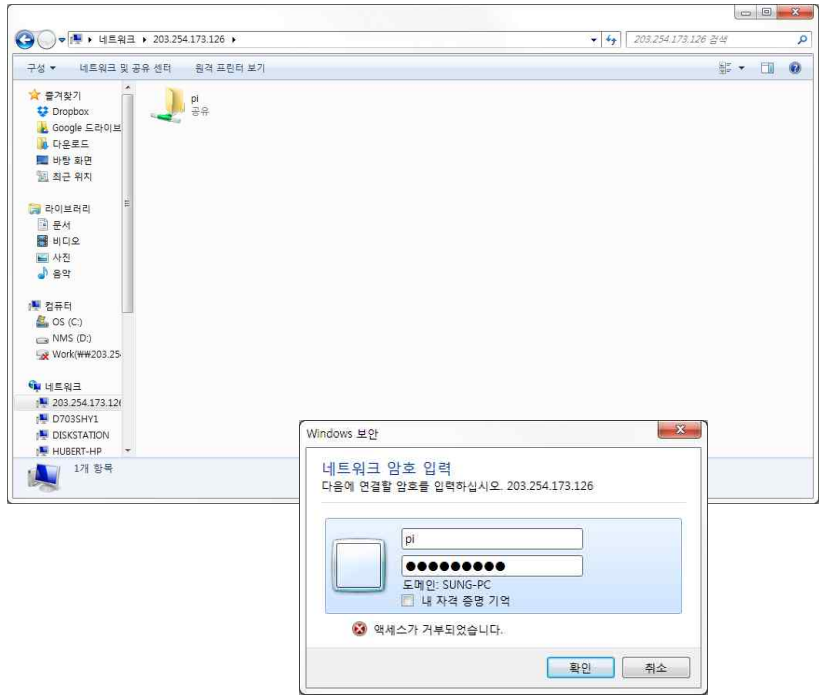
```
pi@raspberrypi ~ $ sudo nano /etc/samba/smb.conf
…… (End of file)
[pi]
comment = raspberry pi folder
path = /home/pi
valid user = pi
writable = yes
browseable = yes
〈Ctrl〉+〈X〉 → Y → 〈Enter〉
```

10.5 Runtime Environment

- Samba file server setup in Raspberry-Pi
  - Repository Update

```
pi@raspberrypi ~ $ sudo service samba restart
[ ok ] Stopping Samba daemons: nmbd smbd.
[ ok ] Starting Samba daemons: nmbd smbd.
```

- Connect to Samba folder in Windows
  - Run File Explorer
  - Input (raspberry-pi IP address)
    ex) ₩₩203.254.173.126
  - pi folder double click
  - Input (ID and password)

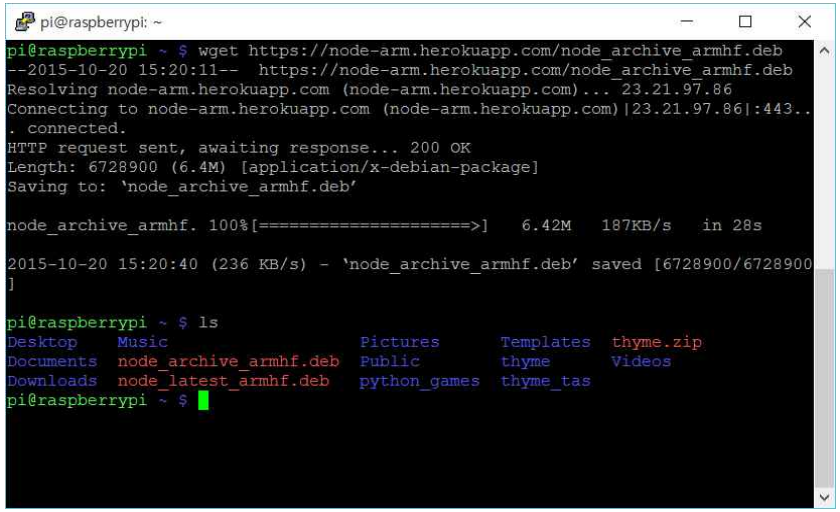


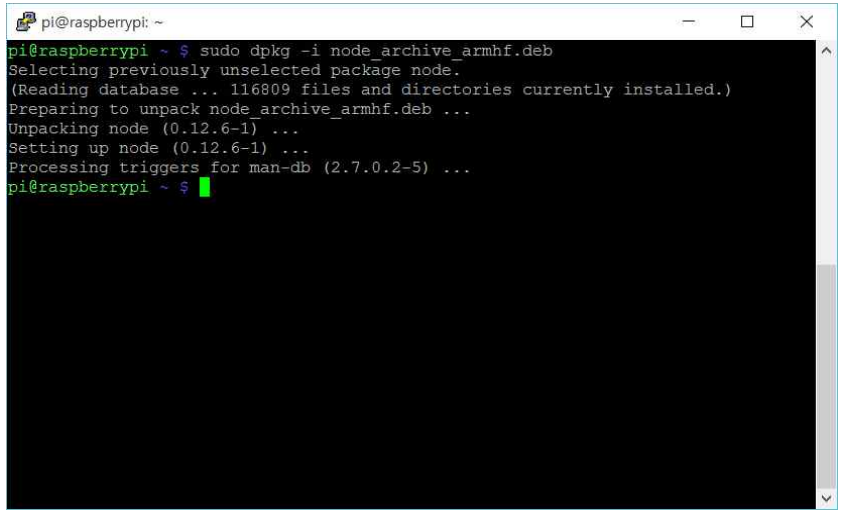
10.5 Runtime Environment

▪ Node.js setup in Raspberry-Pi

▪ Node.js installation

```
pi@raspberrypi ~ $ mkdir node
pi@raspberrypi ~ $ cd node
pi@raspberrypi ~/node $ sudo apt-get update
pi@raspberrypi ~/node $ sudo apt-get upgrade
pi@raspberrypi ~/node $ sudo apt-get remove nodejs
pi@raspberrypi ~/node $ sudo wget https://node-arm.herokuapp.com/node_archive_armhf.deb
pi@raspberrypi ~/node $ sudo dpkg -i node_archive_armhf.deb (Package install command)
pi@raspberrypi ~/node $ node -v (Node.js version check command)
pi@raspberrypi ~/node $ npm -v (External library installation tool npm version check command)
```





Assignment

- Assignment 10

Raspberry pi 와 임의의 센서 디바이스를 연동 한 후 해당 센서를 컨트롤
또는 조회하는 프로그램을 작성해 보시오.
(프로그래밍 언어 제한 없음)

Reference

[10.2] Mobius – &Cube

Interconnection Structure

그림 출처: http://www.iotocean.org/main/

[10.3] Utilization

How IoT Devices Work

그림 출처: http://www.iotocean.org/main/

Development Method

그림 출처: http://www.iotocean.org/main/

11장. &Cube 설치와 모비우스

Chapter 11. Installation of &Cube and Mobius

11.1 Install &Cube and TAS
11.2 Run &Cube and TAS
11.3 Test Device
11.4 Install Mobius Yellow Turtle
11.5 Run Mobius

- Assignment
- Reference

강의 개요

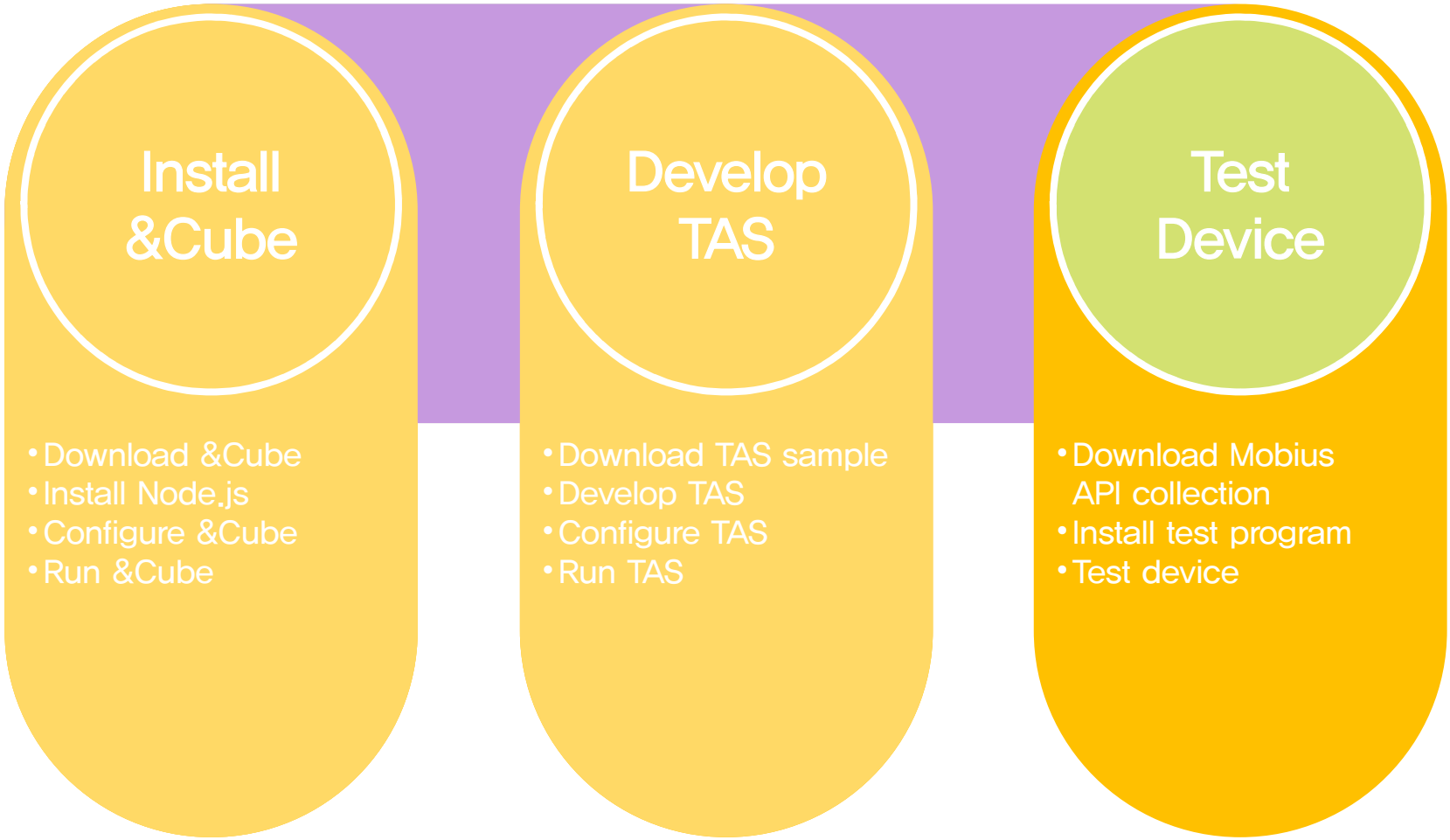
- **강의 목표**

  &Cube를 동작 시키기 위한 방법과 모비우스와의 연동 Flow를 이해하고
  로컬 모비우스 설치 방법을 학습한다.

- **강의 내용**

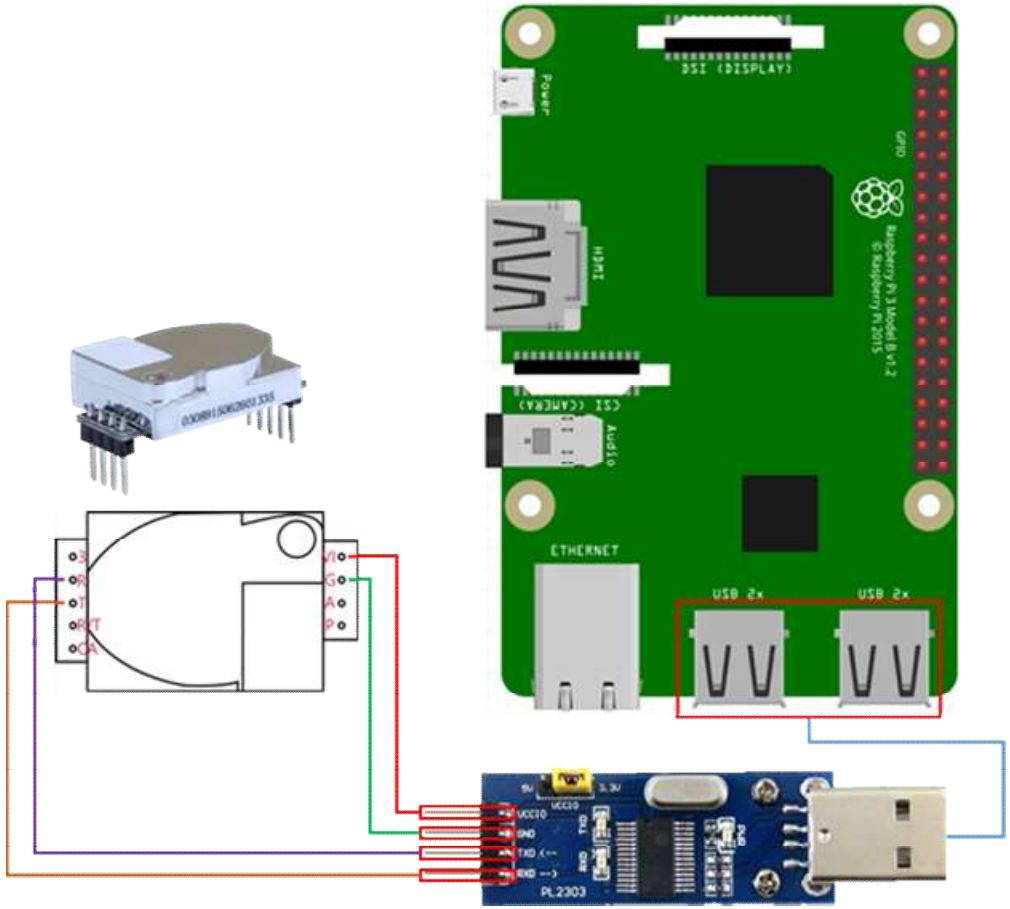
  - &Cube 구동
  - TAS 개발 및 구동
  - 디바이스 연동 및 조회
  - 디바이스 컨트롤
  - Mobius 설치 실습
  - Mobius 플랫폼을 통한 oneM2M 표준 이해

11.1 Install &Cube and TAS



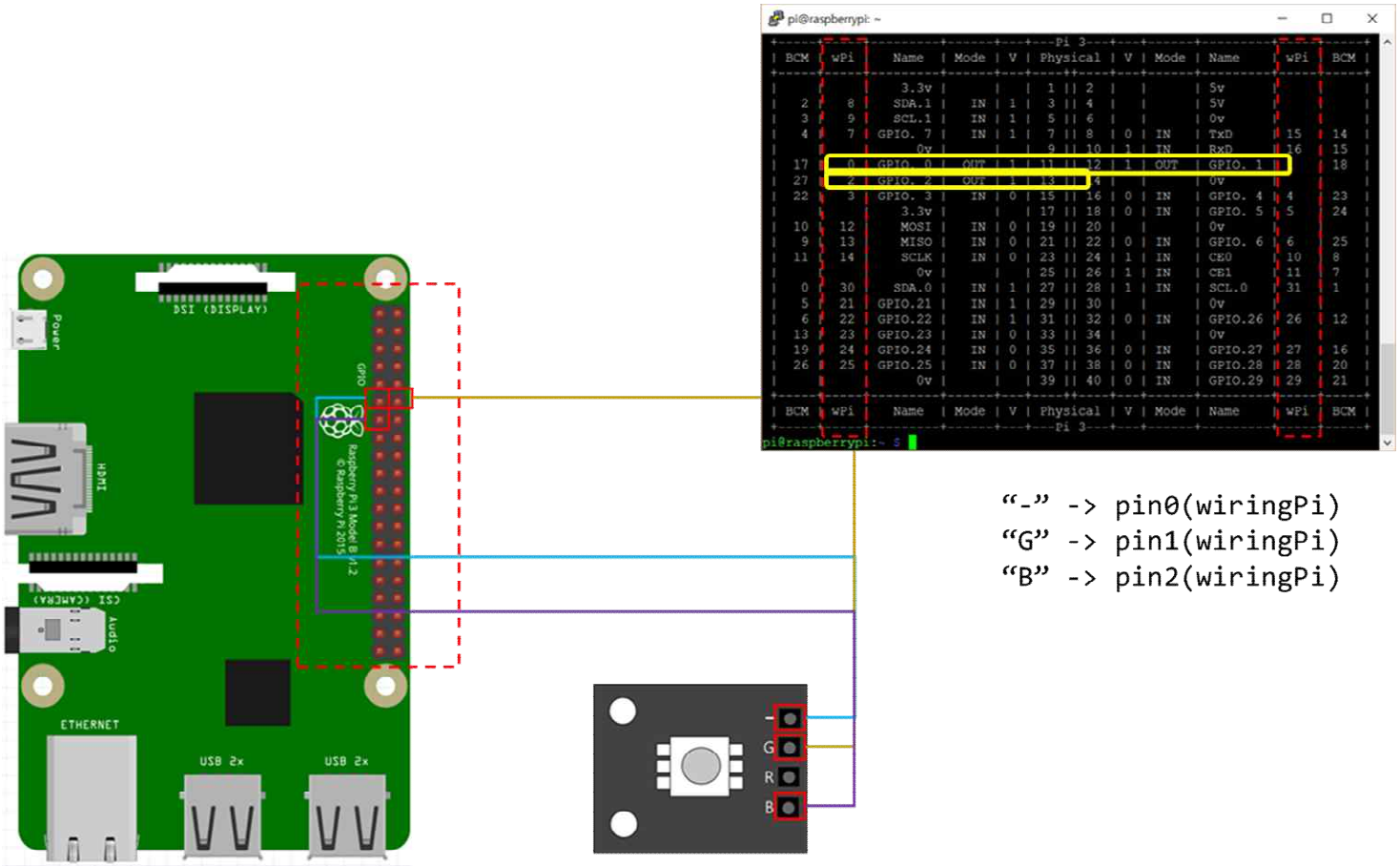
11.1 Install &Cube and TAS

- CO2 sensor setup



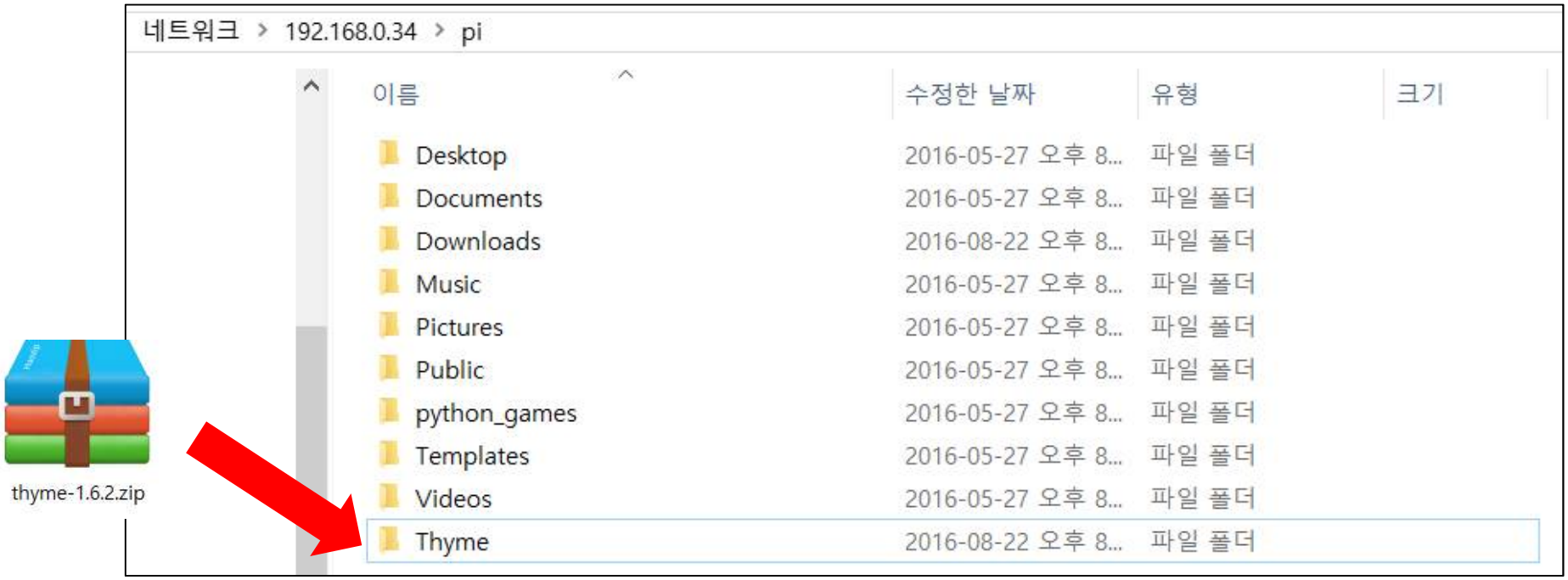
11.1 Install &Cube and TAS

- RGB LED setup



11.1 Install &Cube and TAS

- &Cube: Thyme and TAS Download and Copy to Raspberry Pi
  - Download &Cube: Thyme and TAS in www.iotocean.org
  - Copy the Thyme and TAS source file downloaded to Raspberry Pi with samba



11.1 Install &Cube and TAS

- &Cube: Thyme Setup with Raspberry Pi
  - Login Raspberry Pi with putty SSH program
  - Create Thyme folder and install Thyme

```
pi@raspberrypi ~/node $ mkdir thyme
pi@raspberrypi ~/node $ cd thyme
pi@raspberrypi ~/node/thyme $ unzip thyme-1.6.x.zip
pi@raspberrypi ~/node/thyme $ sudo npm install
```

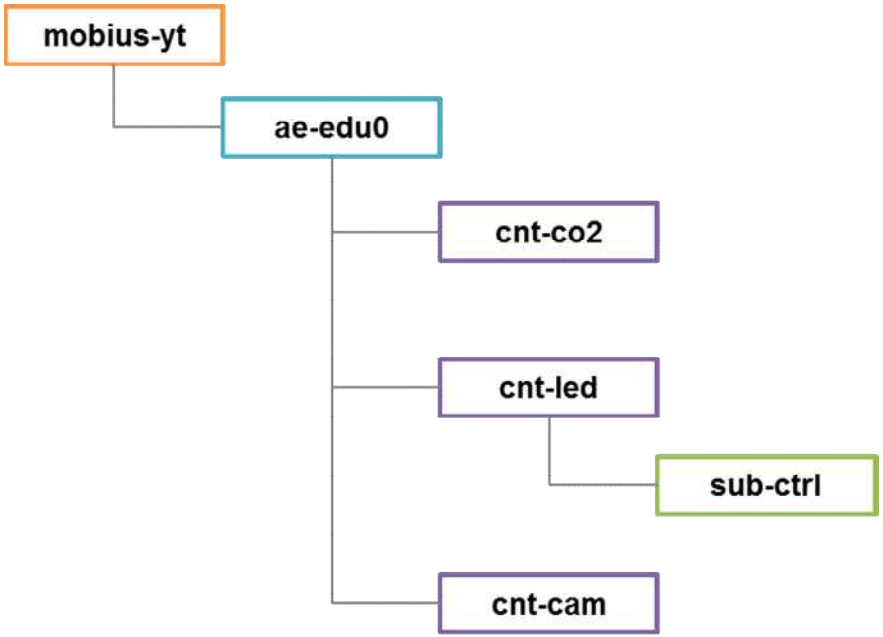
```
mqtt@1.14.0 node_modules/mqtt
├── inherits@2.0.1
├── reinterval@1.1.0
├── xtend@4.0.1
├── help-me@0.1.0
├── minimist@1.2.0
├── readable-stream@1.0.34 (string_decoder@0.10.31, isarray@0.0.1, core-util-is@
1.0.2)
├── commist@1.0.0 (leven@1.0.2)
├── mqtt-connection@2.1.1 (through2@0.6.5, reduplexer@1.1.0)
├── mqtt-packet@3.4.7 (bl@0.9.5)
├── end-of-stream@1.1.0 (once@1.3.3)
├── pump@1.0.1 (once@1.3.3)
├── concat-stream@1.5.1 (typedarray@0.0.6, readable-stream@2.0.6)
├── split2@2.1.0 (through2@2.0.1)
└── websocket-stream@3.2.1 (ws@1.1.1, through2@2.0.1, duplexify@3.4.5)

xmlbuilder@2.6.5 node_modules/xmlbuilder
└── lodash@3.10.1

xml2js@0.4.17 node_modules/xml2js
├── sax@1.2.1
└── xmlbuilder@4.2.1 (lodash@4.15.0)
pi@raspberrypi:~/Thyme $
```

```
pi@raspberrypi:~/Thyme $ unzip thyme-1.6.2.zip
Archive:  thyme-1.6.2.zip
  inflating: thyme.js
  inflating: thyme/adn.js
  inflating: thyme/mqtt_adn.js
  inflating: app.js
  inflating: conf.json
  inflating: conf.json.old
  inflating: mqtt_app.js
  inflating: notificationURI.txt
  inflating: package.json
pi@raspberrypi:~/Thyme $
```

11.1 Install &Cube and TAS

- &Configure &Cube: Thyme
  - Open configuration file and edit

```
pi@raspberrypi ~/node/thyme $ sudo nano conf.json
…… (Edit configuration file)
〈Ctrl〉+〈X〉 → Y → 〈Enter〉
```




```
{
    "useprotocol": "http",
    "cse": {
        "cbhost": "203.253.128.151",
        "cbport": "7579",
        "cbname": "mobius-yt",
        "cbcseid": "/mobius-yt"
    },
    "ae": {
        "aeid": "S",
        "appid": "0.2.481.1.1",
        "appname":  "ae-edu0",
        "appport": "9727",
        "bodytype": "json",
        "tasport": "3105"
    },
    "cnt": [
        {
            "parentpath": "/ae-edu0",
            "ctname":  "cnt-co2"
        },
        {
            "parentpath": "/ae-edu0",
            "ctname":  "cnt-led"
        },
        {
            "parentpath": "/ae-edu0",
            "ctname":  "cnt-cam"
        }
    ],
    "sub": [
        {
            "parentpath": "/ae-edu0/cnt-led",
            "subname": "sub-ctrl",
            "nu": "mqtt://autoset"
        }
    ]
}
```

11.1 Install &Cube and TAS

▪ &Configure &Cube: Thyme

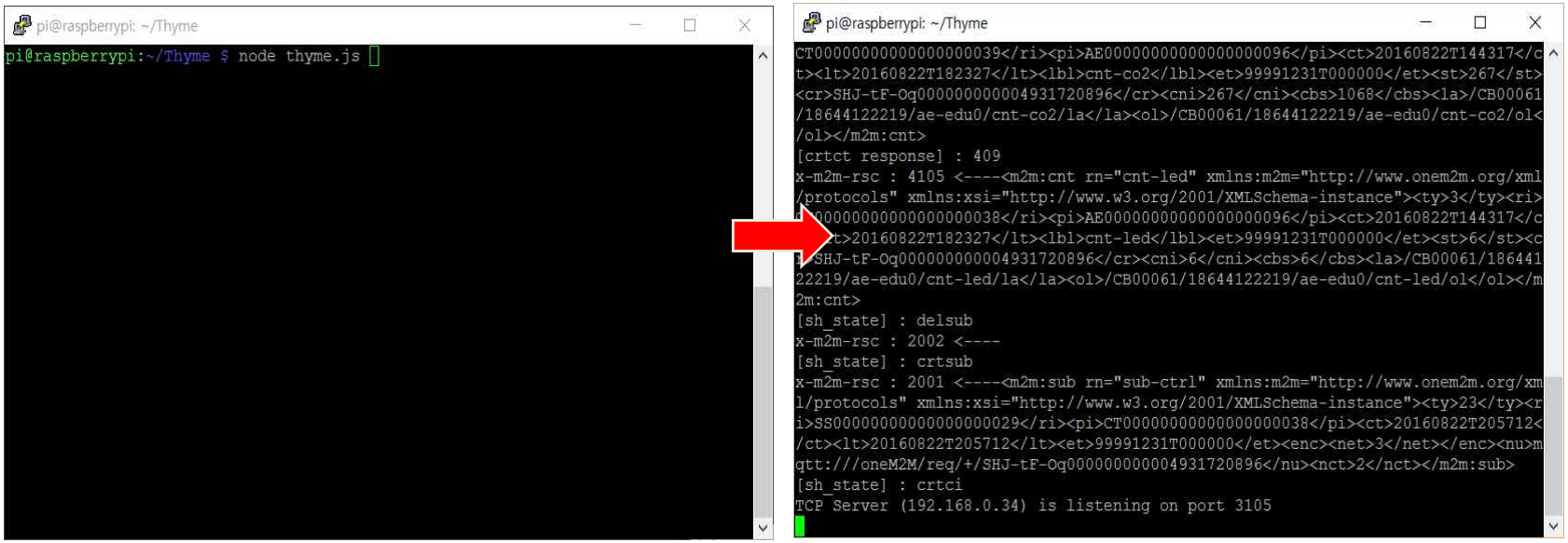
▪ Parameters

Parameter	Description	Content
useprotocol	Configure communication protocol between thyme and mobius	String (opt: “http”, “mqtt”, “coap”)
cse	Common Service Entity configuration area	JSON Object
cbhost	Configure CSEBase host server IP	String (ex: “192.168.0.2”)
cbport	Configure CSEBase host server port	String (ex: “7579”)
cbname	Configure CSEBase resource name	String (ex: “mobius-yt”)
cbcseid	Configure CSEBase resource ID	String (ex: “/mobius-yt”)
mqttport	Configure MQTT broker host server port	String (ex: “1883”)
ae	Application Entity configuration area	JSON Object
aeid	Configure AE ID	String (ex: “S12345”) note: must start with “S” or “C”
appid	Configure Application ID	String (ex: “0.13123.0012”)
appname	Configure AE name	String (ex: “example”)
bodytype	Configure message type of request/response	String (opt: “json”, “xml”)
tasport	Configure TCP port for communication with TAS	String (ex: “3105”) note: must same with TAS config info
cnt	Container configuration area	JSON Array
parentpath	Configure parent path of creating container	String (ex: “/example”) note: must start with “/”
ctname	Configure name of creating container	String (ex: “temp”)
sub	Subscription configuration area	JSON Array
parentpath	Configure parent path of creating subscription	String (ex: “/example/ctrl)” note: must start with “/”
subname	Configure name of creating subscription	String (ex: “sub”)
nu	Configure notification URL of subscription	String (ex: “mqtt://192.168.0.12/S123145”) note: MQTT case must match “mqtt://[ip]/[AE-ID]” format HTTP case must match “http://[ip]/[path]”

11.1 Install &Cube and TAS

- Run &Cube: Thyme
  - Input &Cube: Thyme run command

```
pi@raspberrypi ~/node/thyme $ node thyme.js
```



11.1 Install &Cube and TAS

▪ TAS Setup with Raspberry Pi

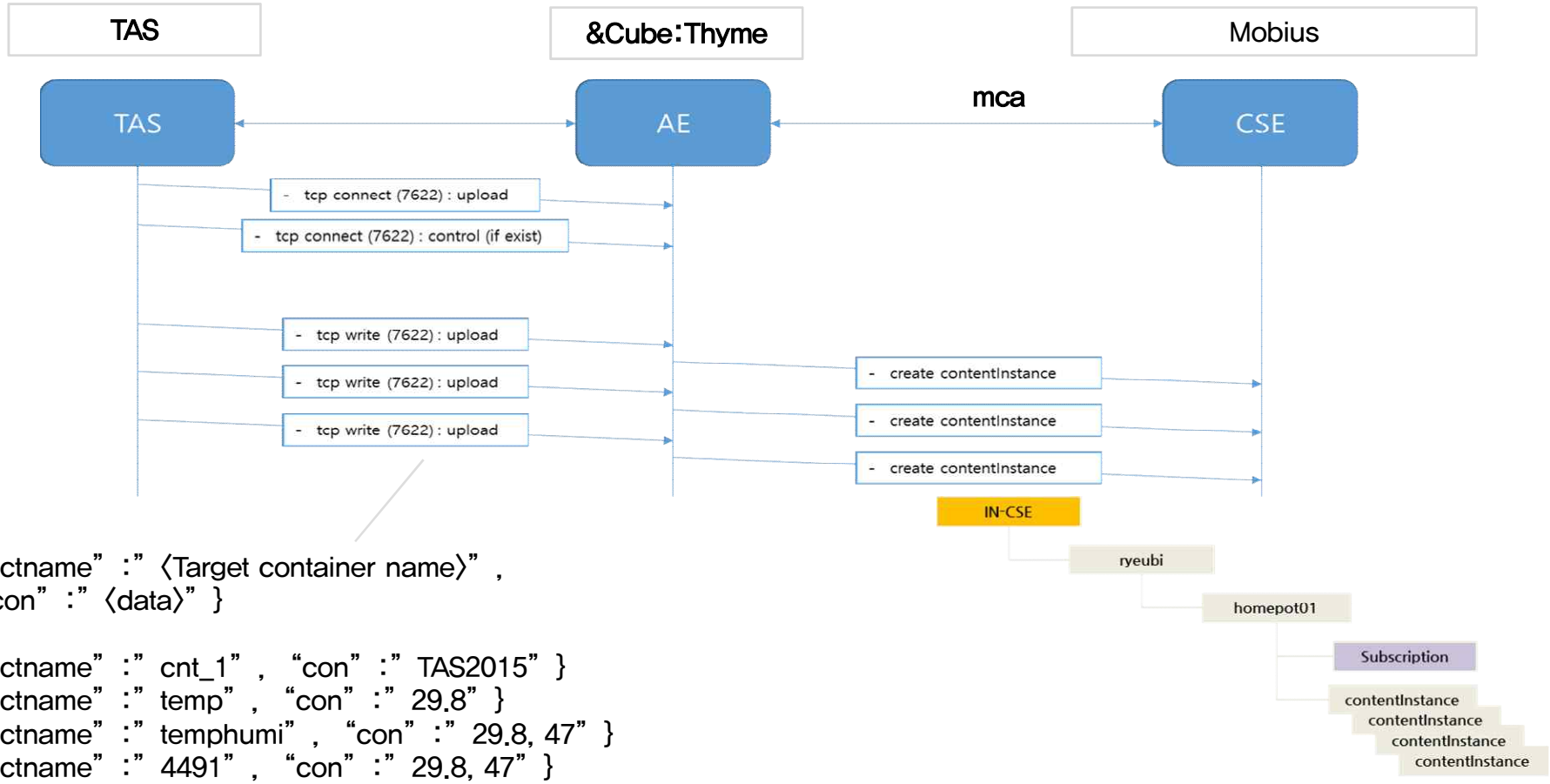
▪ Create TAS folder and install Thyme

```
pi@raspberrypi ~/node $ mkdir tas_co2
pi@raspberrypi ~/node $ cd tas_co2
pi@raspberrypi ~/node/tas_co2 $ unzip keti_tas_thyme-v1.X.zip
pi@raspberrypi ~/node/tas_co2 $ sudo npm install
```

```
mqtt@1.14.0 node_modules/mqtt
├── inherits@2.0.1
├── reinterval@1.1.0
├── xtend@4.0.1
├── help-me@0.1.0
├── minimist@1.2.0
├── readable-stream@1.0.34 (string_decoder@0.10.31, isarray@0.0.1, core-util-is@
1.0.2)
├── commist@1.0.0 (leven@1.0.2)
├── mqtt-connection@2.1.1 (through2@0.6.5, reduplexer@1.1.0)
├── mqtt-packet@3.4.7 (bl@0.9.5)
├── end-of-stream@1.1.0 (once@1.3.3)
├── pump@1.0.1 (once@1.3.3)
├── concat-stream@1.5.1 (typedarray@0.0.6, readable-stream@2.0.6)
├── split2@2.1.0 (through2@2.0.1)
└── websocket-stream@3.2.1 (ws@1.1.1, through2@2.0.1, duplexify@3.4.5)

xmlbuilder@2.6.5 node_modules/xmlbuilder
└── lodash@3.10.1

xml2js@0.4.17 node_modules/xml2js
├── sax@1.2.1
└── xmlbuilder@4.2.1 (lodash@4.15.0)
```

11.1 Install &Cube and TAS

- TAS (Thing Adaptation Software) Flow



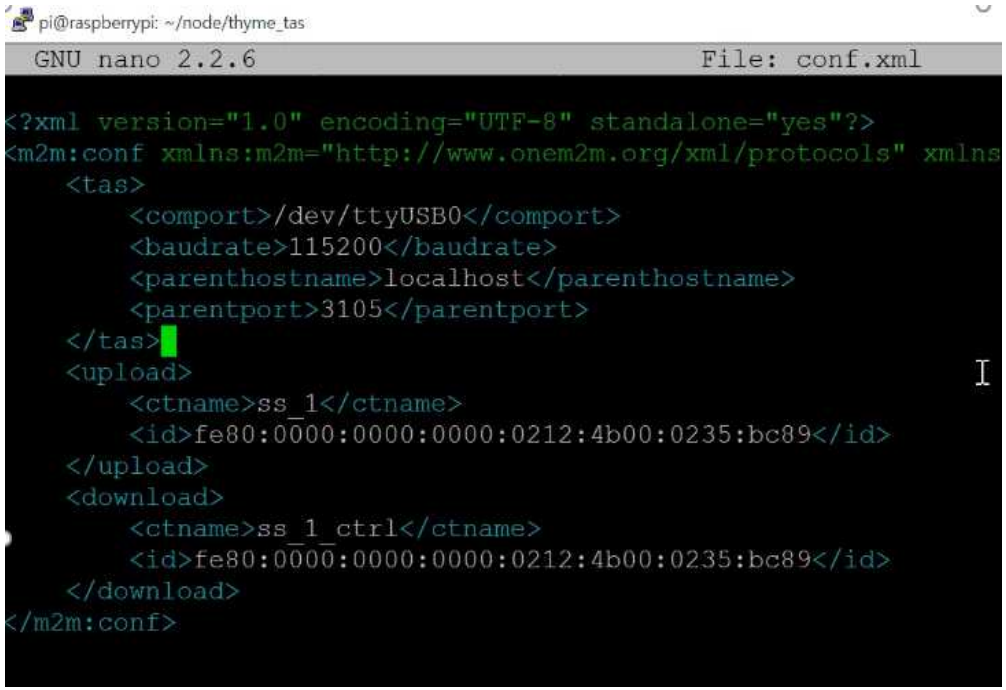
{ “ctname” :” 〈Target container name〉” ,
“con” :” 〈data〉” }
예)
{ “ctname” :” cnt_1” , “con” :” TAS2015” }
{ “ctname” :” temp” , “con” :” 29.8” }
{ “ctname” :” temphumi” , “con” :” 29.8, 47” }
{ “ctname” :” 4491” , “con” :” 29.8, 47” }

11.1 Install &Cube and TAS

▪ Configure TAS

▪ Open configuration file and edit

```
pi@raspberrypi ~/node/tas_co2 $ sudo nano conf.json
…… (Edit configuration file)
〈Ctrl〉+〈X〉 → Y → 〈Enter〉
```




```
〈?xml version="1.0" encoding="UTF-8"
standalone="yes"?〉
〈m2m:conf
xmlns:m2m="http://www.oneM2M.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"〉
    〈tas〉
        〈comport〉/dev/ttyUSB0〈/comport〉
        〈baudrate〉9600〈/baudrate〉
        〈parenthostname〉localhost〈/parenthostname〉
        〈parentport〉7622〈/parentport〉
    〈/tas〉
    〈upload〉
        〈ctname〉cnt-co2〈/ctname〉

〈id〉fe80:0000:0000:0000:0212:4b00:0235:bce5〈/id〉
    〈/upload〉
〈/m2m:conf〉
```

11.1 Install &Cube and TAS

▪ &Configure &Cube: Thyme

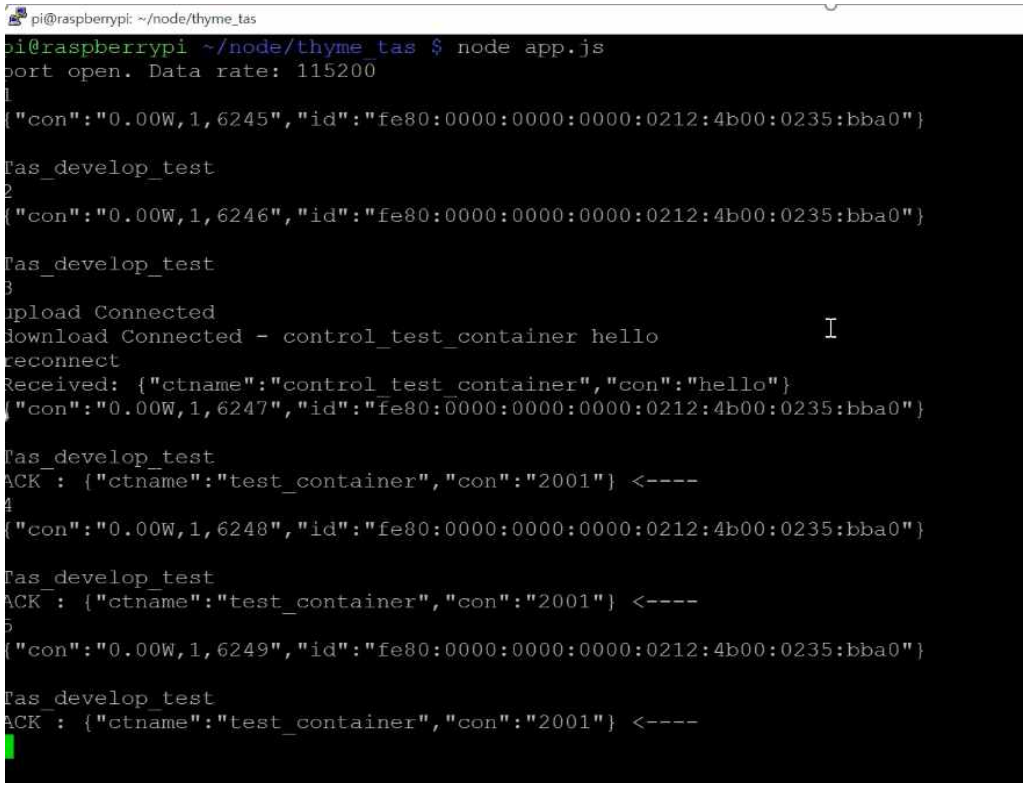
▪ Parameters

Parameter	Description	Content
tas	Thing Adaptation Software configuration area	Low level XML tag
comport	Configure device serial port name	String (Linux ex: “/dev/ttyUSB0”, Win ex: “COM0”)
baudrate	Configure device serial port baud rate	String (ex: “115200”
parenthostname	Configure thyme TCP Server IP number	String (ex: “127.0..0.1”)
parentport	Configure thyme TCP Server port number	String (ex: “3105”)
upload	Sensor data upload configuration area	Low level XML tag
download	Actuator command download configuration area	Low level XML tag
ctname	Configure container name for mapping with thyme	String (ex: temp)
id	Configure device ID for mapping data container	String

11.2 Run &Cube and TAS

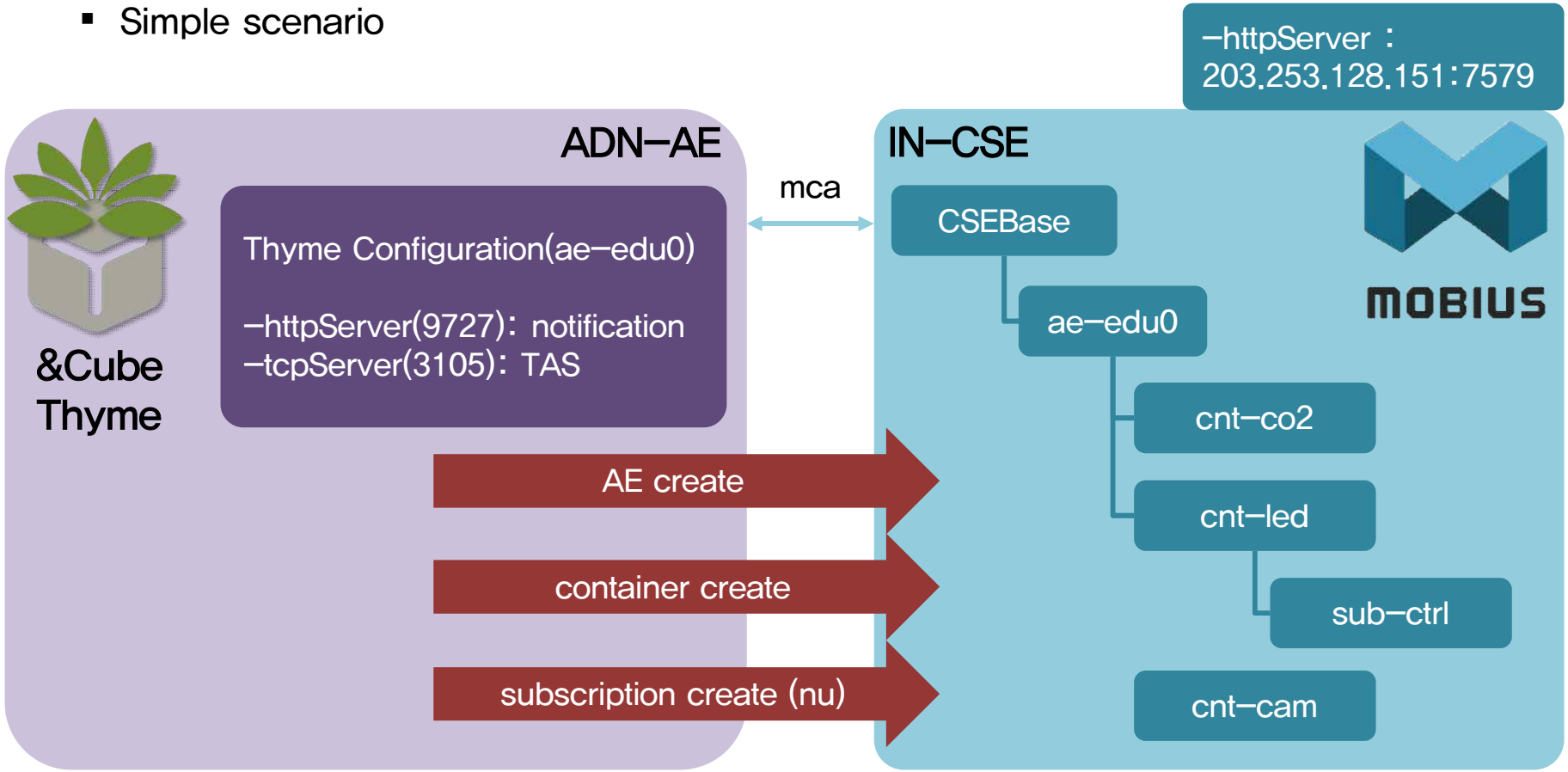
- Run TAS
  - Input TAS run command

```
pi@raspberrypi ~/node/tas_co2 $ node app.js
```



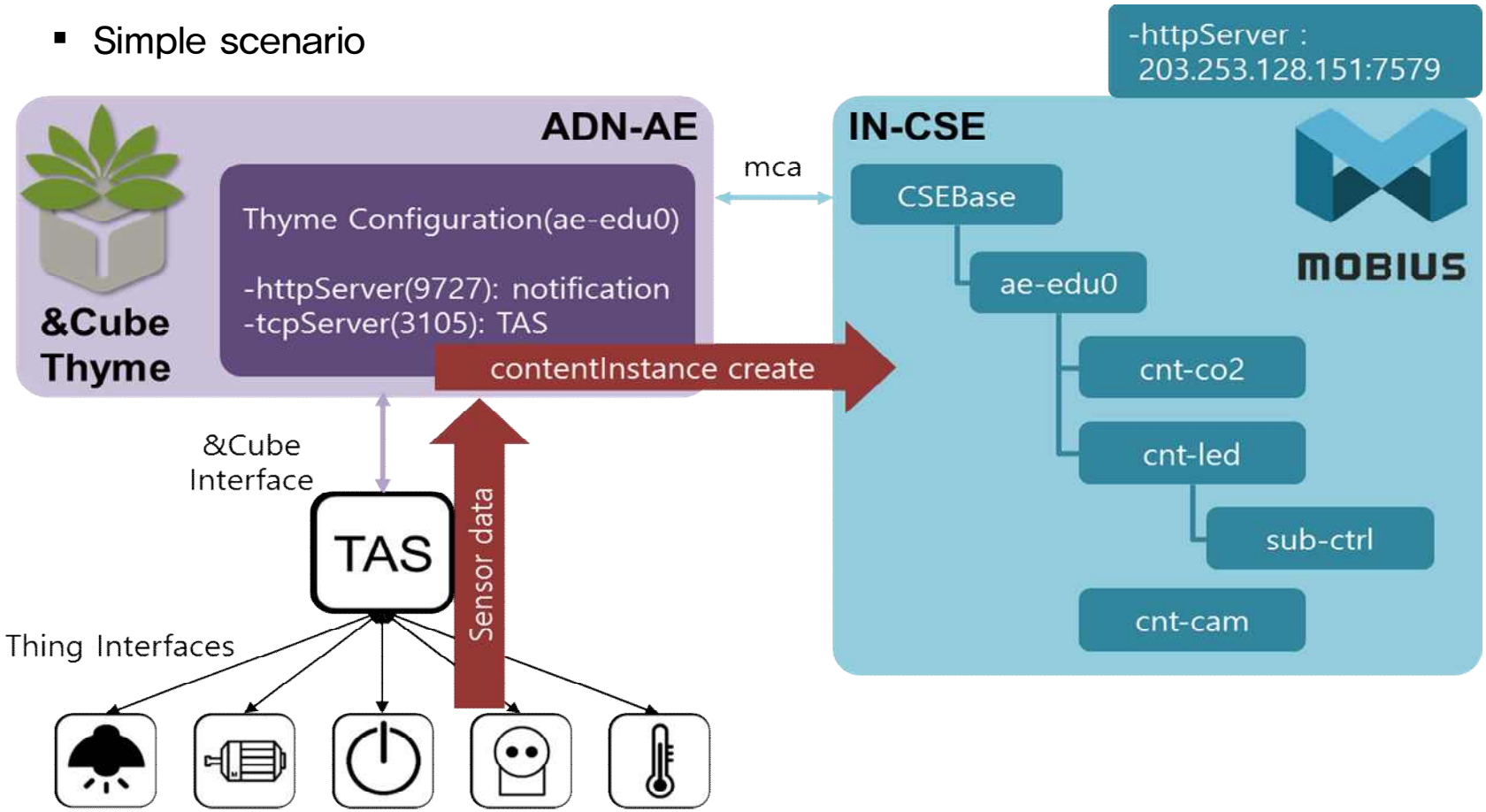
11.2 Run &Cube and TAS

- Run &Cube: Thyme
  - Simple scenario



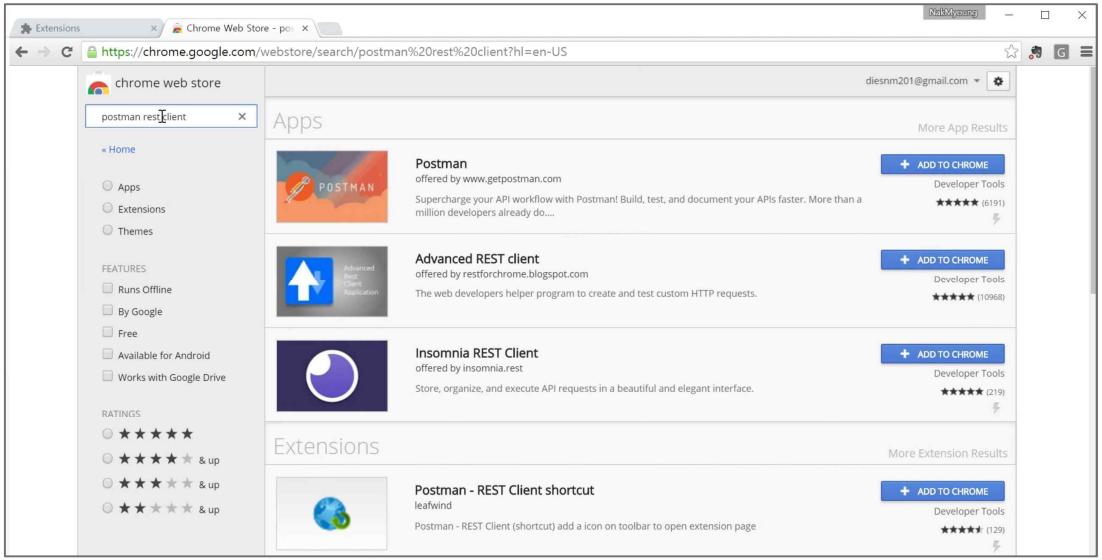
11.2 Run &Cube and TAS

- Run TAS
  - Simple scenario



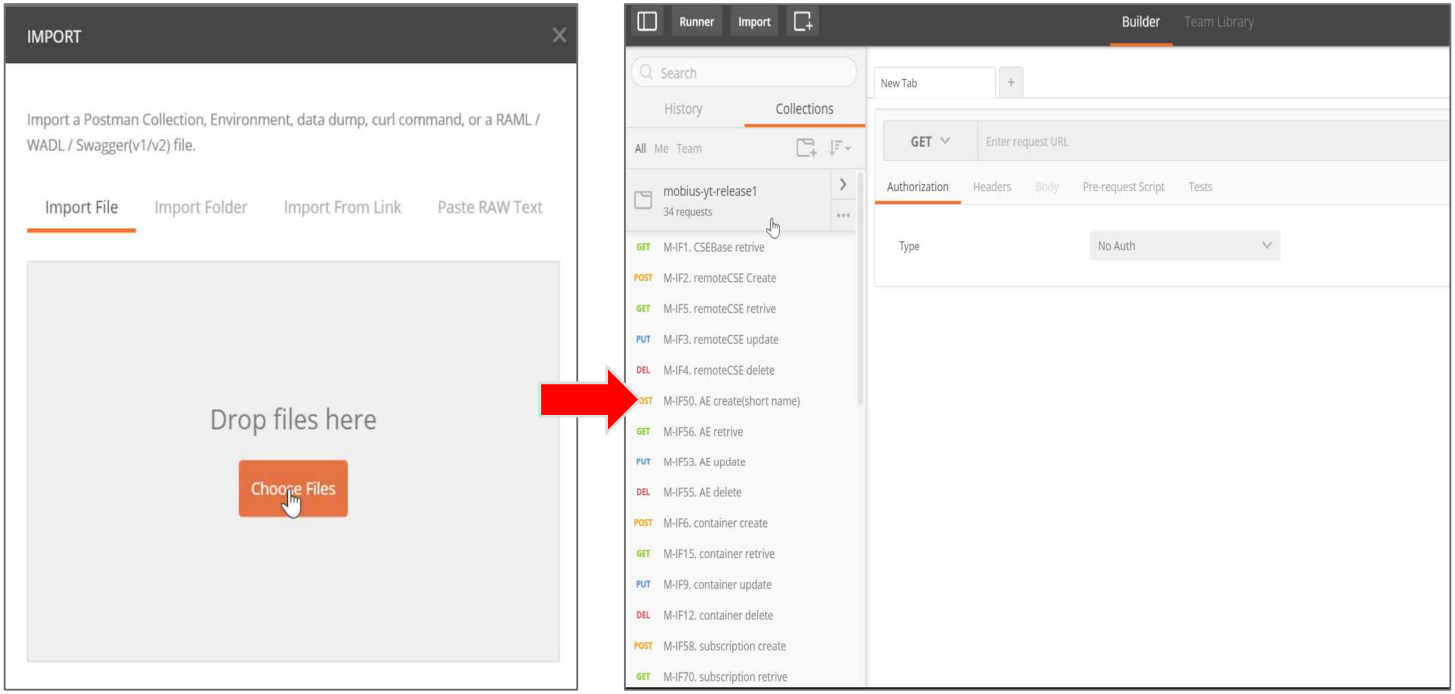
11.3 Test Device

- Postman installation
  - Download API Collection (Download → Mobius → Yellow Turtle)
    ex) mobius-yt-releaseXXX.json.postman_collection
  - Install and run Postman REST Client
    Google Chrome Browser Menu → More tools → Extensions → Get more extensions → Search 〈postman rest client〉 → ADD TO CHROME



11.3 Test Device

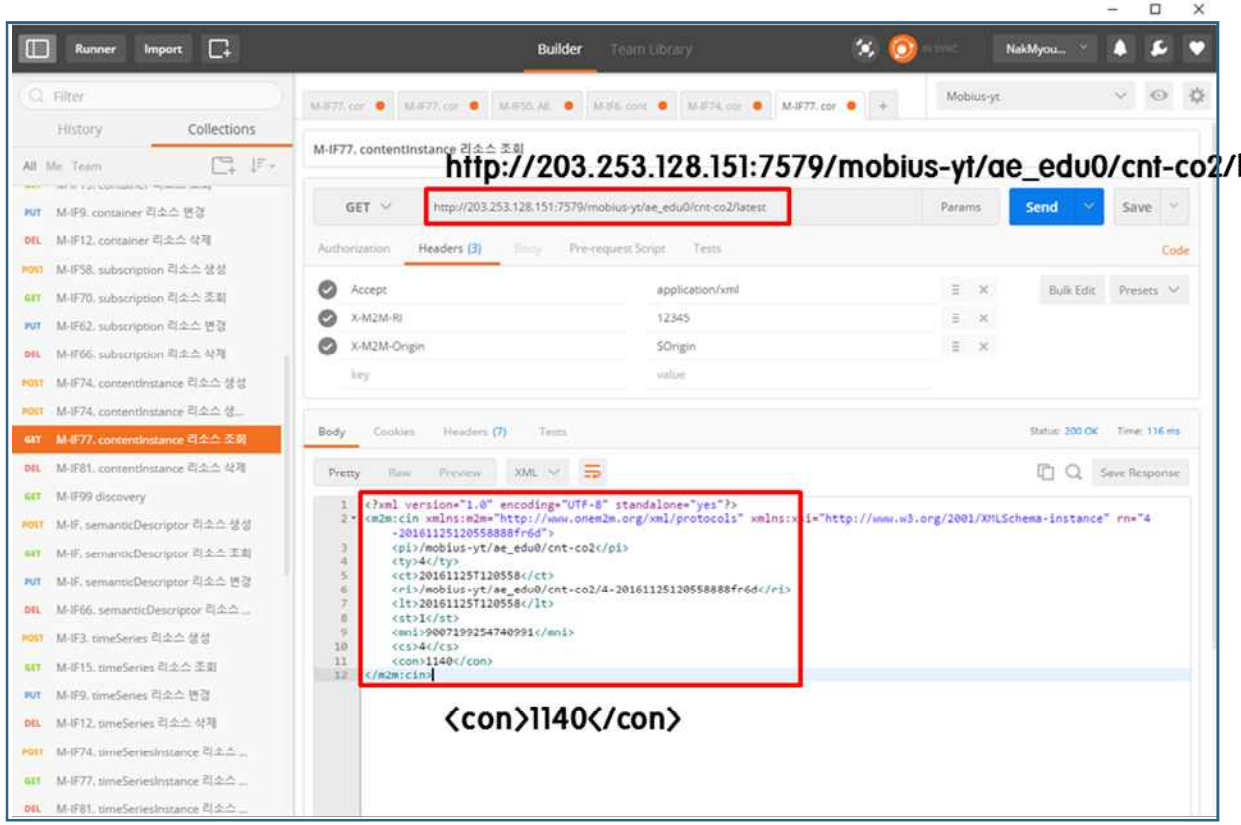
- Postman installation
  - Import Test API collection
    1. Collections → Import Collection
    2. Upload files → Choose 〈mobius-yt-releaseXXX.…〉 → Import



11.3 Test Device

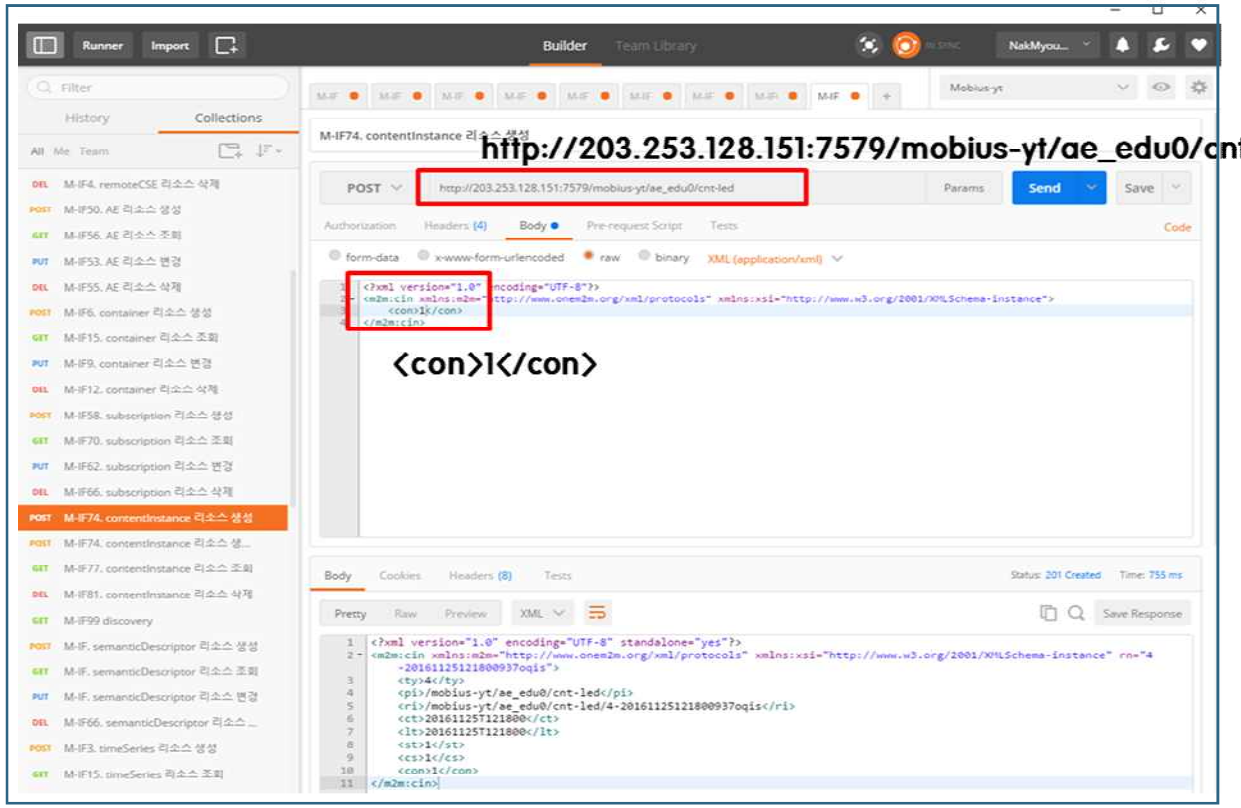
- Data retrieve
  - Data retrieval example using Mobius API

    Collections → ⟨mobius-yt-releaseXXX.…⟩ → contentInstance retrieve



11.3 Test Device

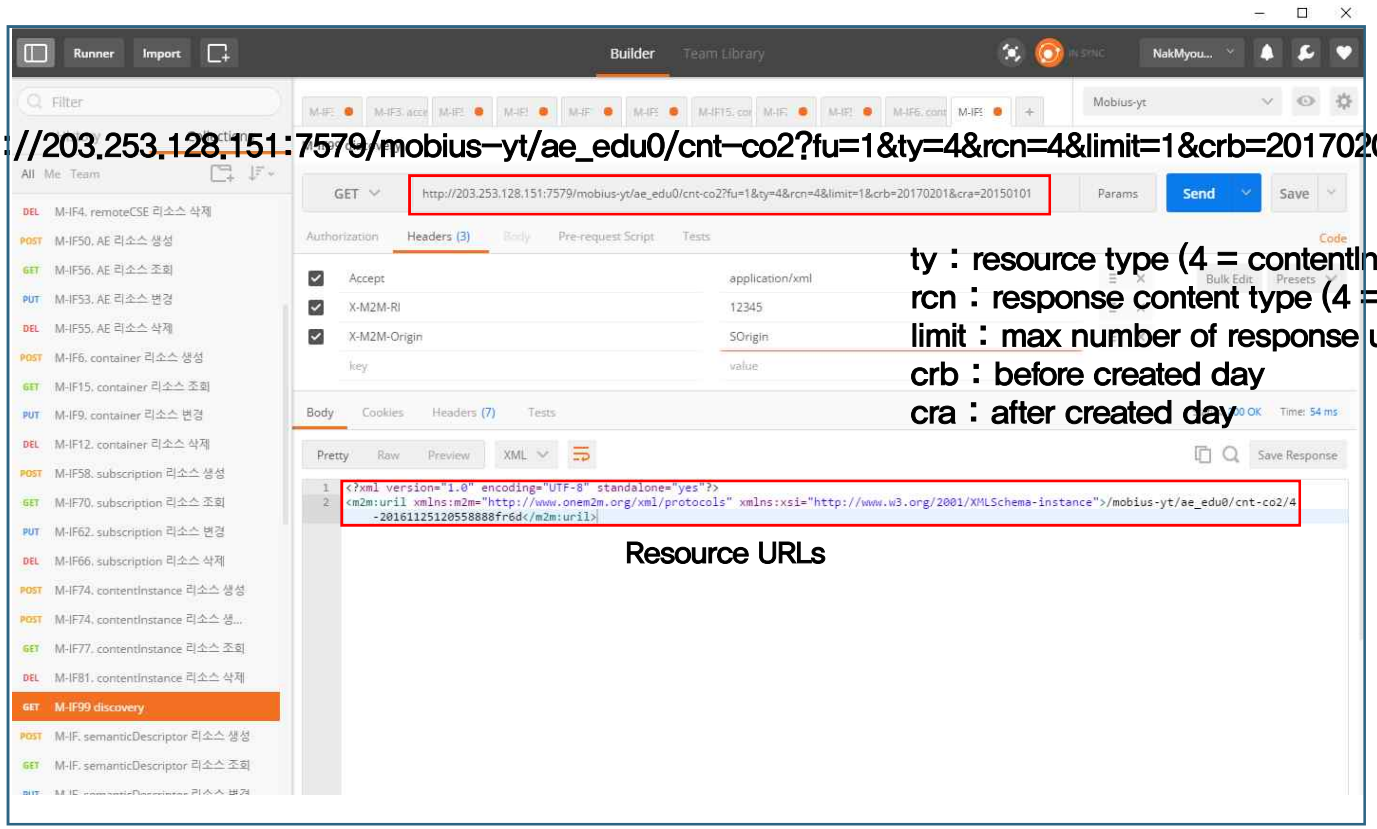
- Control request
  - Control request example using Mobius API
    Collections → ⟨mobius-yt-releaseXXX.…⟩ → contentInstance create



11.3 Test Device

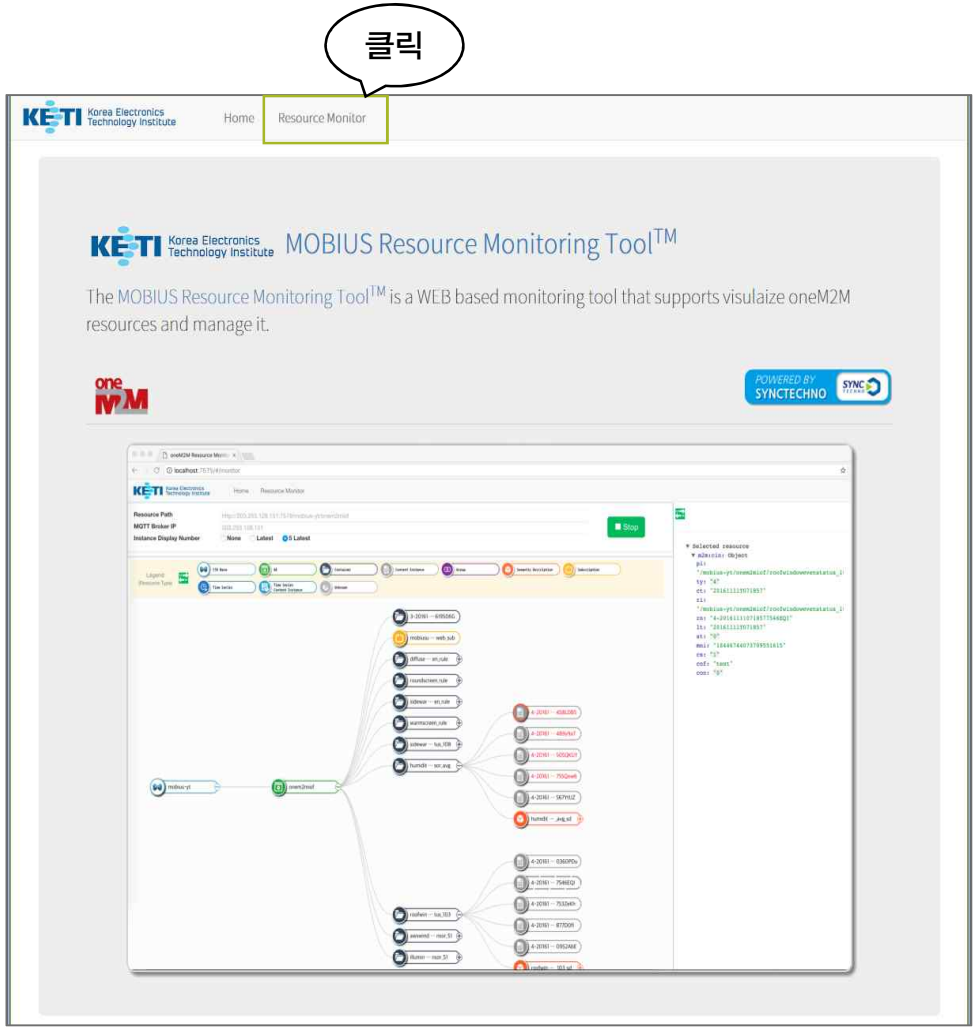
- Discovery
  - Discovery request example using Mobius API

    Collections → 〈mobius-yt-releaseXXX.…〉 → discovery



11.3 Test Device

- Mobius Resource Viewer
  - http://203.253.128.151:7575
  - 자신이 만든 AE의 리소스 상태를 볼 수 있는 웹 리소스 뷰어

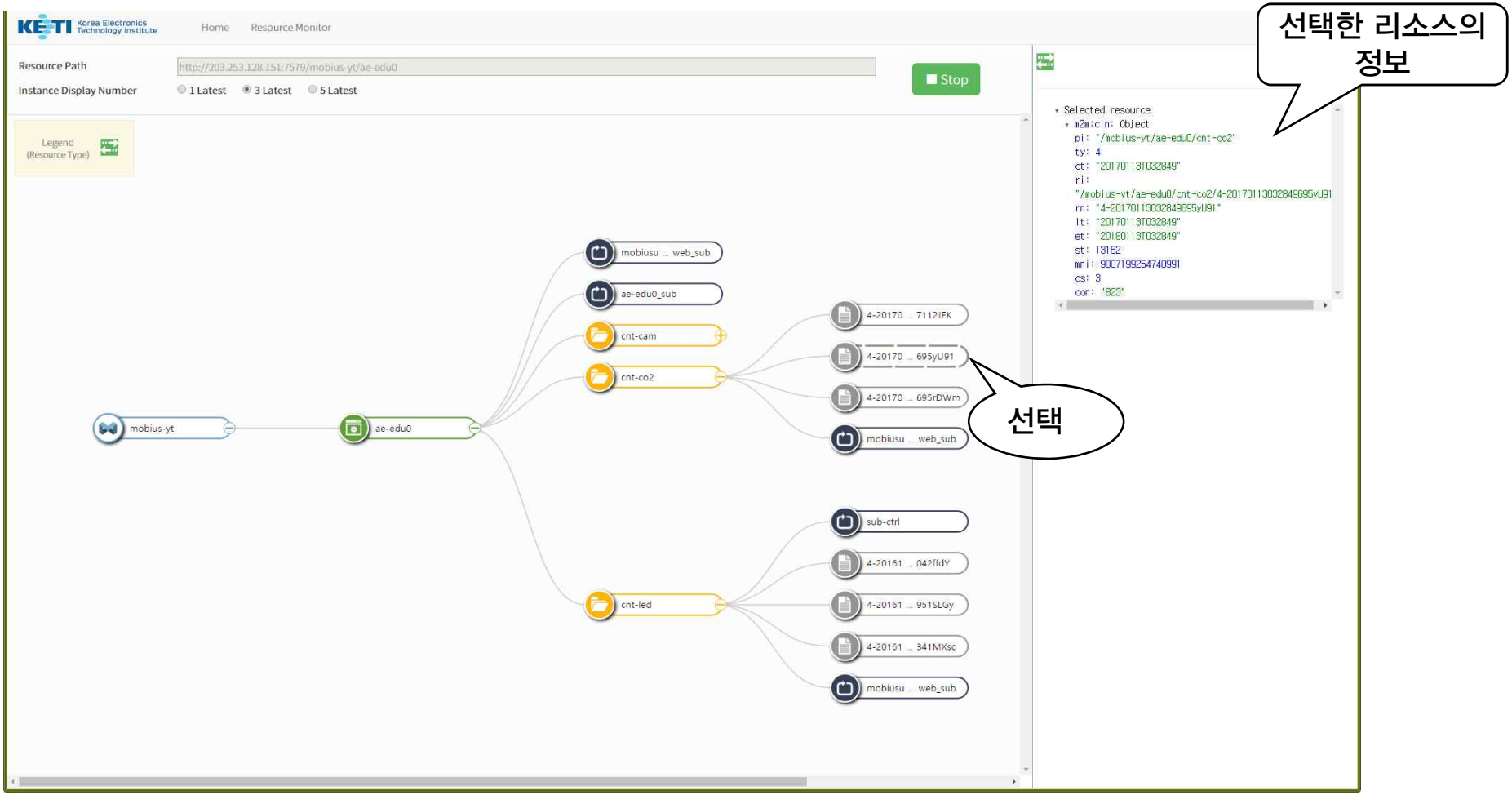


11.3 Test Device

- Mobius Resource Viewer

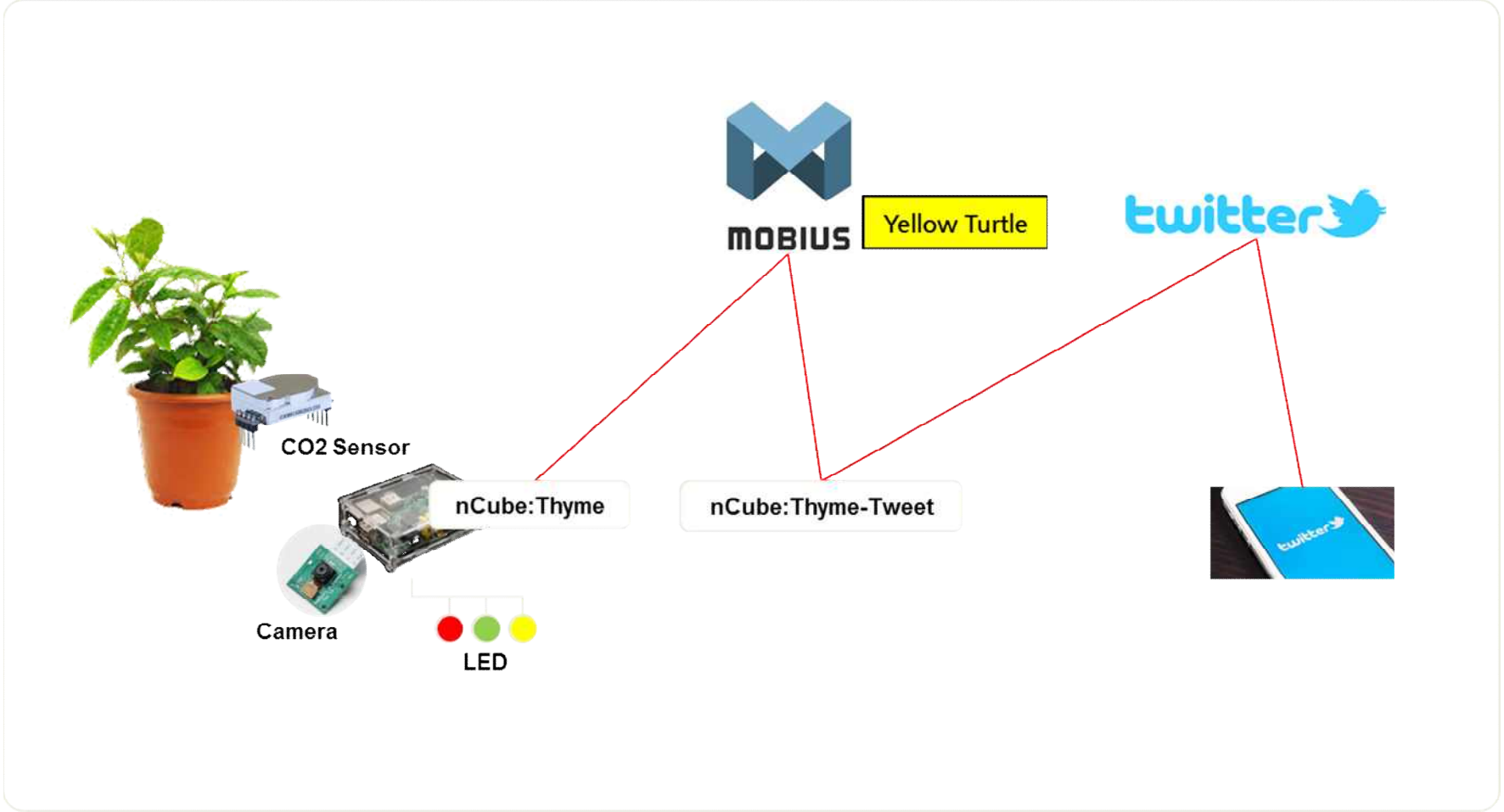
11.3 Test Device

- Mobius Resource Viewer



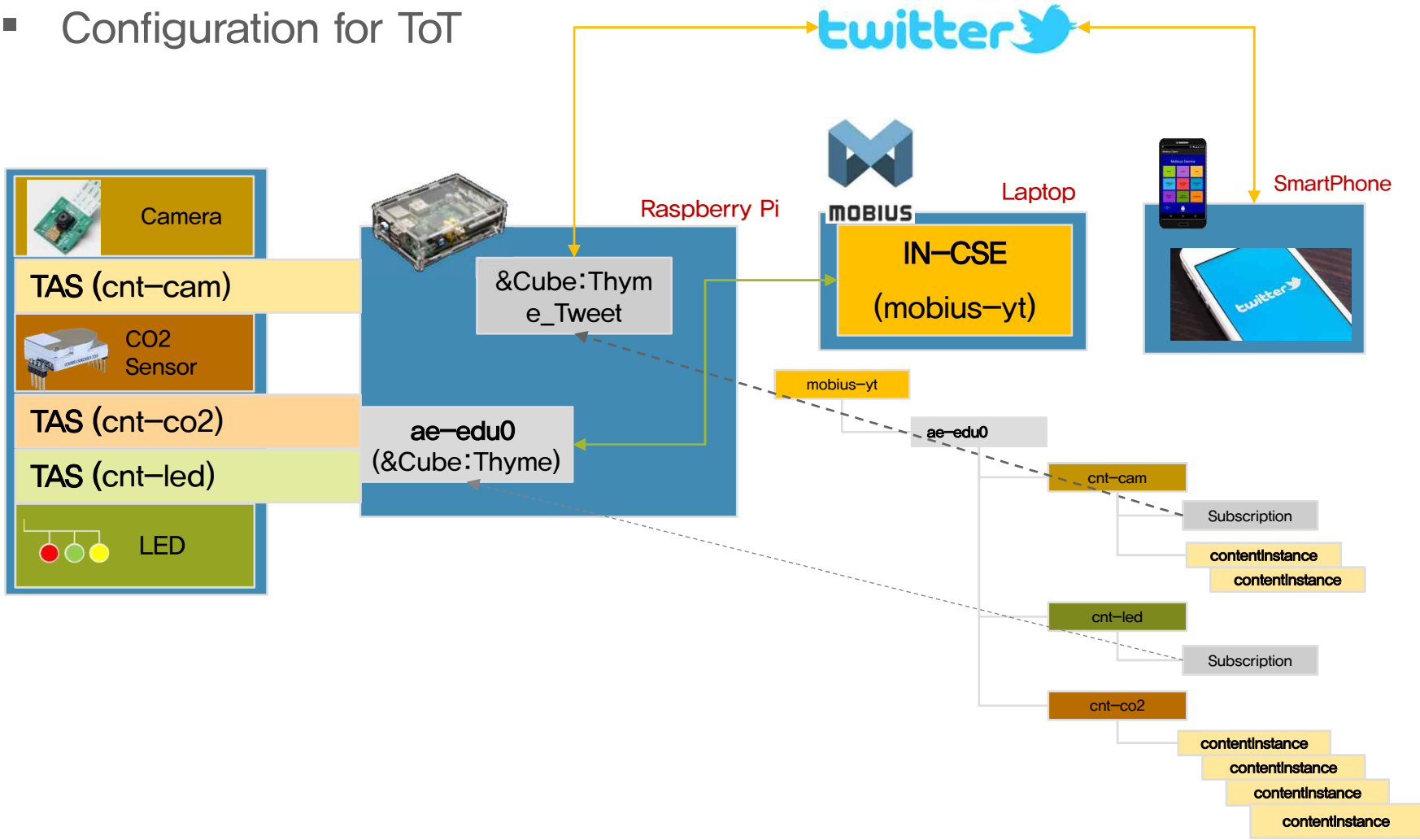
11.3 Test Device

- Case study for ToT (Tweet of Thing)



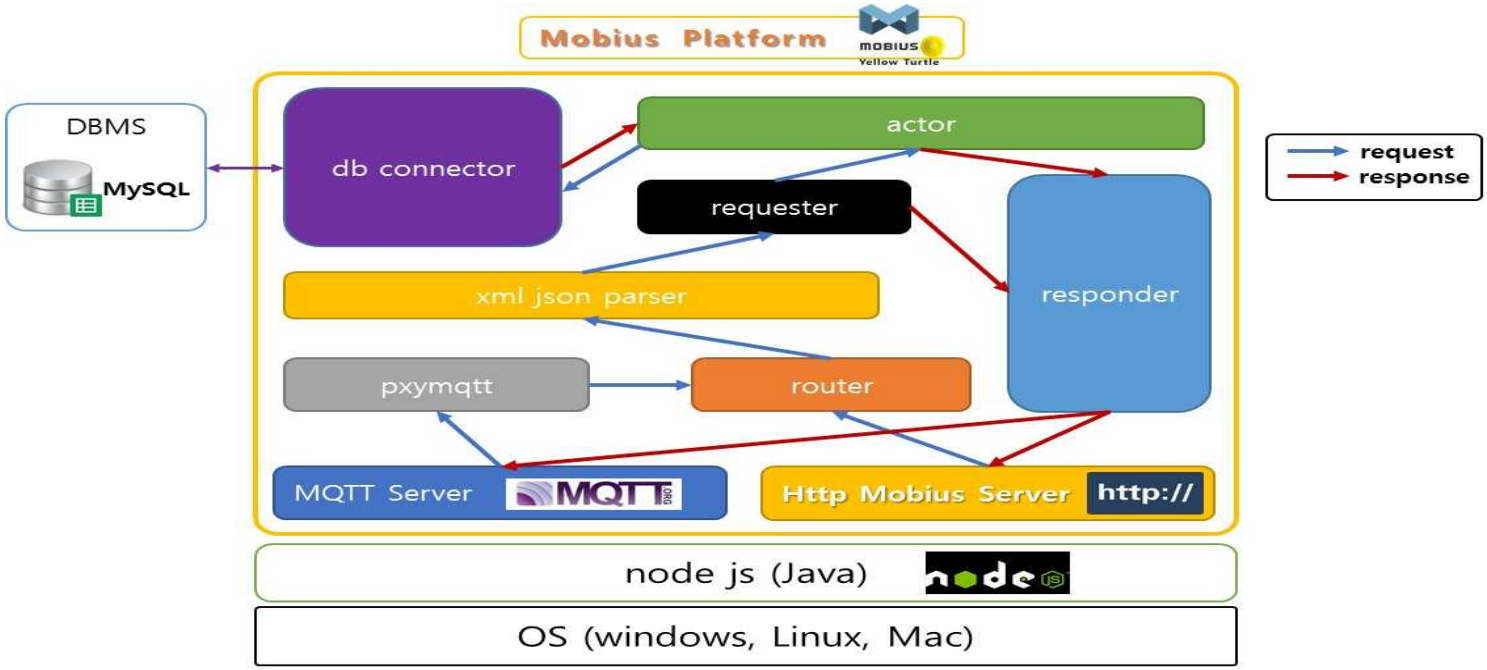
11.3 Test Device

- Configuration for ToT



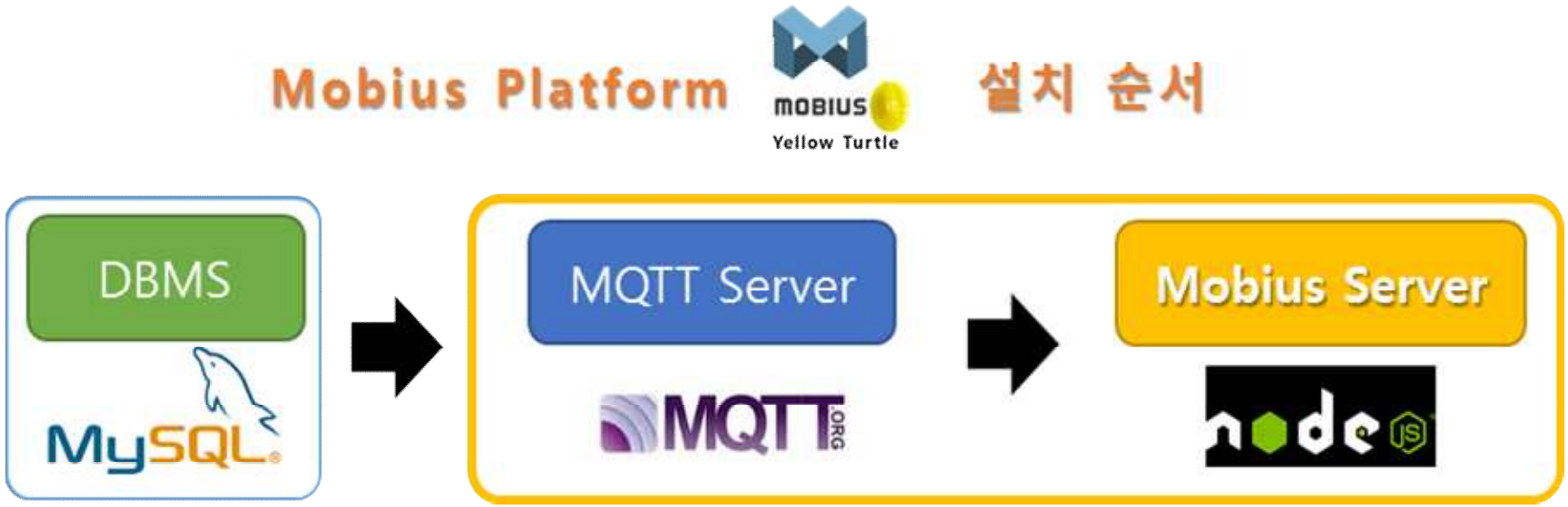
11.4 Mobius Yellow Turtle

- Mobius-yt(Yellow Turtle)
  - Mobius-yt is a IoT server platform based on Node.js
    - Using Java Script
    - Support MySQL Database
    - Support HTTP, MQTT network protocols



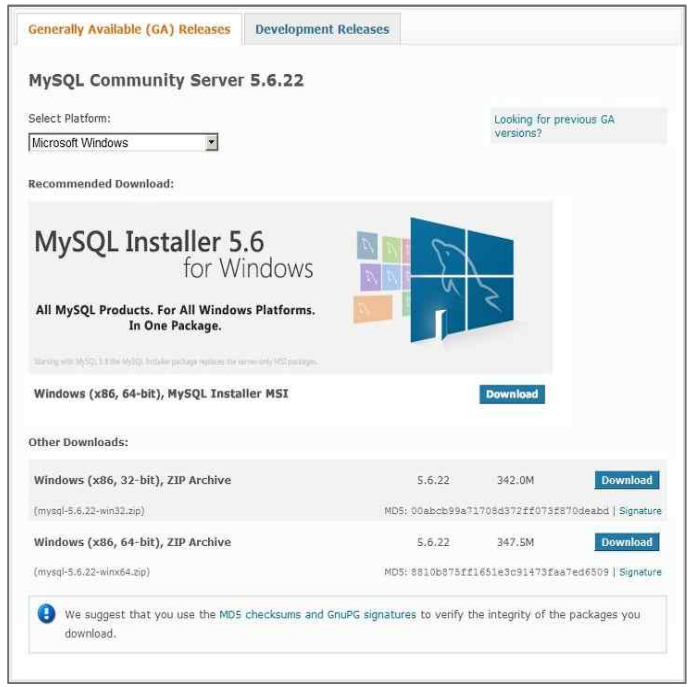
11.4 Mobius Yellow Turtle

- Construction
  - Reference install guide of Mobius-yt in OCEAN



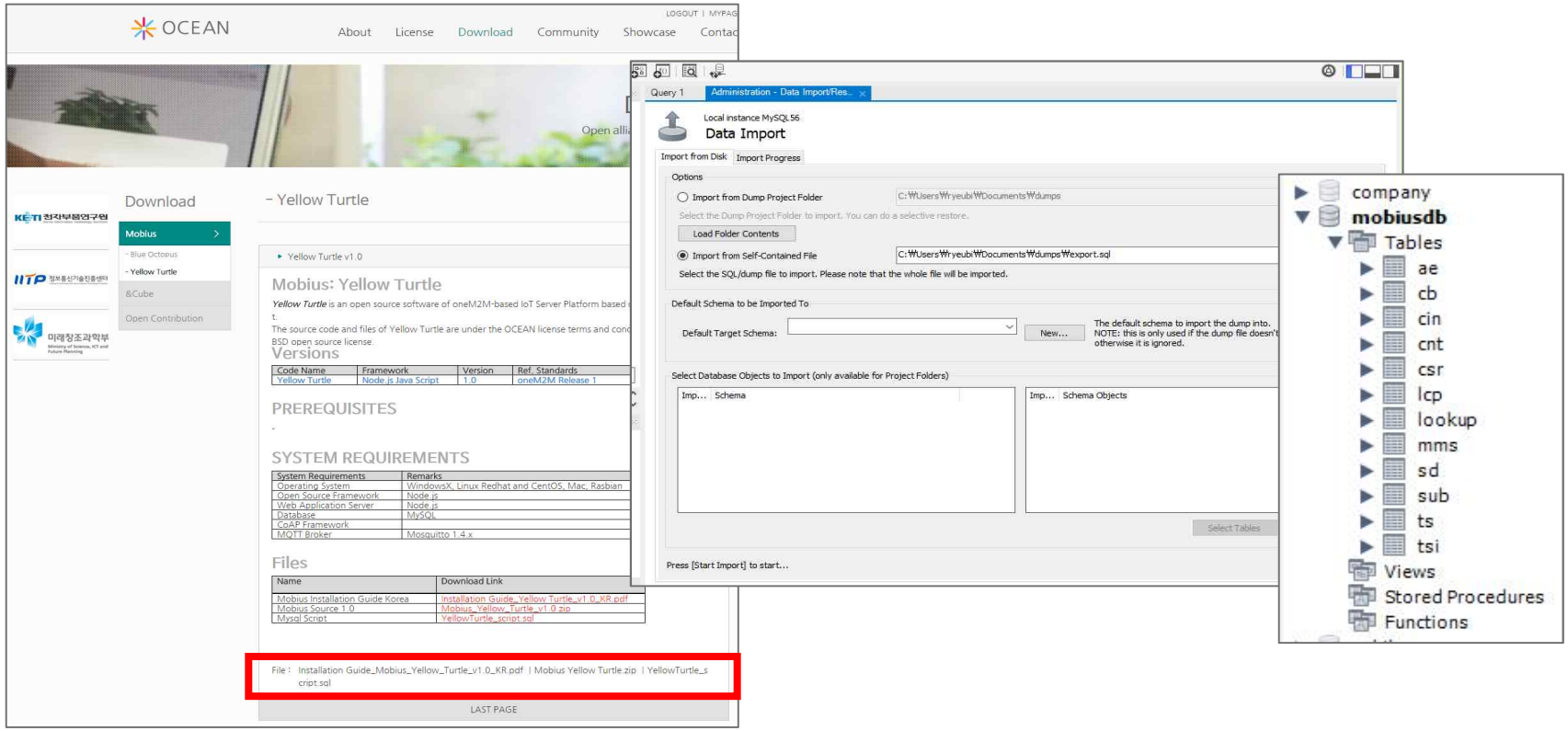
11.4 Mobius Yellow Turtle

- Install MySQL Database
  - Install MySQL Server and MySQL Workbench



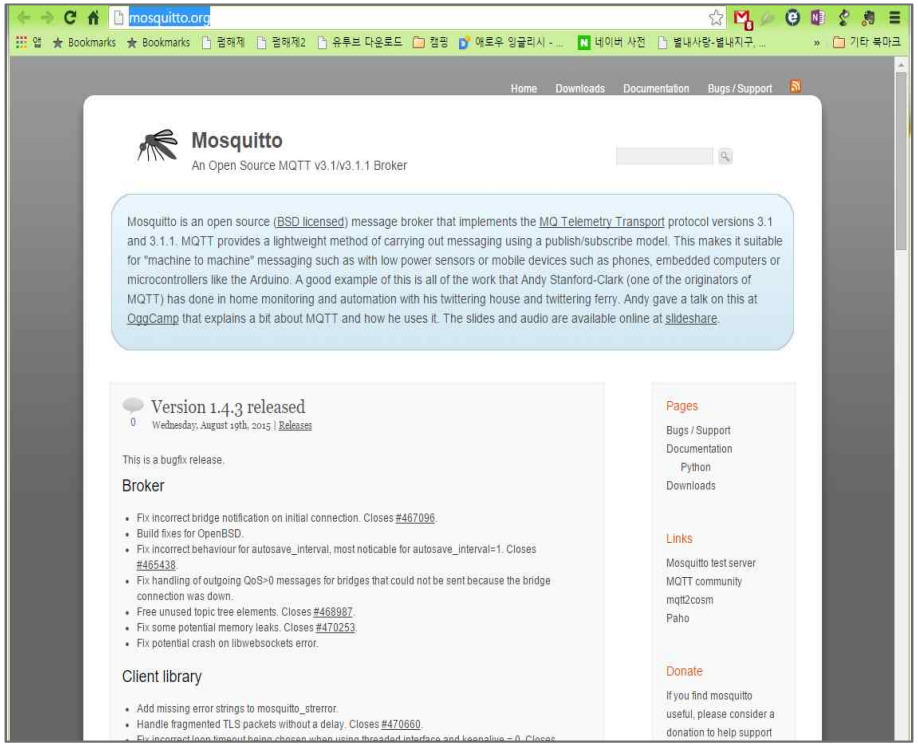
11.4 Mobius Yellow Turtle

- Install MySQL Database
  - Download MySQL Database Tables from OCEAN



11.4 Mobius Yellow Turtle

- Install MQTT Server
  - Mobius-yt supports MQTT protocol to communicate with IoT devices {ex)&Cube}
  - Download and install an open source MQTT broker Mosquitto http://mosquitto.org



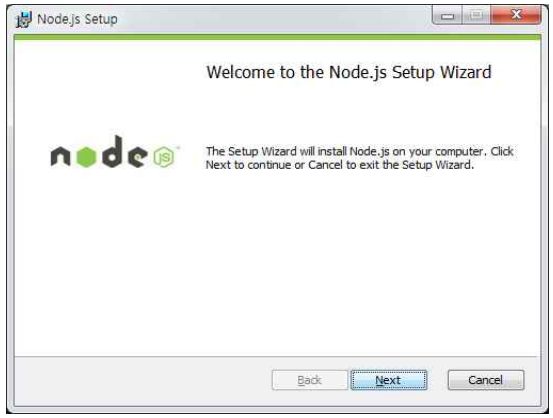
11.4 Mobius Yellow Turtle

- Node.js ?
  - Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine
  - Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient
  - Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world

  - Learn more about Node.js
    - https://nodejs.org/ko/about/
    - https://opentutorials.org/course/2136

11.4 Mobius Yellow Turtle

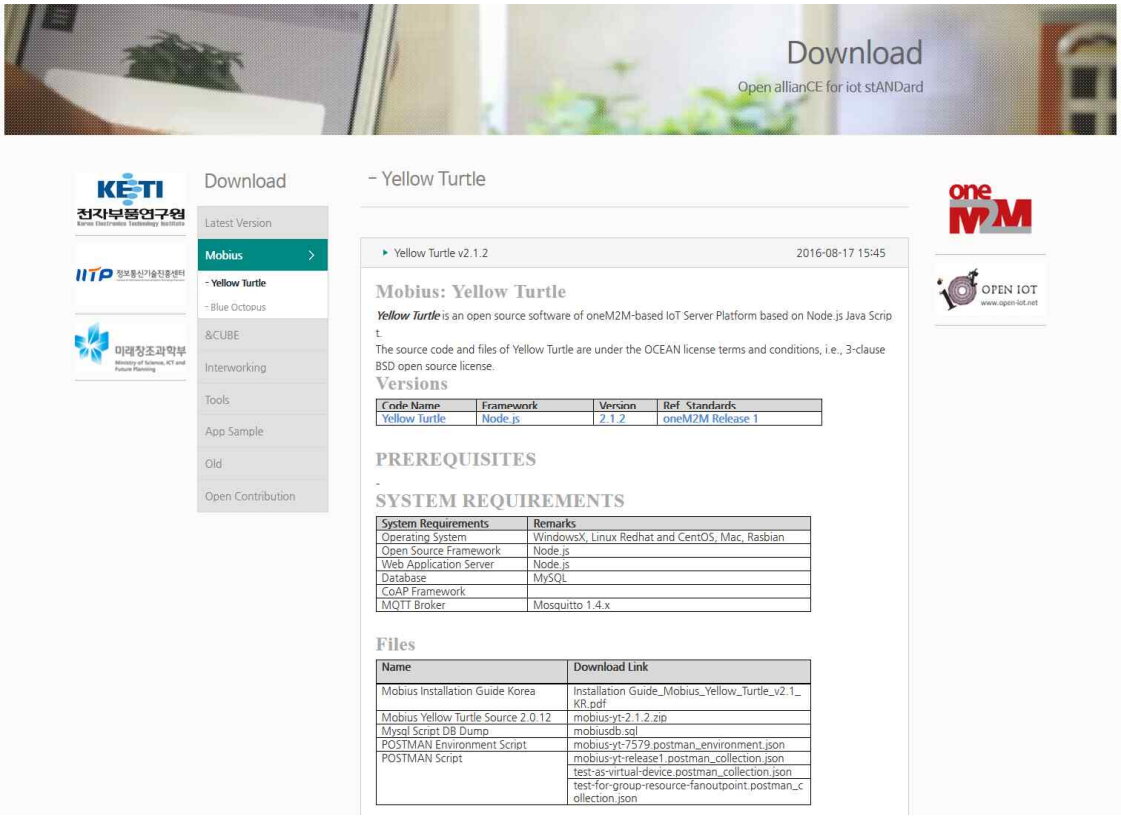
- Install Node.js
  - Download and install Node.js http://www.nodejs.org



  - Check installation state

11.4 Mobius Yellow Turtle

- Install Mobius-yt
  - Download Mobius-yt from OCEAN http://www.iotocean.org



11.5 Run Mobius

- Install Mobius-yt
  - Setting of configuration file (conf.json)
    - csebaseport : Mobius-yt open port
    - dbpass : MySQL Database access password

```
{
      “csebaseport": "7579",
     "dbpass":   “keti123"
}
```

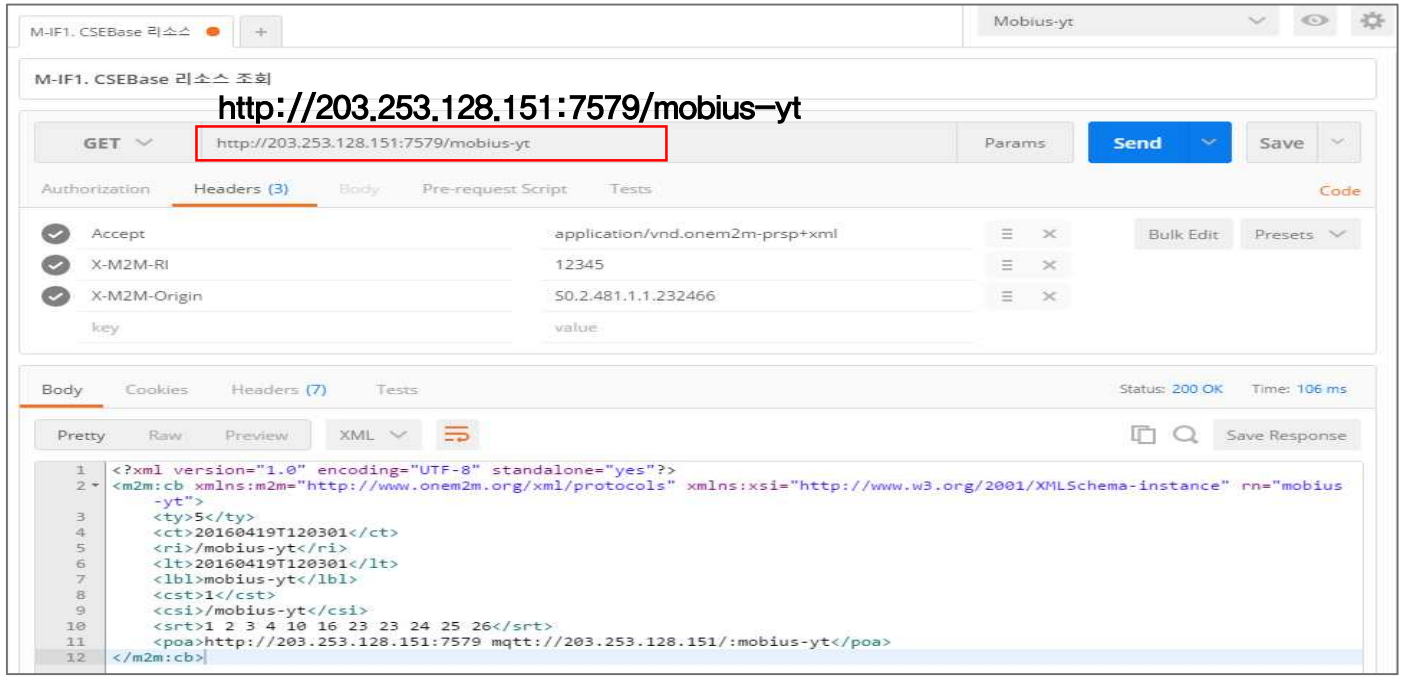
  - Run Mobius-yt
    - run 〈node mobius.js〉 in command prompt

11.5 Run Mobius

- Mobius-yt CSEBase retrieve
  - CSEBase retrieval example using Mobius API

    Run POSTMAN

    Collections → 〈mobius-yt-releaseXXX.…〉 → CSEBase retrieve



Assignment

- Assignment 11

실습 예제를 활용하여 자신의 oneM2M 리소스를 만들어
모비우스 리소스 뷰어로 **볼 수 있도록 개발해보시오.**

Reference

[11.1] Install &Cube and TAS

TAS (Thing Adaptation Software) Flow

그림 출처: http://www.iotocean.org/main/

[11.3] Test Device

Mobius Resource Viewer

그림 출처: http://www.iotocean.org/main/

12장. IoT 서비스 개발

Chapter 12. Development of IoT Service

12.1 Run &Cube and TAS for Local Mobius
12.2 Extension
12.3 Develop Environment
12.4 Practice

- Assignment
- Reference

강의 개요

▪ 강의 목표

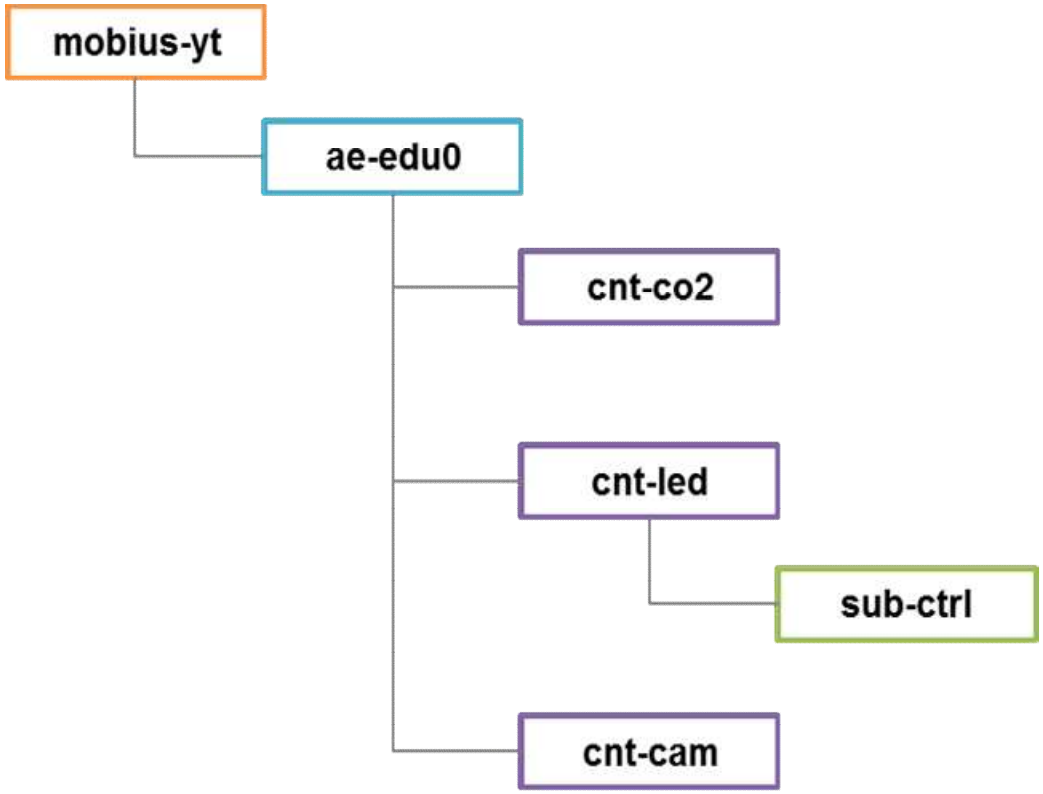
디바이스와 로컬 Mobius 플랫폼과의 연동을 이해하고
디바이스를 컨트롤 할 수 있는 서비스 어플리케이션 개발 방법을 학습한다.

▪ 강의 내용

- 로컬 모비우스 연동 &cube 및 TAS 설치 실습
- 안드로이드 서비스 어플리케이션 개발
- 사물인터넷의 활용 사례 학습

12.1 Run &Cube and TAS for Local Mobius

- &Cube configuration
  - Open configuration file and edit

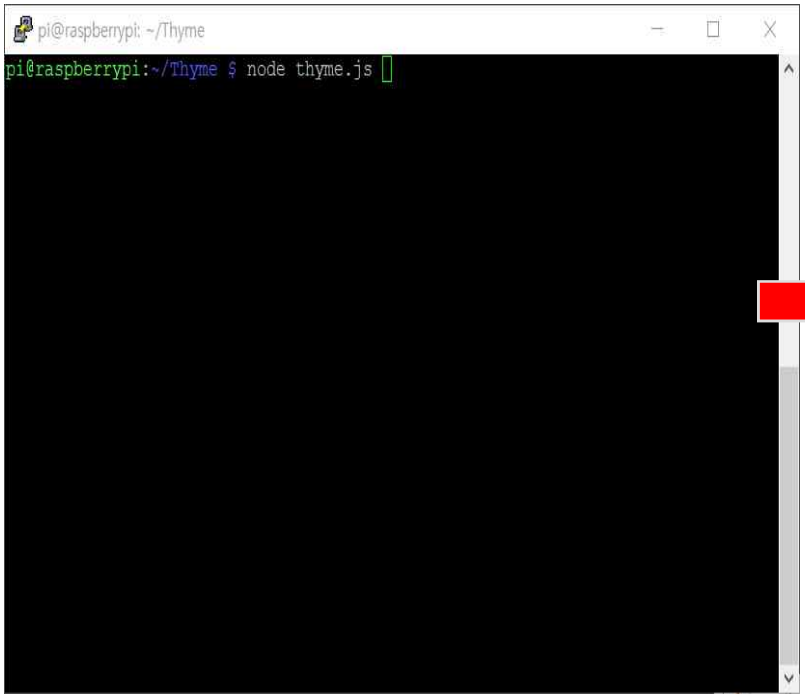


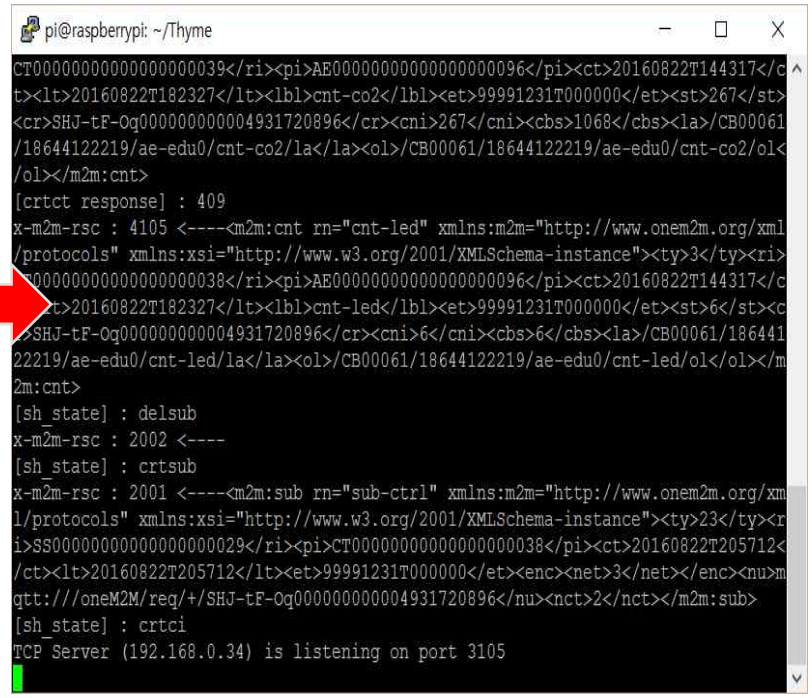

```
{
    "useprotocol": "http",
    "cse": {
        "cbhost":   "Local Mobius IP Address",
        "cbport": "7579",
        "cbname": "mobius-yt",
        "cbcseid": "/mobius-yt"
    },
    "ae": {
        "aeid": "S",
        "appid": "0.2.4812.12.1",
        "appname":   "ae-edu0",
        "appport": "9727",
        "bodytype": "json",
        "tasport": "3105"
    },
    "cnt": [
        {
            "parentpath": "/ae-edu0",
            "ctname":   "cnt-co2"
        },
        {
            "parentpath": "/ae-edu0",
            "ctname":   "cnt-led"
        },
        {
            "parentpath": "/ae-edu0",
            "ctname":   "cnt-cam"
        }
    ],
    "sub": [
        {
            "parentpath": "/ae-edu0/cnt-led",
            "subname": "sub-ctrl",
            "nu": "mqtt://autoset"
        }
    ]
}
```

12.1 Run &Cube and TAS for Local Mobius

- 라이브러리 설치 성공 화면
  - Input &Cube: Thyme run command

```
pi@raspberrypi ~/node/thyme $ node thyme.js
```

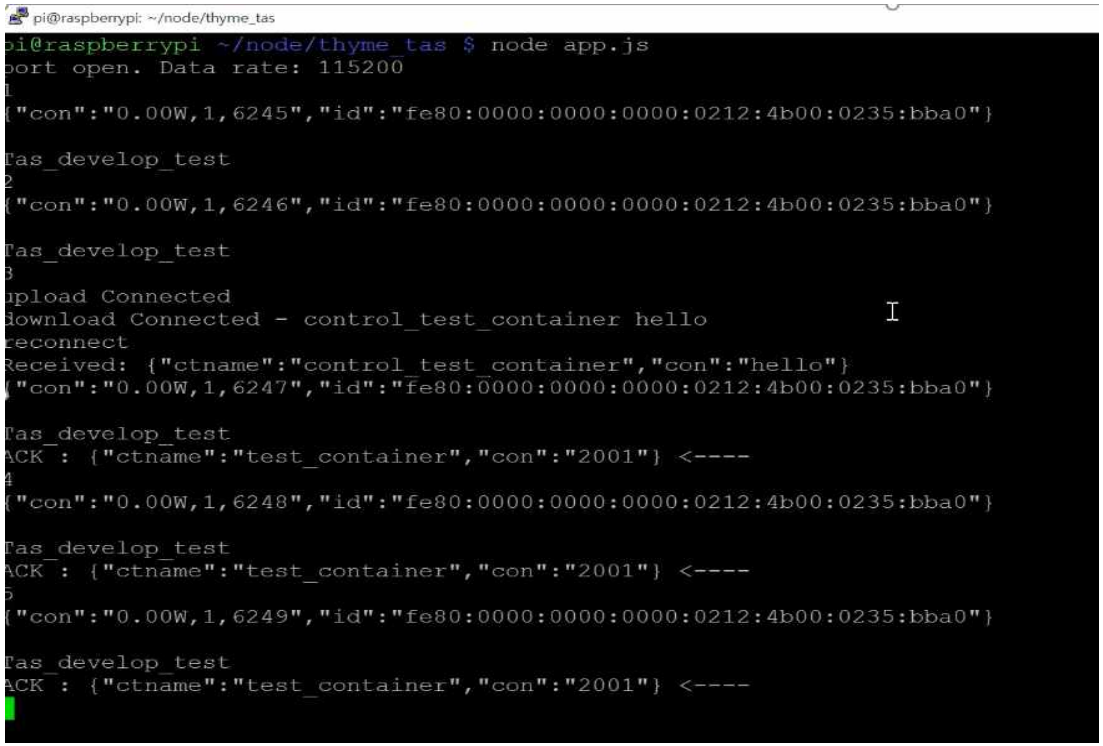




12.1 Run &Cube and TAS for Local Mobius

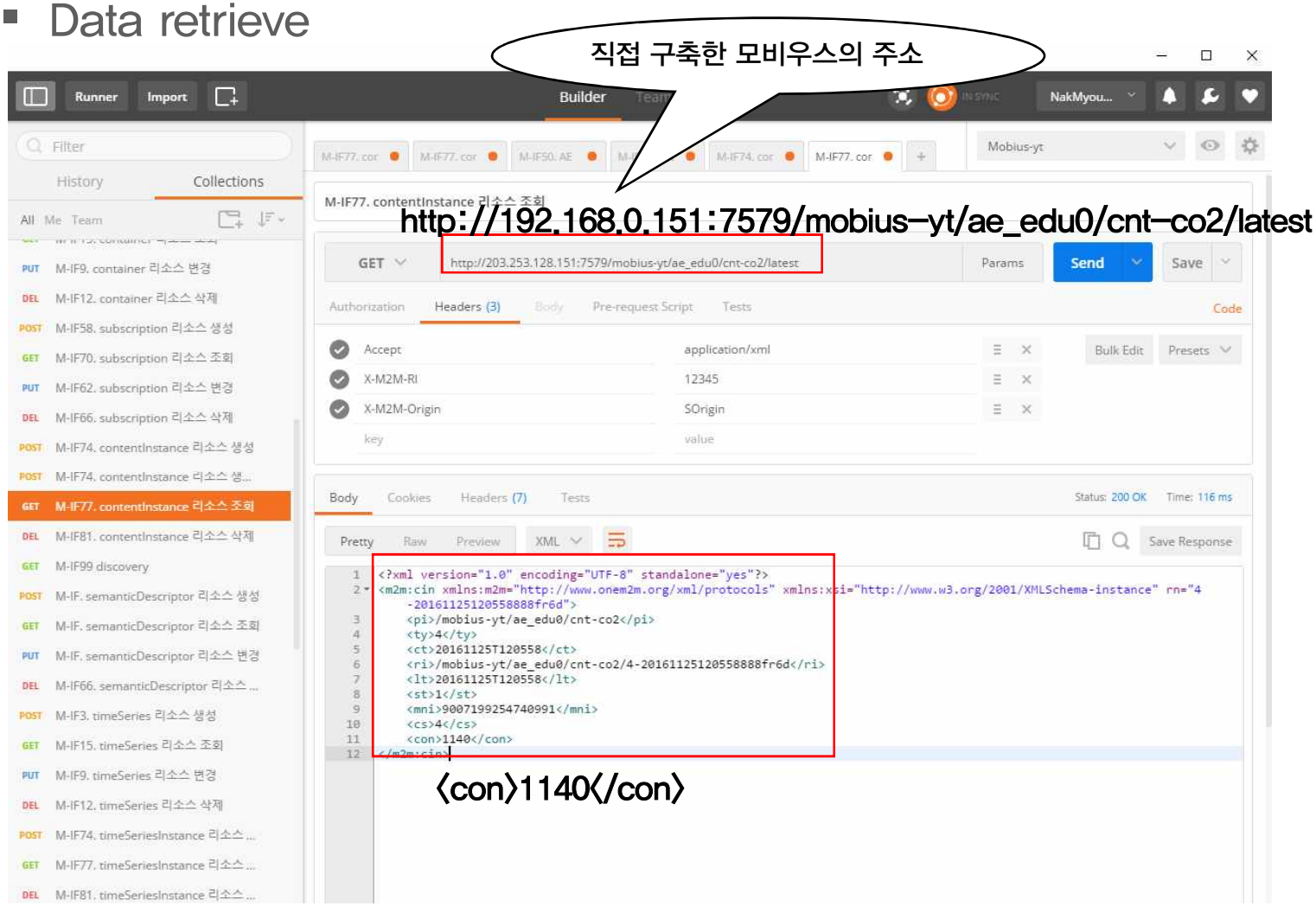
- Run TAS
  - Input TAS run command

```
pi@raspberrypi ~/node/tas_co2 $ node app.js
```



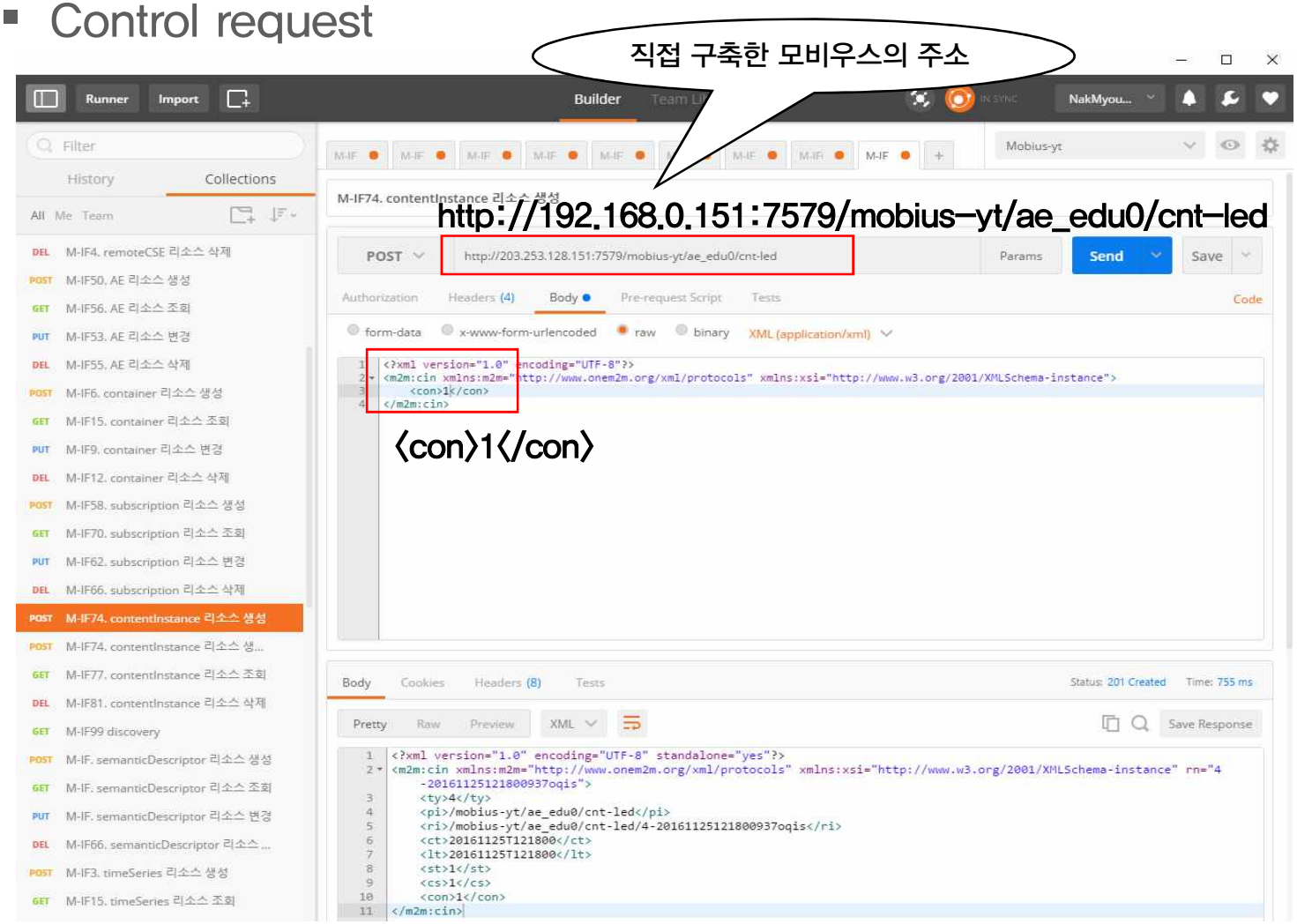
12.1 Run &Cube and TAS for Local Mobius

- Data retrieve



12.1 Run &Cube and TAS for Local Mobius

- Control request



12.2 Extension

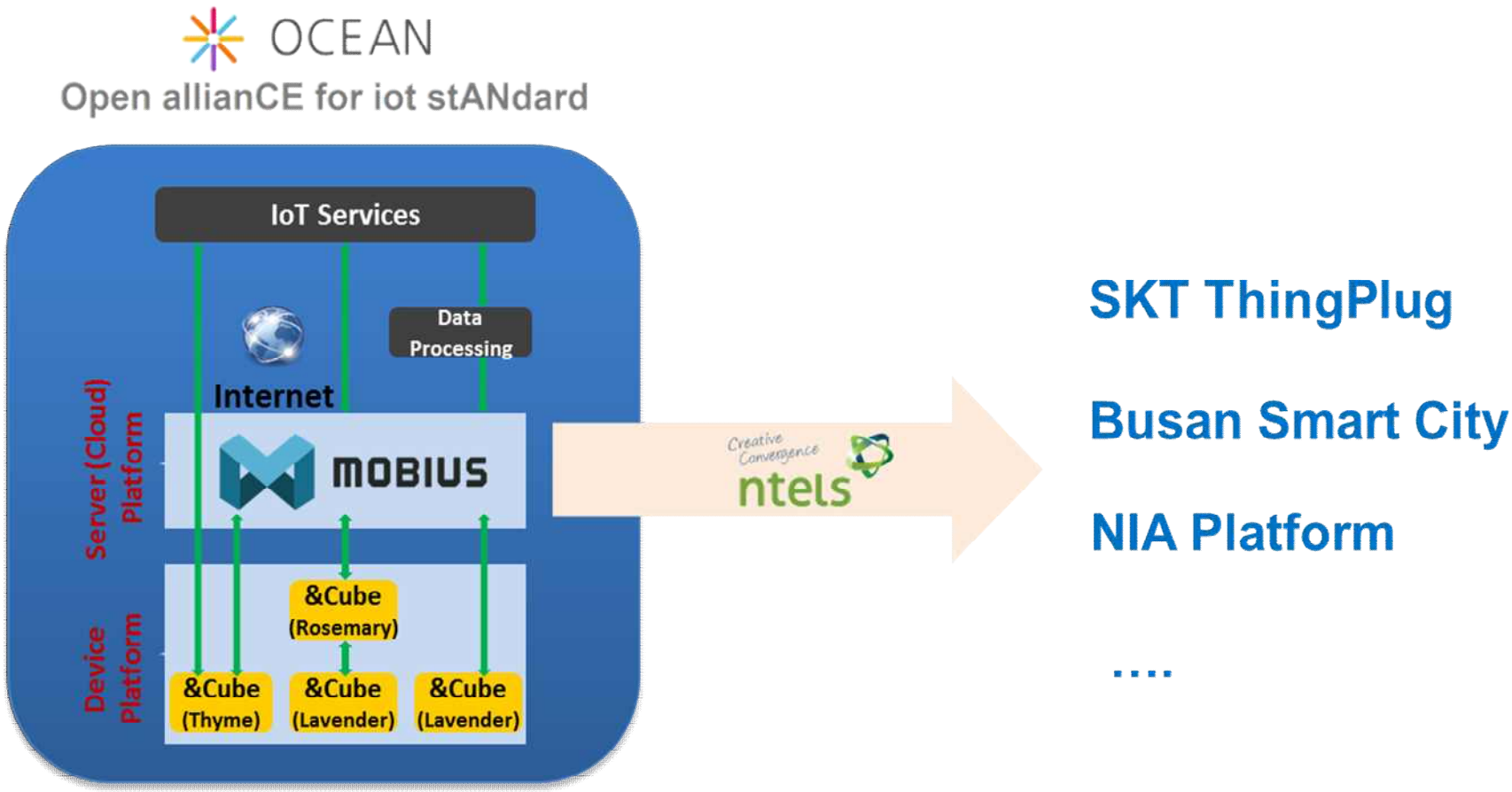
- SK ThingPlug (Jun. 2015)

12.2 Extension

- Busan Smart City

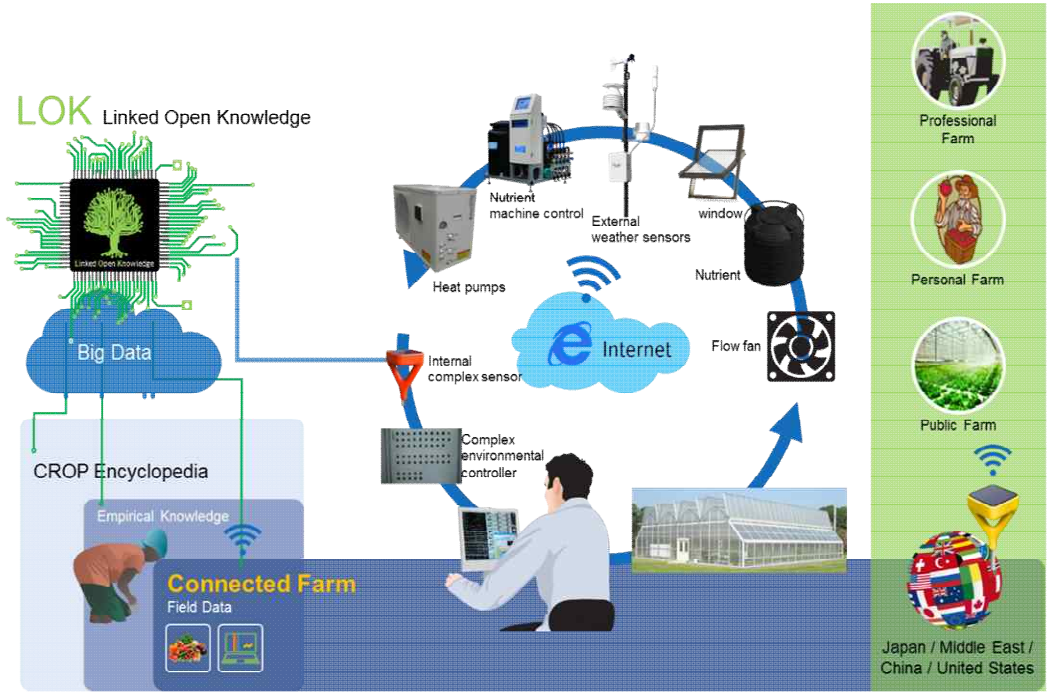
12.2 Extension

- nTels (2015 ~)



12.2 Extension

- Seedream, Maxfor
  - IoT Smart Farm

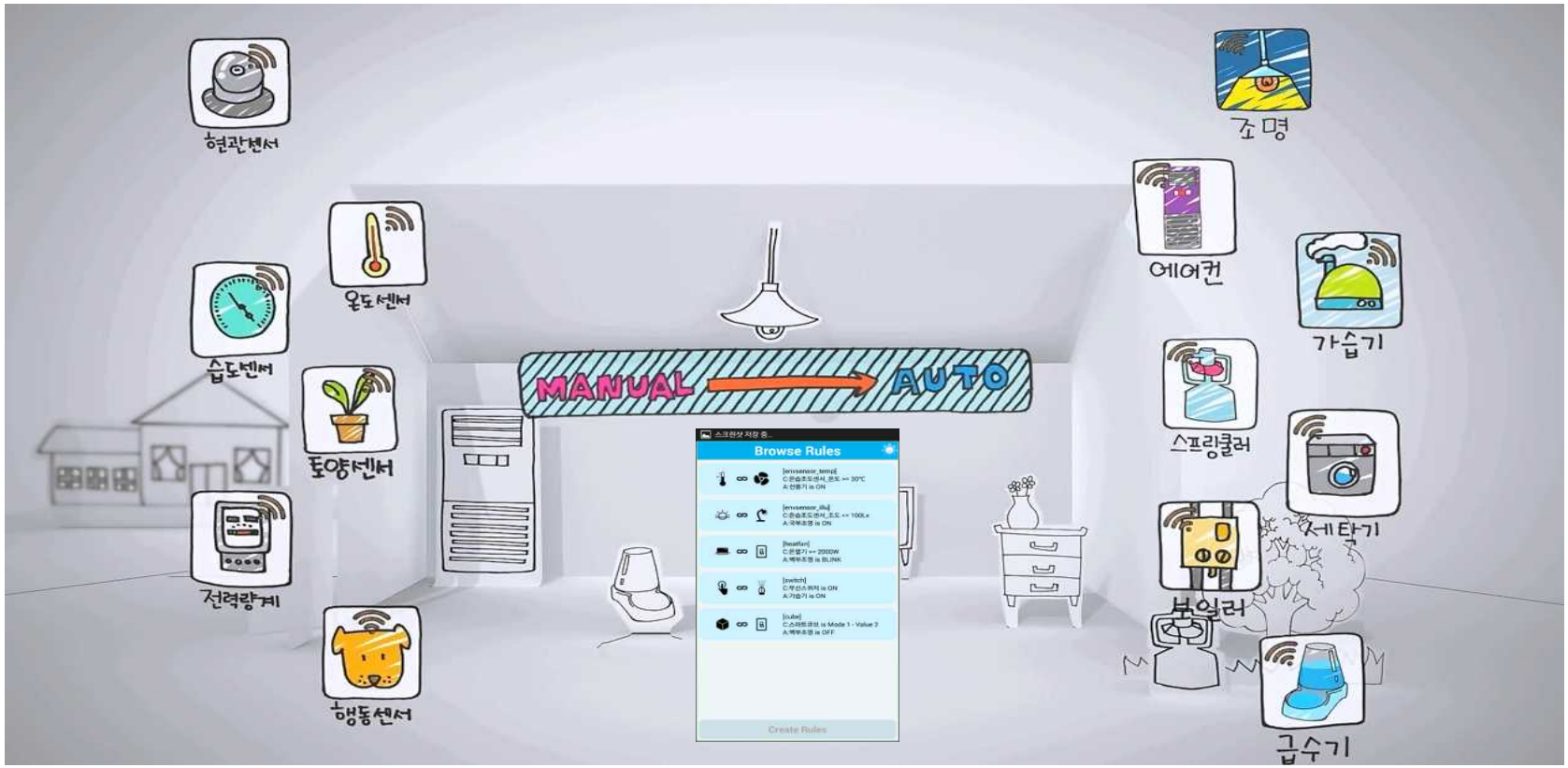


12.2 Extension

- iThing service

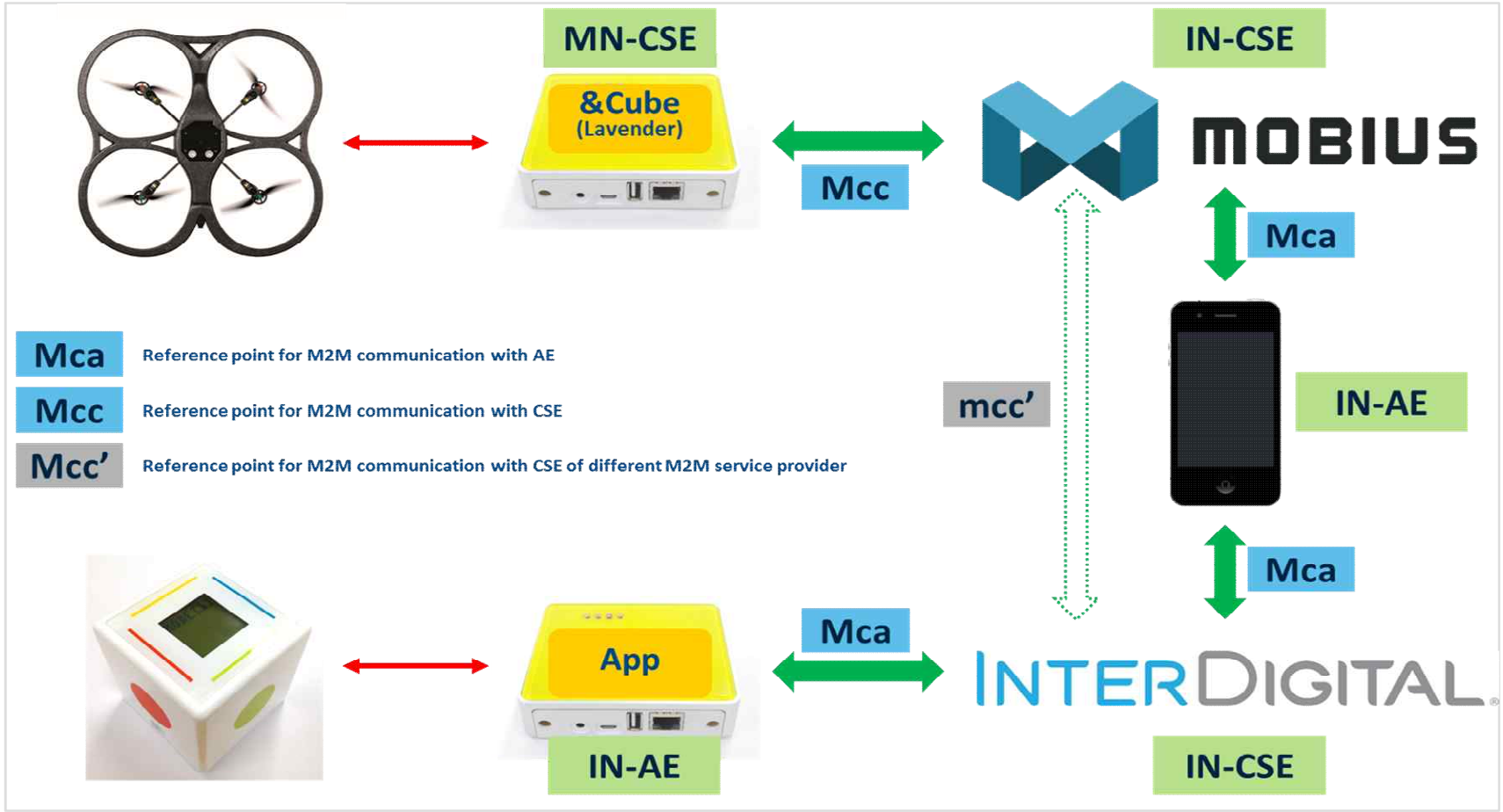
12.2 Extension

- TTEO service



12.2 Extension

- Global Interworking (Drone)



12.2 Extension

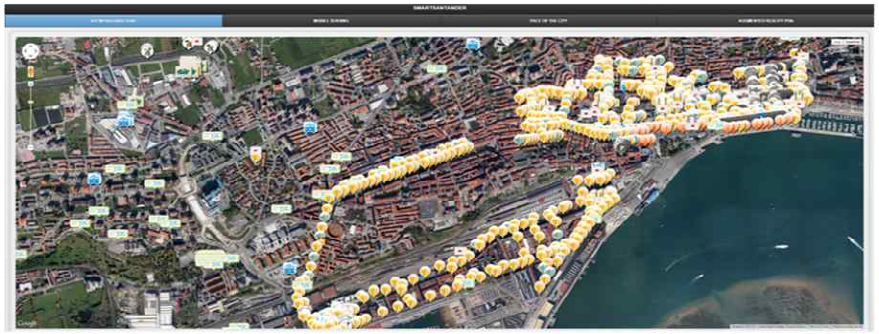
- Global Interworking

A smart city service

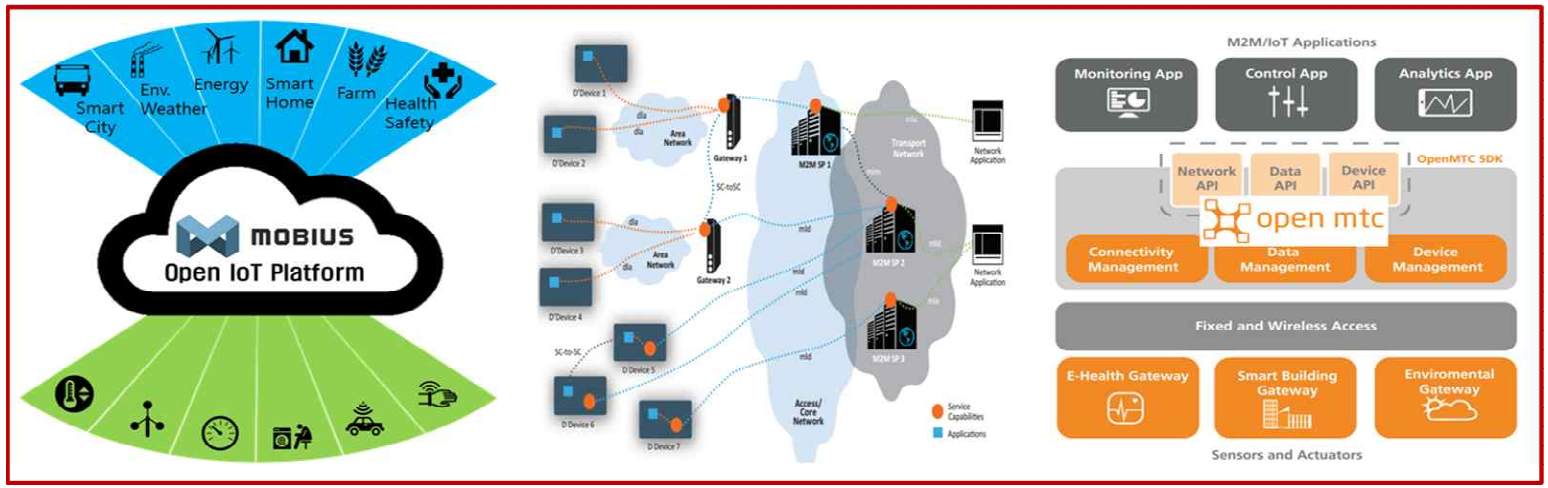
+

3 IoT platforms

NEC Laboratories Europe

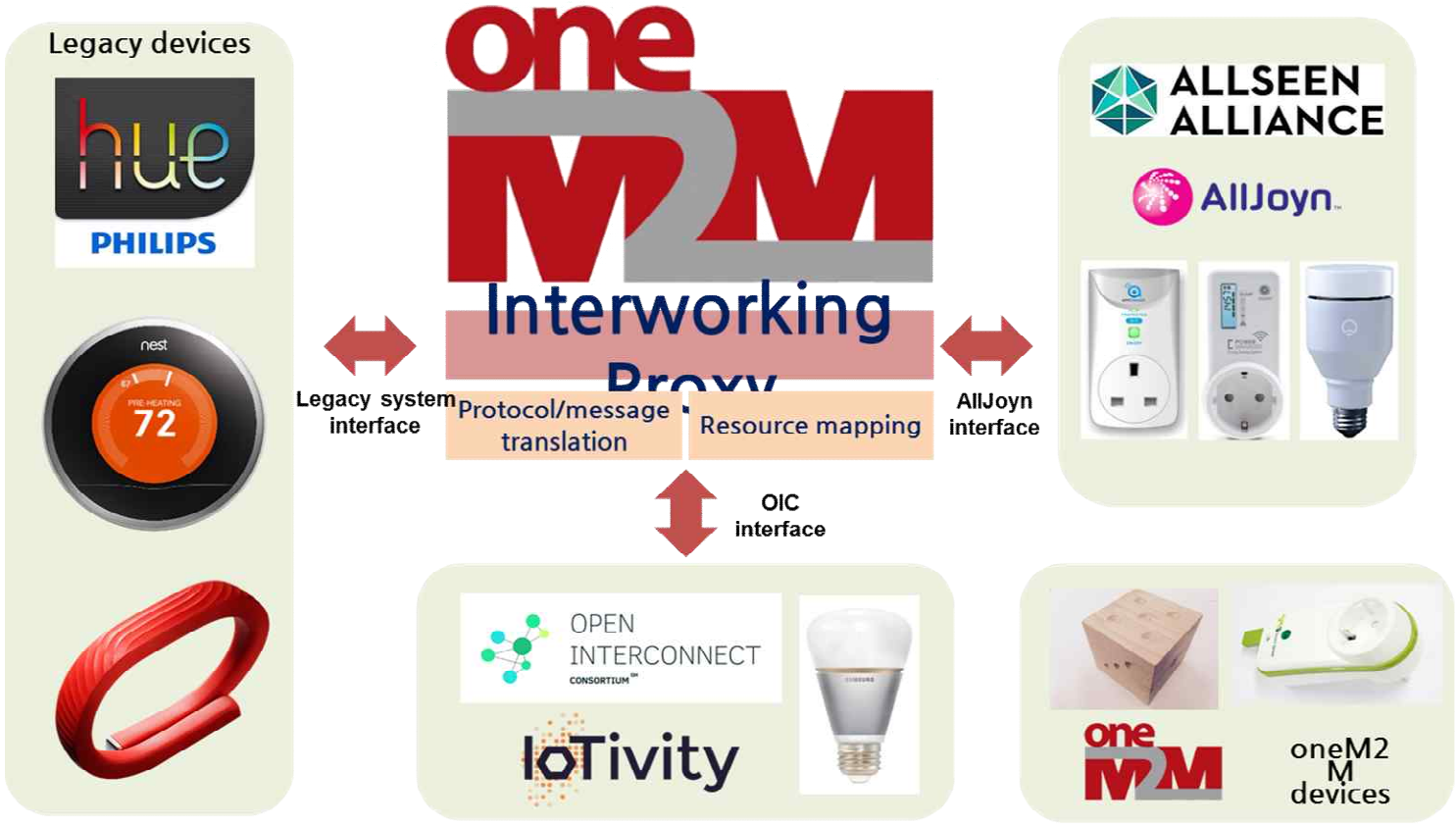


KETI/SKT | **Convida Wireless** | **Fraunhofer**



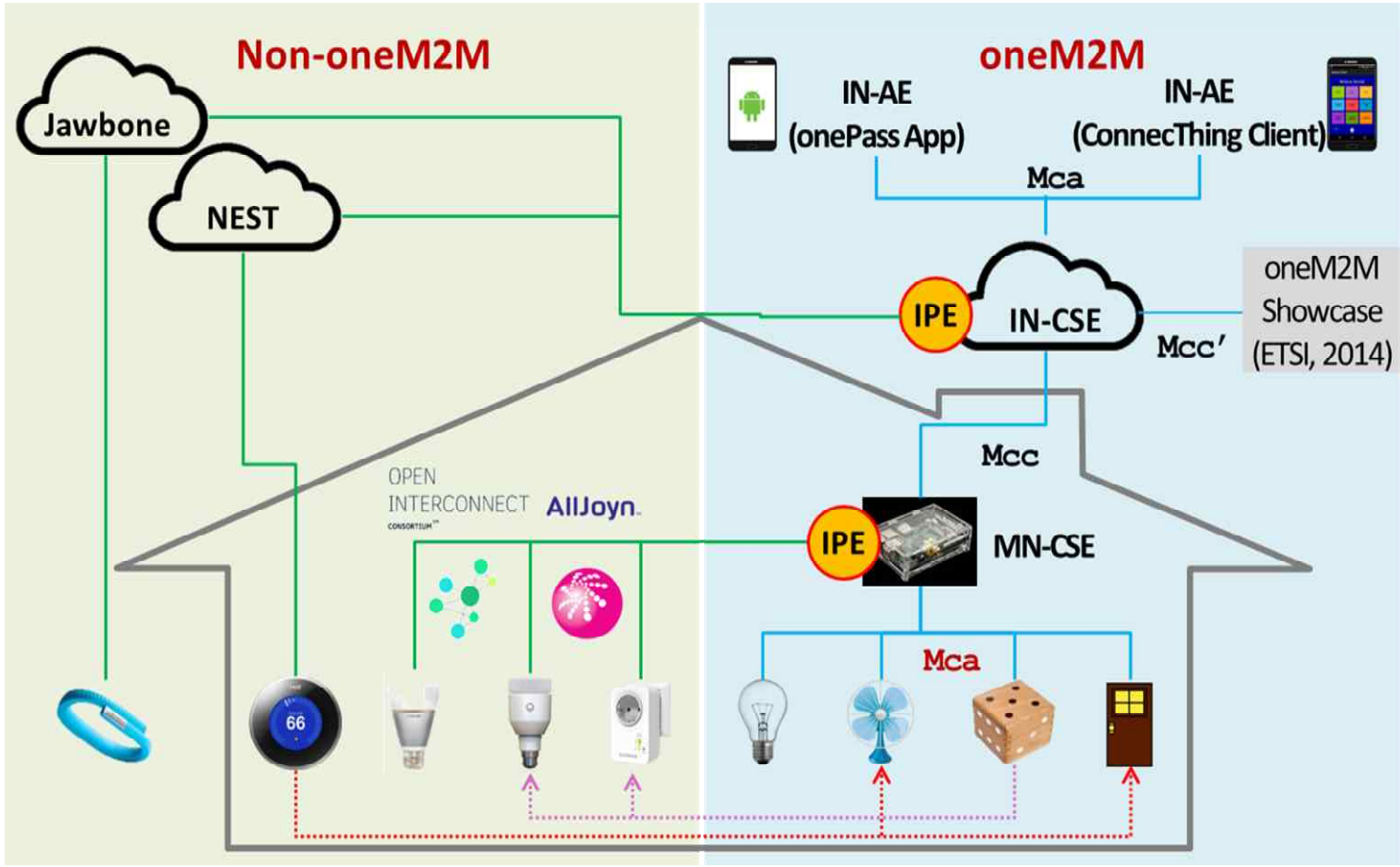
12.2 Extension

- oneM2M Interworking
  - AllJoyn, OIC(OCF), and Legacy systems



12.2 Extension

- oneM2M Interworking



12.2 Extension

- ConnecThing Demo

Source: https://youtu.be/FngoyncRhCM

12.2 Extension

- ConnecThing Demo
  - Interworking scenarios (oneM2M, AllJoyn, Legacy)

https://youtu.be/zHfIQwLBPsM

12.2 Extension

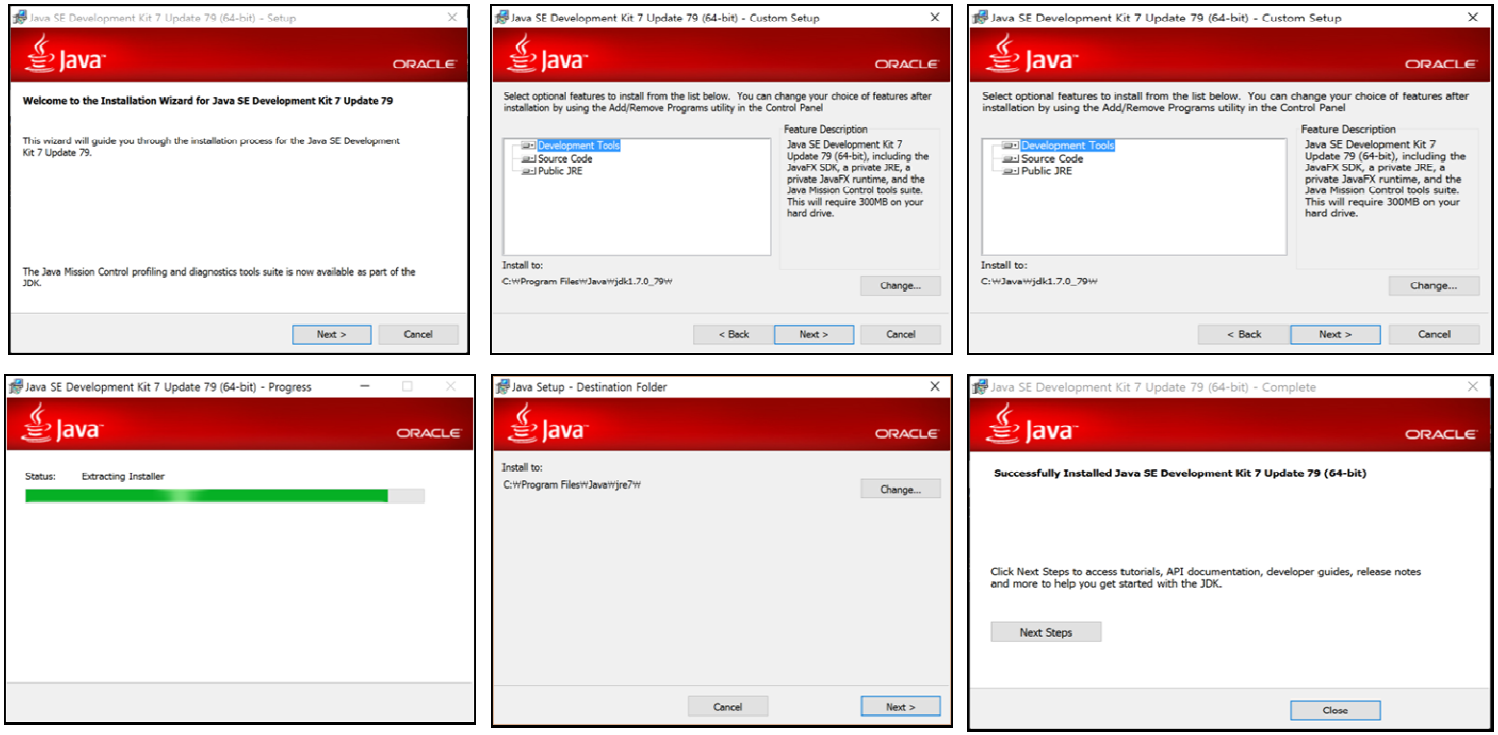
- ConnecThing Demo
  - All-in-one app & Interworking (OIC)

12.3 Develop Environment

- JDK Installation
  - JDK download and installation

https://developer.android.com/sdk/index.html

(Android Studio Install …, Java Development Kit 7u79 Install)

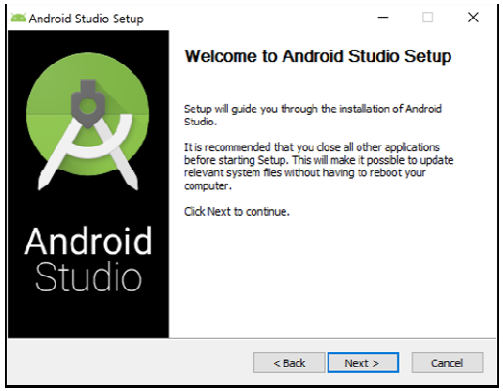


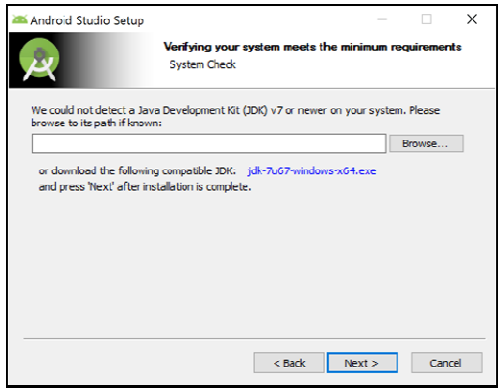
12.3 Develop Environment

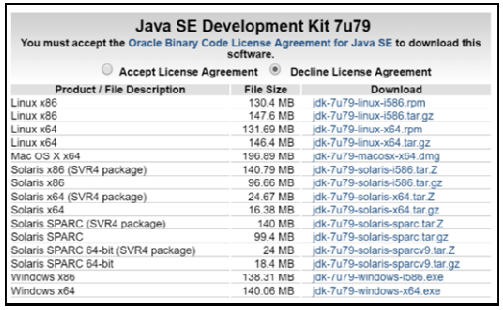
- Android Studio Installation
  - Android Studio download and installation

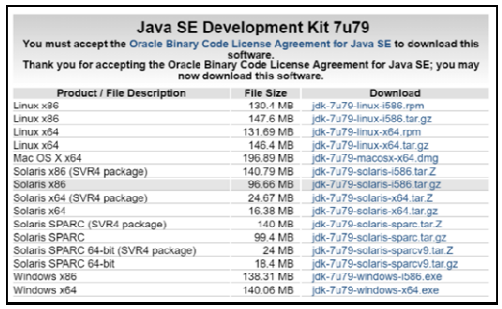
    https://developer.android.com/sdk/index.html

    (Android Studio Install …, Java Development Kit 7u79 Install)







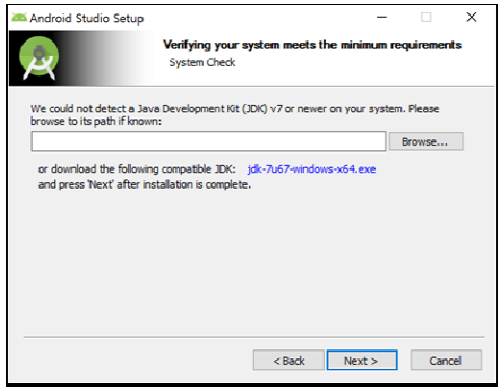


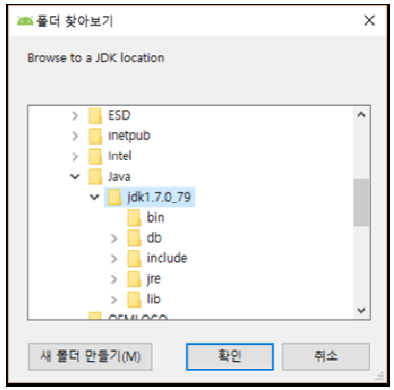
12.3 Develop Environment

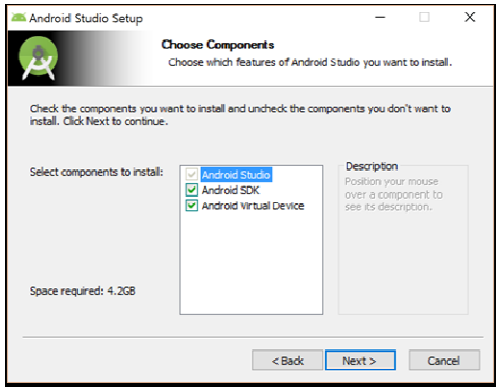
▪ Android Studio Installation

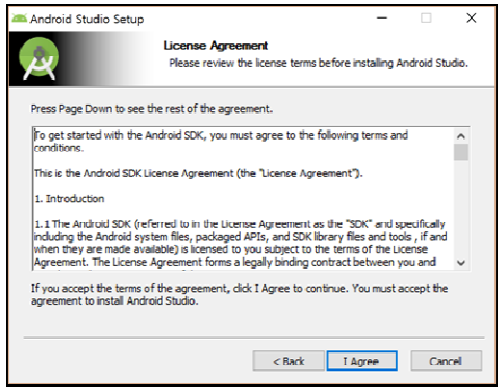
▪ Android Studio download and installation

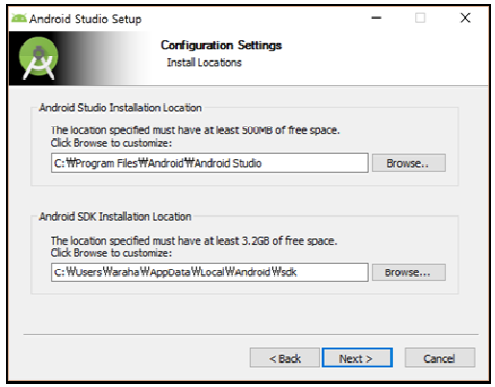
https://developer.android.com/sdk/index.html (Android Studio Install)

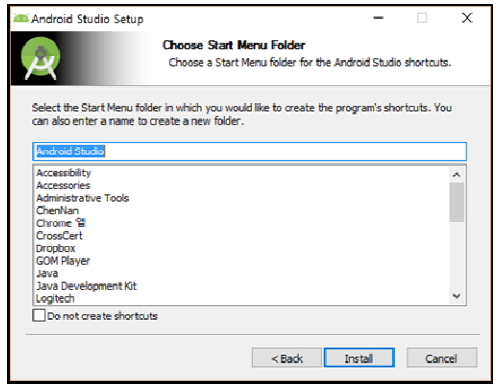










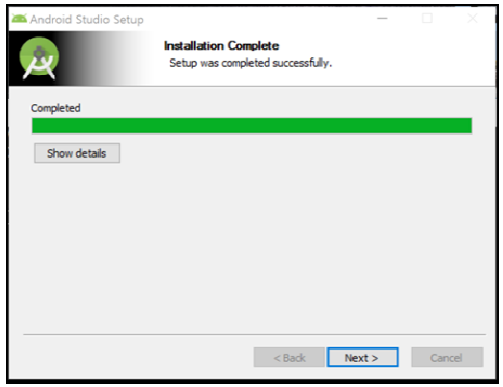


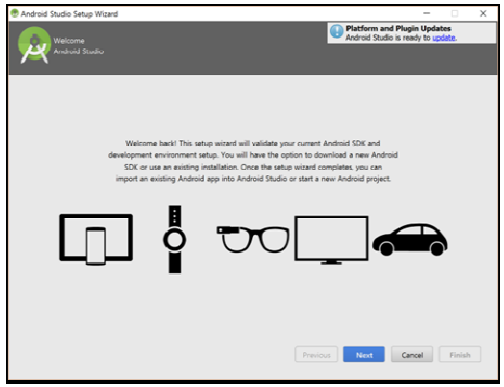
12.3 Develop Environment

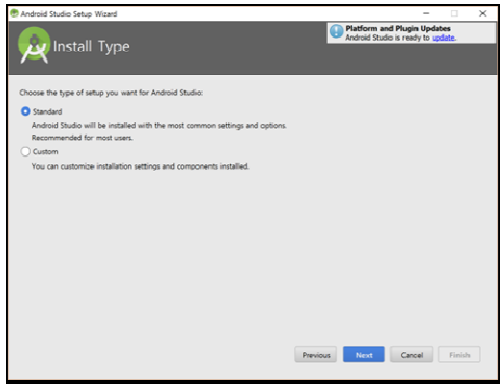
▪ Android Studio Installation

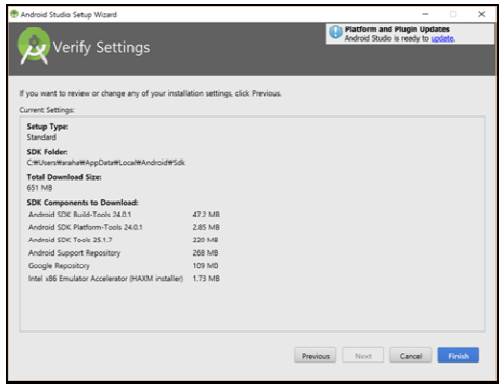
▪ Android Studio download and installation

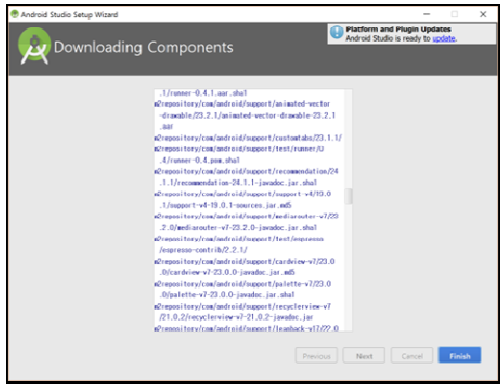
https://developer.android.com/sdk/index.html (Android Studio Install)

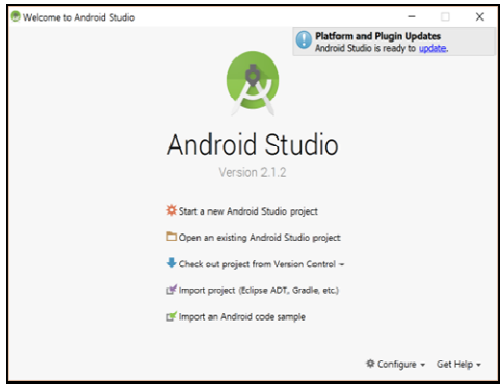










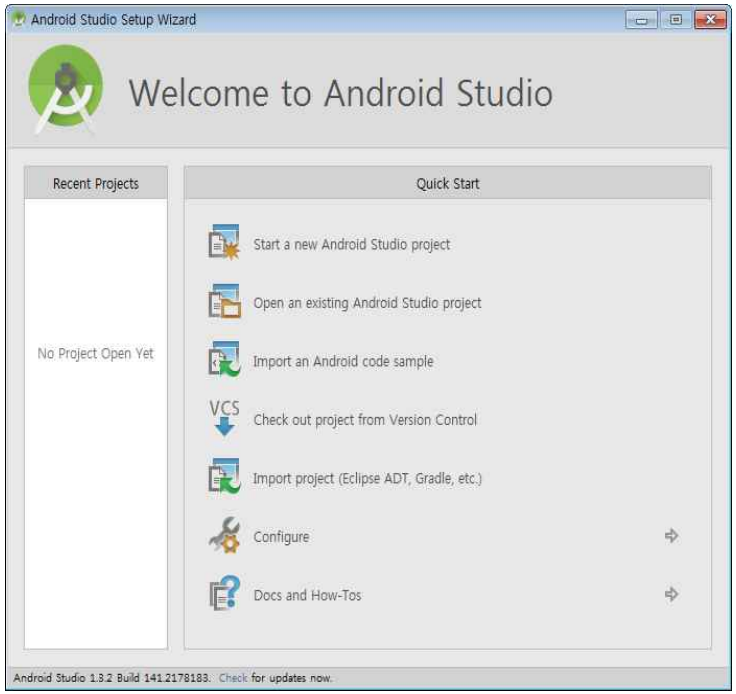


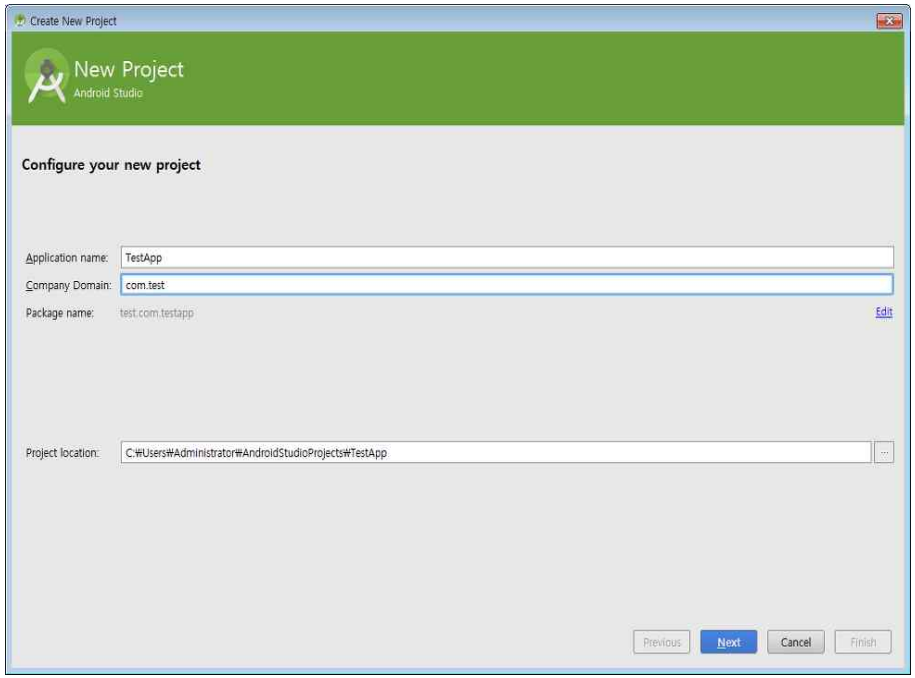
12.3 Develop Environment

▪ Android Studio Installation

▪ Execute Android Studio

- Click “Start a new Android Studio project”.
- Type the application name and company domain



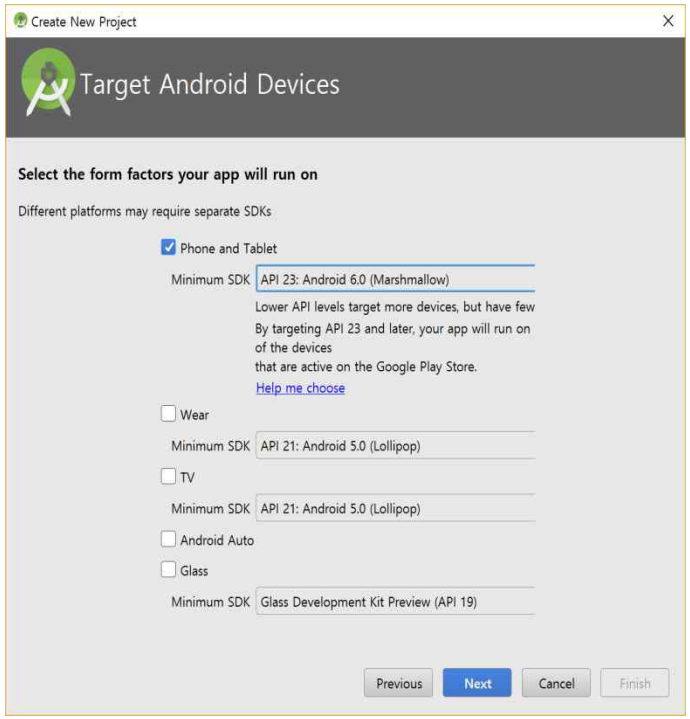


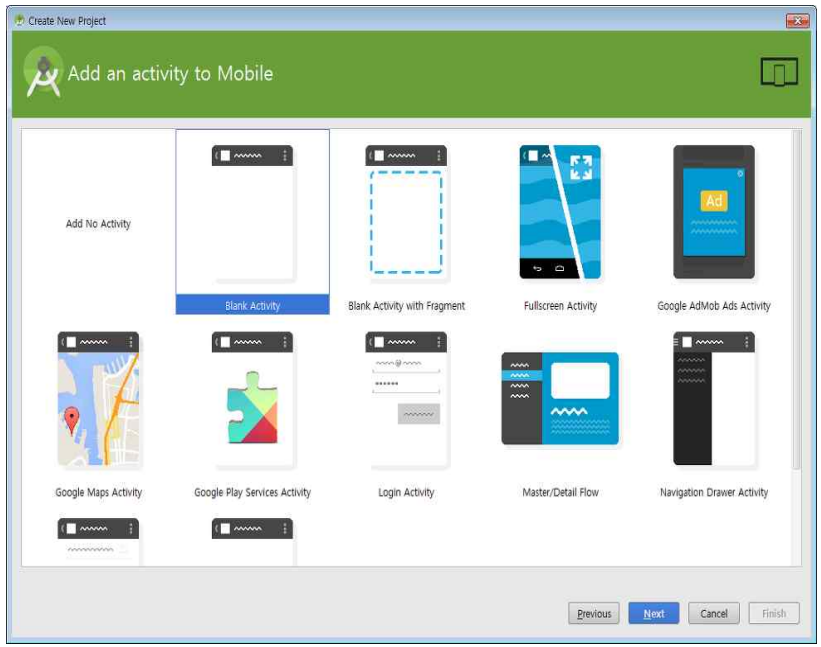
12.3 Develop Environment

▪ Android Studio Installation

▪ Configure ADK version and activity for targeting android devices

- Change Minimum SDK to API 23
- Select “Blank Activity”



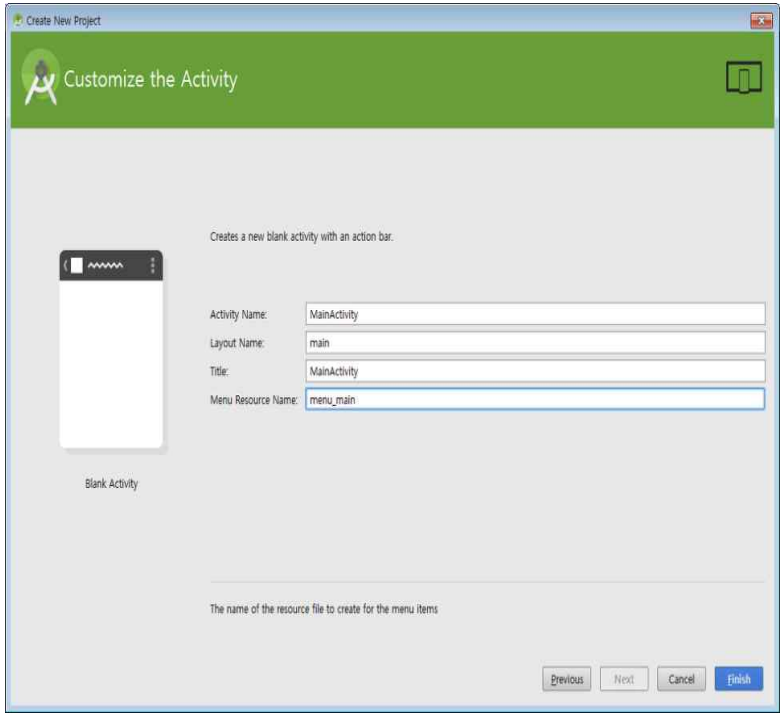


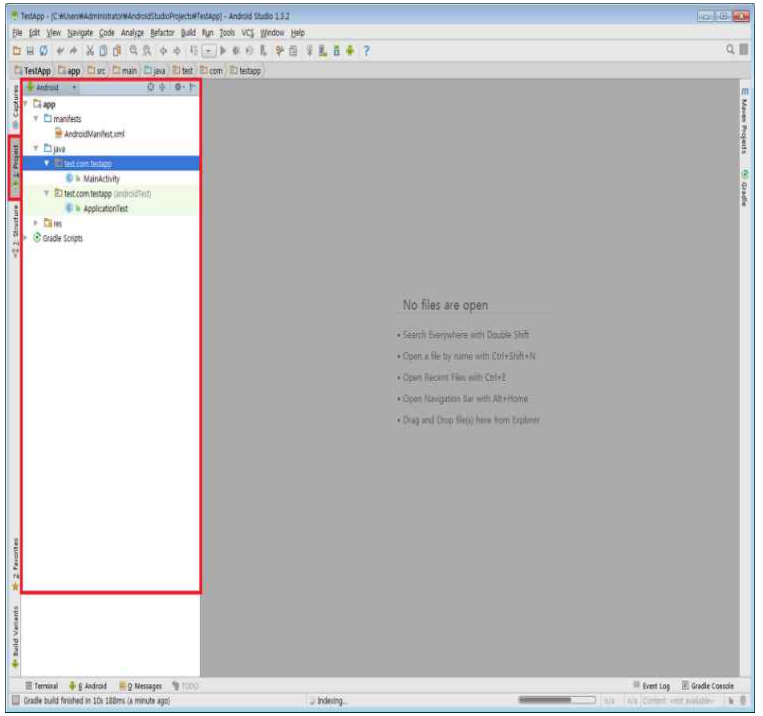
12.3 Develop Environment

▪ Android Studio Installation

▪ Customize the activity

- Input activity name and click “Finish” button.
- Right menu show the project file structure.

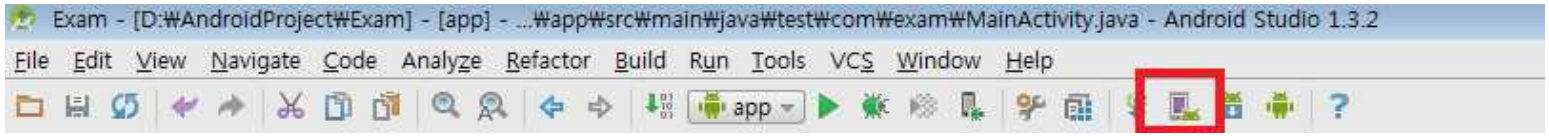




12.3 Develop Environment

▪ Android Studio Installation

▪ Android Studio Virtual Devices



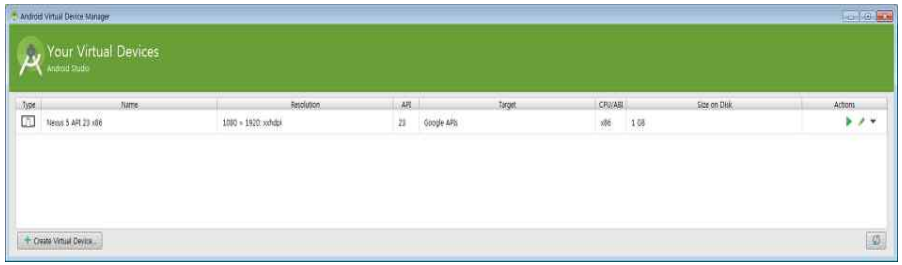
- Click "AVD Manager" menu

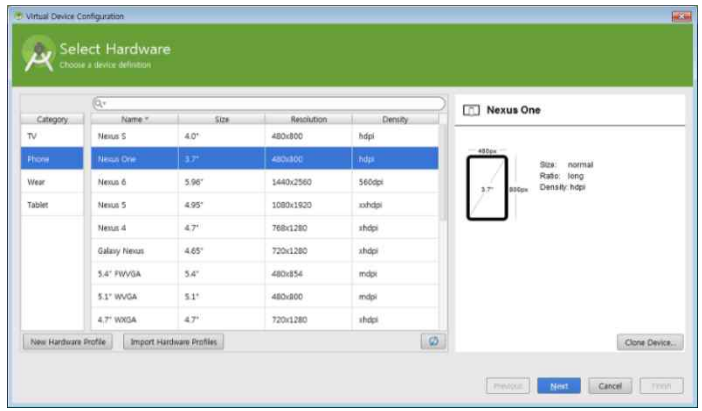
12. The default virtual device is Nexus 5

2. [Add a new virtual device]

3. Click "Create Virtual Device"

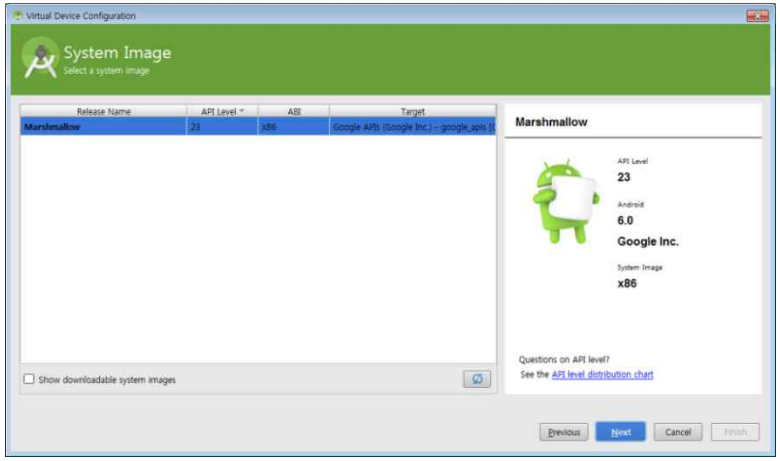
4. Select a the phone type as VD

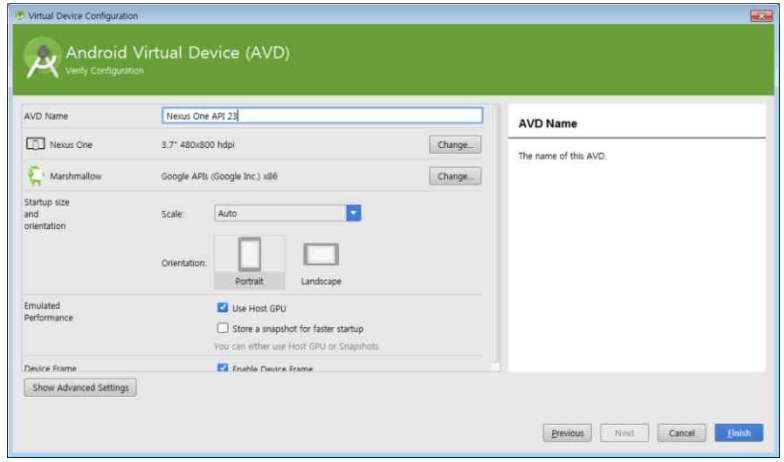


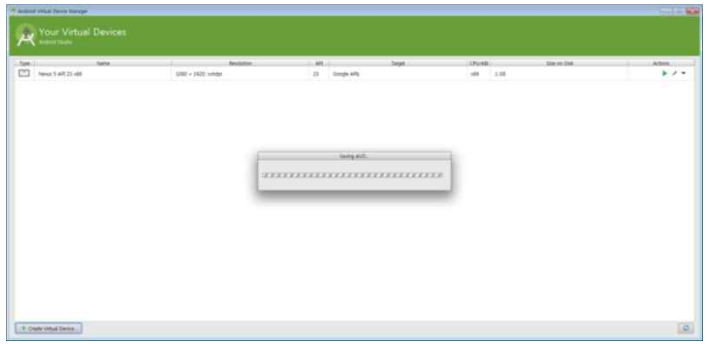


12.3 Develop Environment

- Android Studio Installation
  - Android Studio Virtual Devices

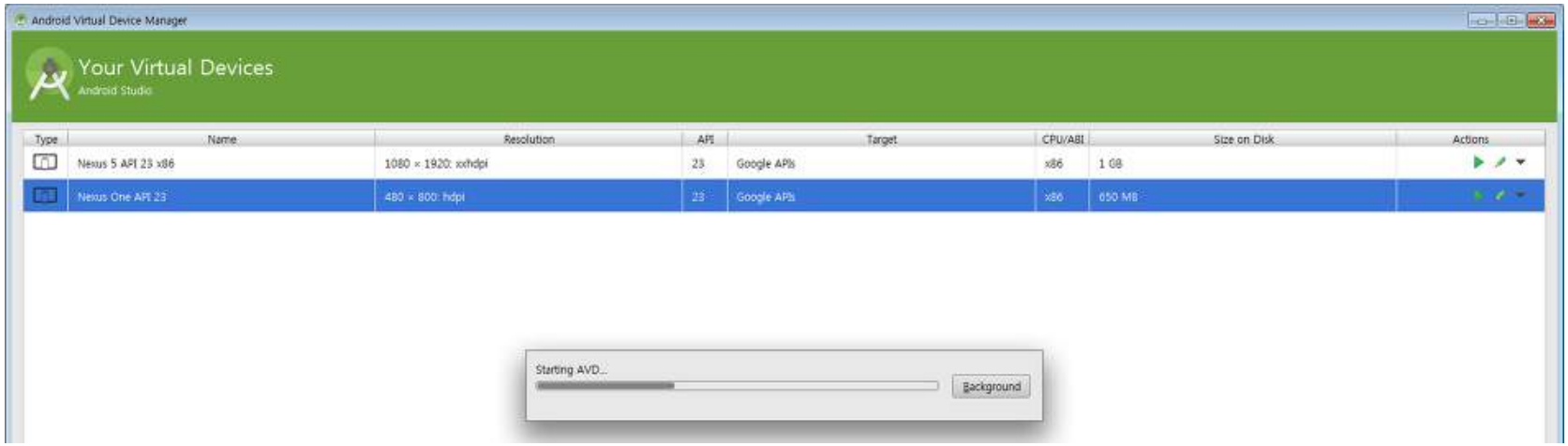






12.3 Develop Environment

- Android Studio Installation
  - Click start button to turn on the VD

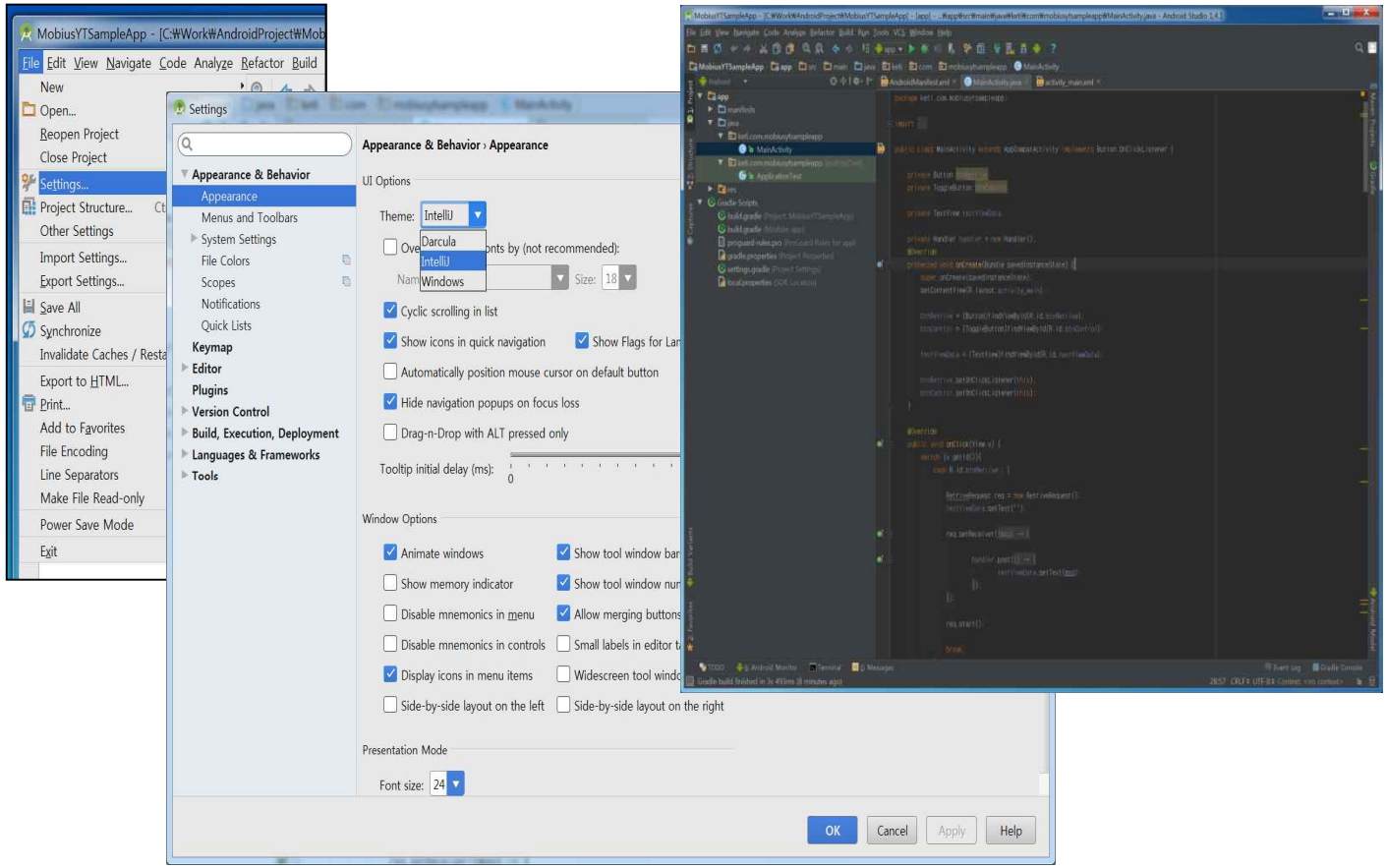


12.3 Develop Environment

- Android Studio Installation
  - Wait for the VD system loading

12.3 Develop Environment

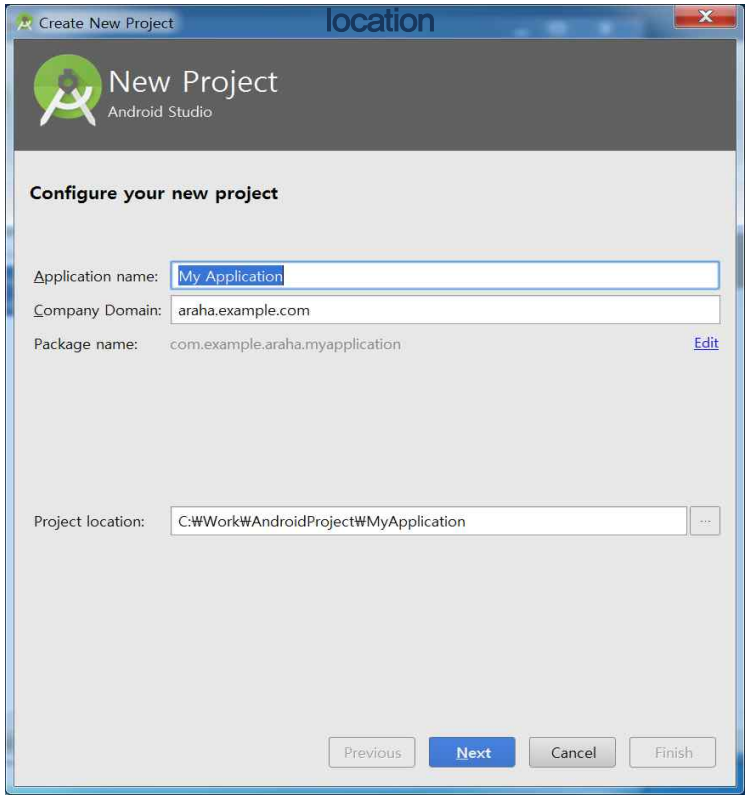
- Android Studio Installation
  - Android Studio Theme(Option)



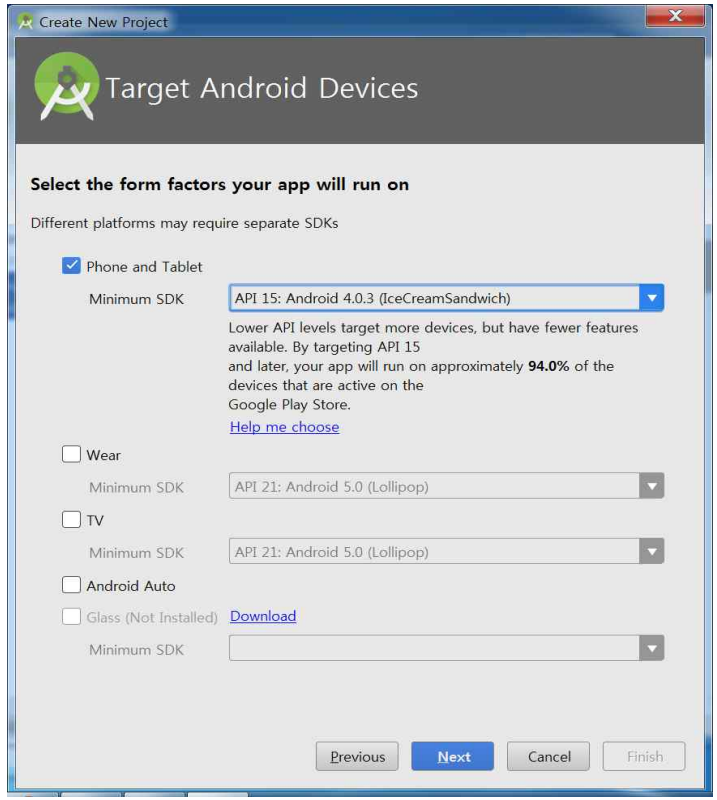
12.4 Practice

- Web View APP
  - New Project

Application Name –〉 Company Domain –〉 Project location



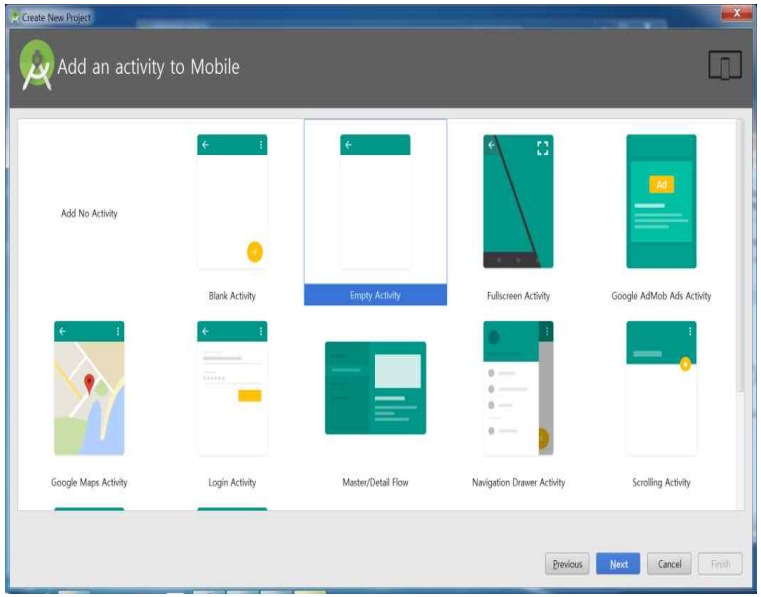
Configure Minimum SDK



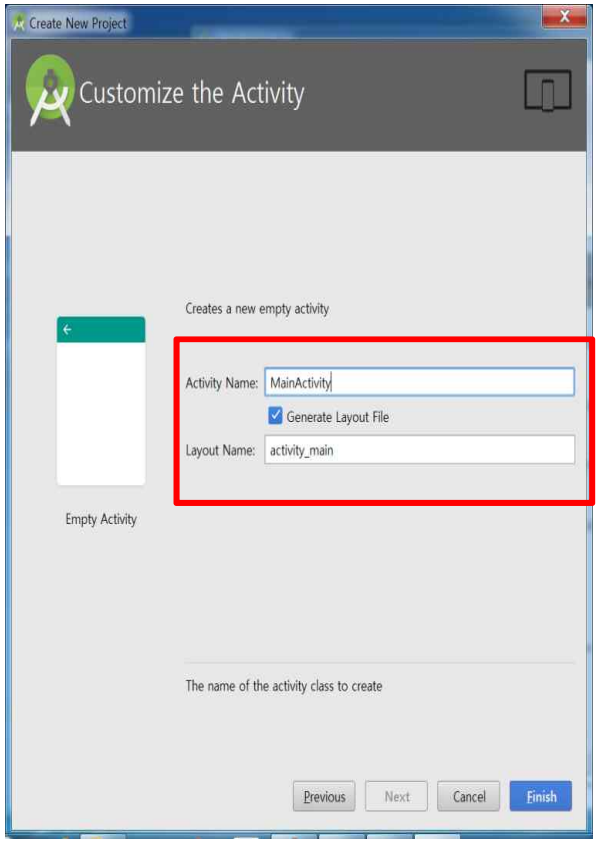
12.4 Practice

- Web View APP
  - New Project

Empty Activity



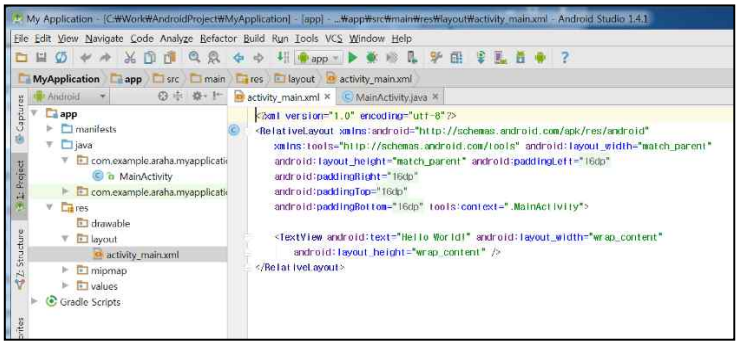
Setting Activity Name and Layout Name



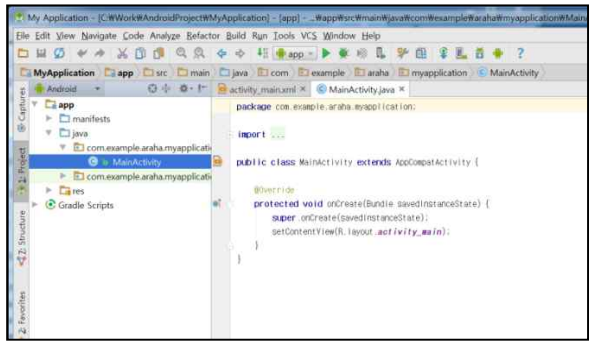
12.4 Practice

- Web View APP
  - Basic Default Code

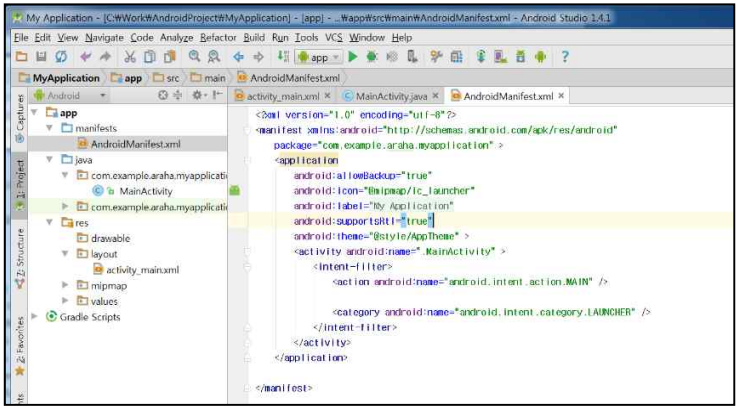
activity_main.xml



MainActivity.Java



AndroidManifest.xml



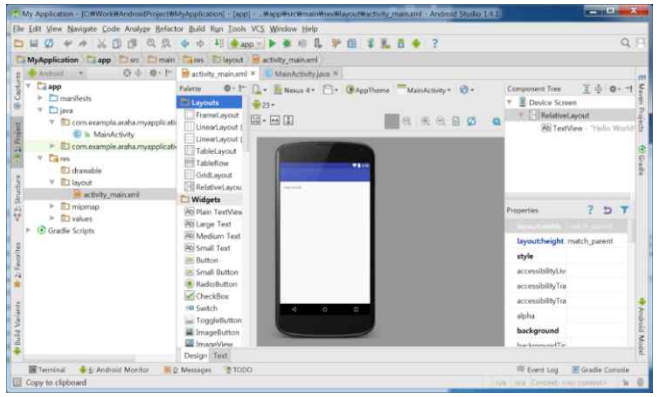
AndroidManifest?
Descript and configure the android project

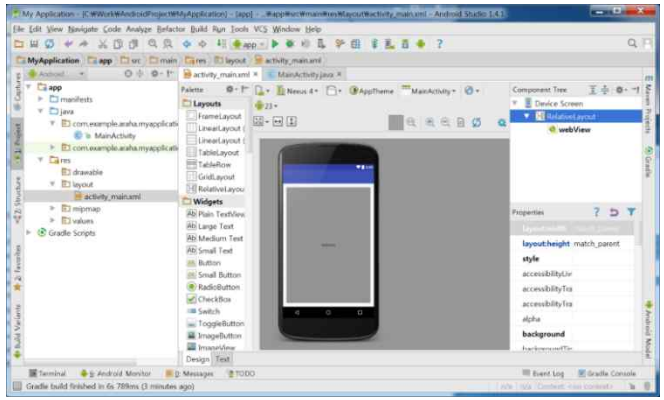
- App name, ID, icon, theme
- App access right
- Activity, Service, Intent

12.4 Practice

▪ Web View APP

▪ WebView Layout Source





activity_main.xml

```
〈?xml version="12.0" encoding="utf-8"?〉
〈RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin  “
    android:paddingBottom="@dimen/activity_vertical_margin”
tools:context=".MainActivity"〉
    〈WebView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/webView"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"/〉
〈/RelativeLayout〉
```

12.4 Practice

- Web View APP
  - WebView Java Source

MainActivity.Java

```
package com.example.araha.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle
savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

```
package com.example.araha.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Window;
import android.webkit.WebChromeClient;
import android.webkit.WebView;
import android.webkit.WebViewClient;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        WebView webView = new WebView(this);
        webView.getSettings().setJavaScriptEnabled(true);
        webView.setWebChromeClient(new WebChromeClient());
        webView.getSettings().setBuiltInZoomControls(true);
        webView.setWebViewClient(new WebViewClient());
        webView.loadUrl("http://www.naver.com");
        setContentView(webView);
    }
}
```

12.4 Practice

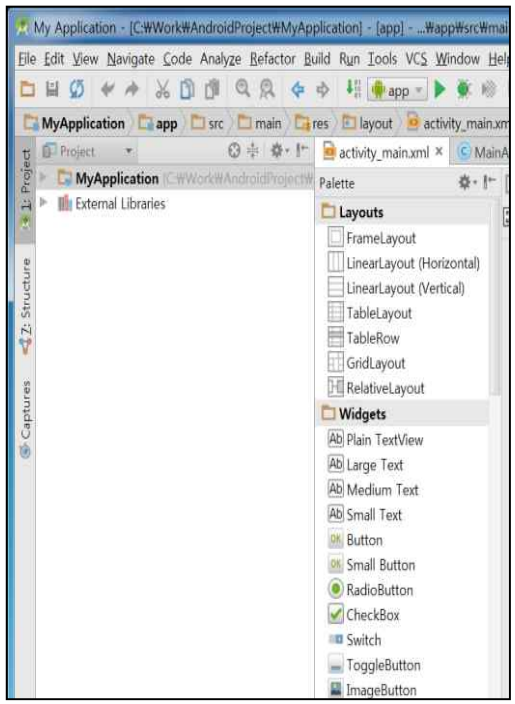
- Web View APP
  - WebView Java Source

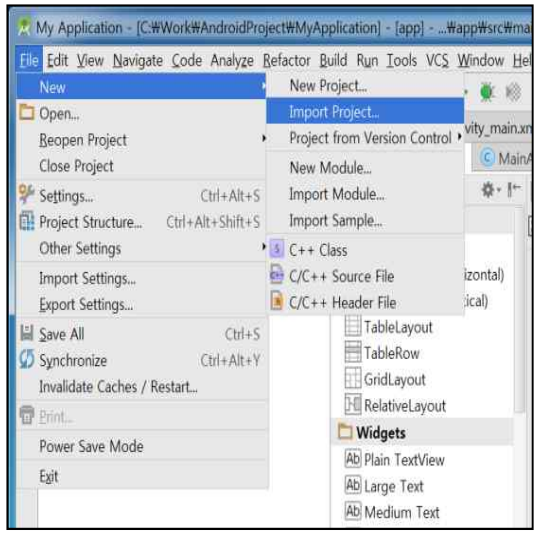
AndoridMainifest.xml

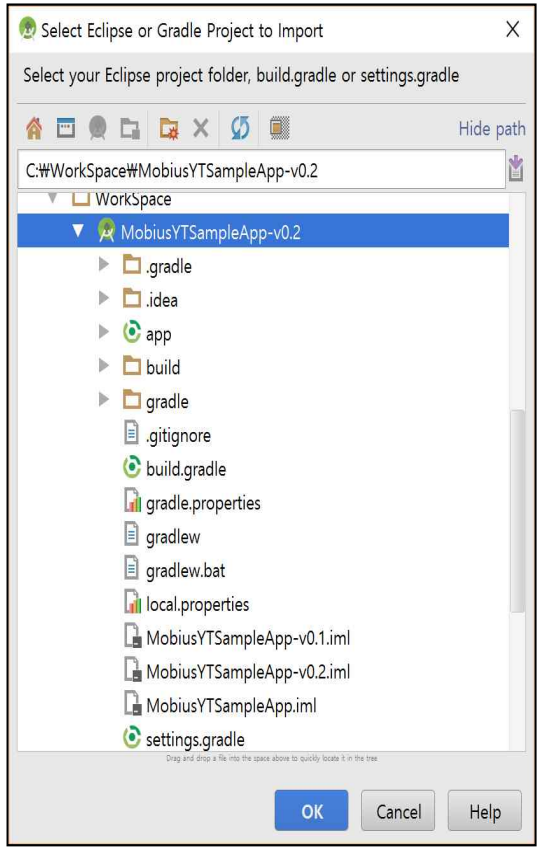
```
<?xml version="12.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.araha.myapplication" >
    <uses-permission android:name="android.permission.INTERNET"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme" >
        <activity android:name=".MainActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

12.4 Practice

- Mobius and APP connectivity
  - Import sample project







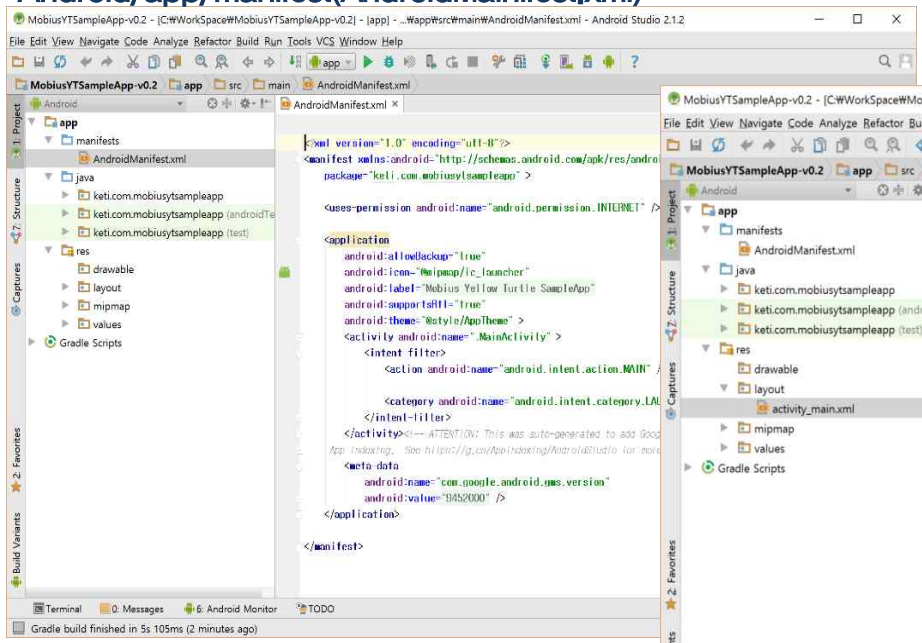
* Download the sample source from IoT Ocean(www.iotocean.org)

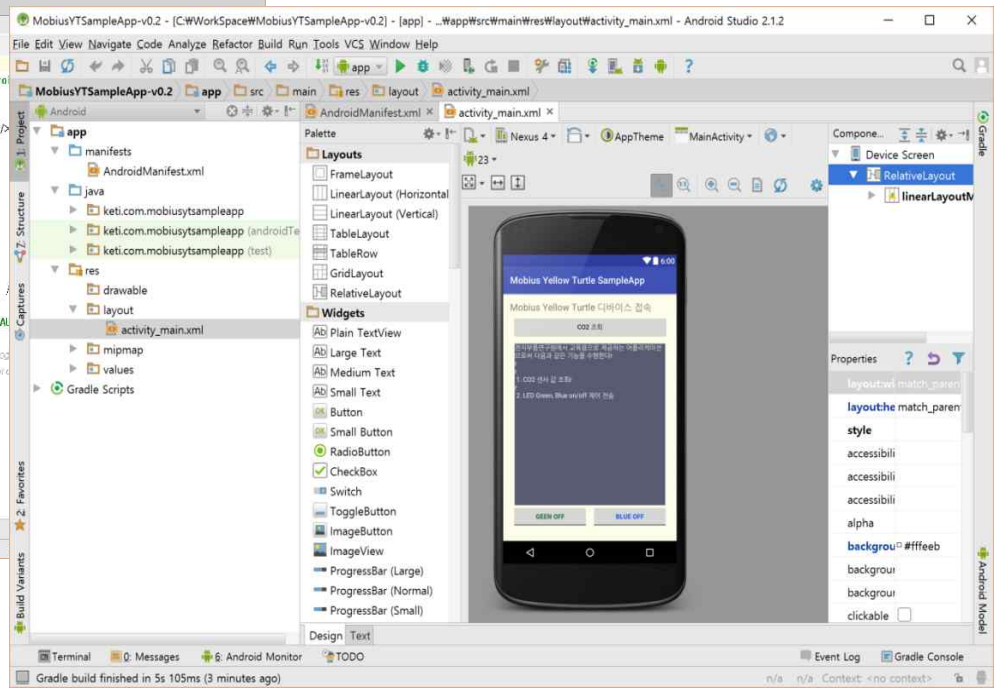
[Download/App Sample/Android App/MobiusYTSampleApp-v0.3]

12.4 Practice

- Mobius and APP connectivity
  - Import sample project

Android/app/manifest(AndroidMainifest.xml)



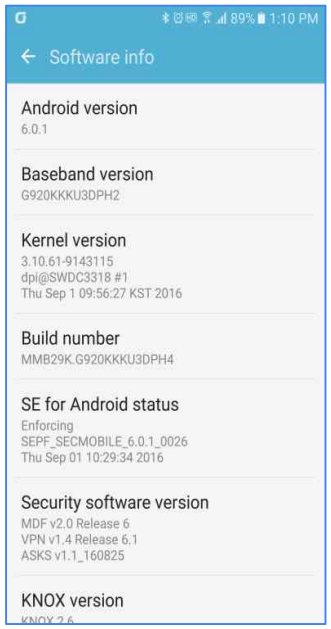


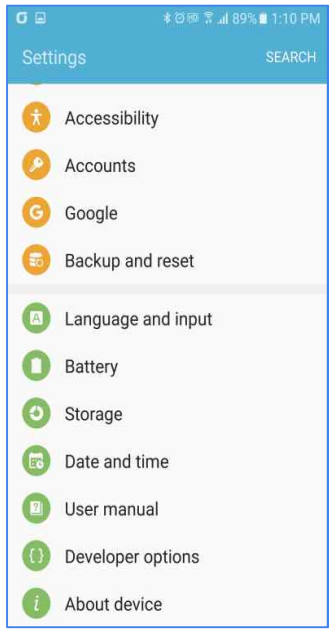
Android/app/res/layout/activity_main.xml(Design)

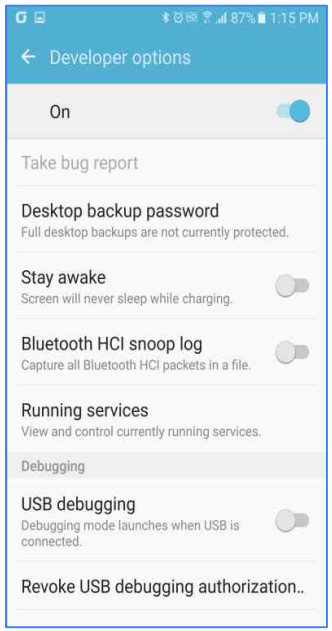
12.4 Practice

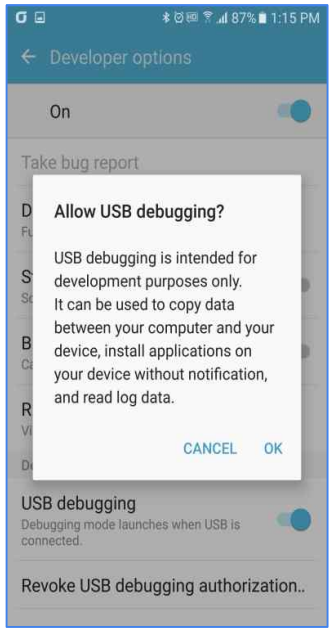
- Mobius and APP connectivity
  - USB Debugging Mode on Android

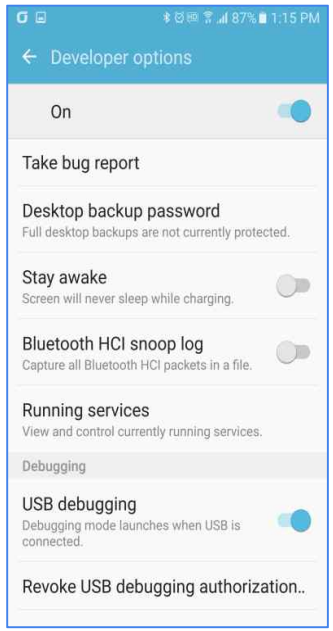
Change device to debugging mode









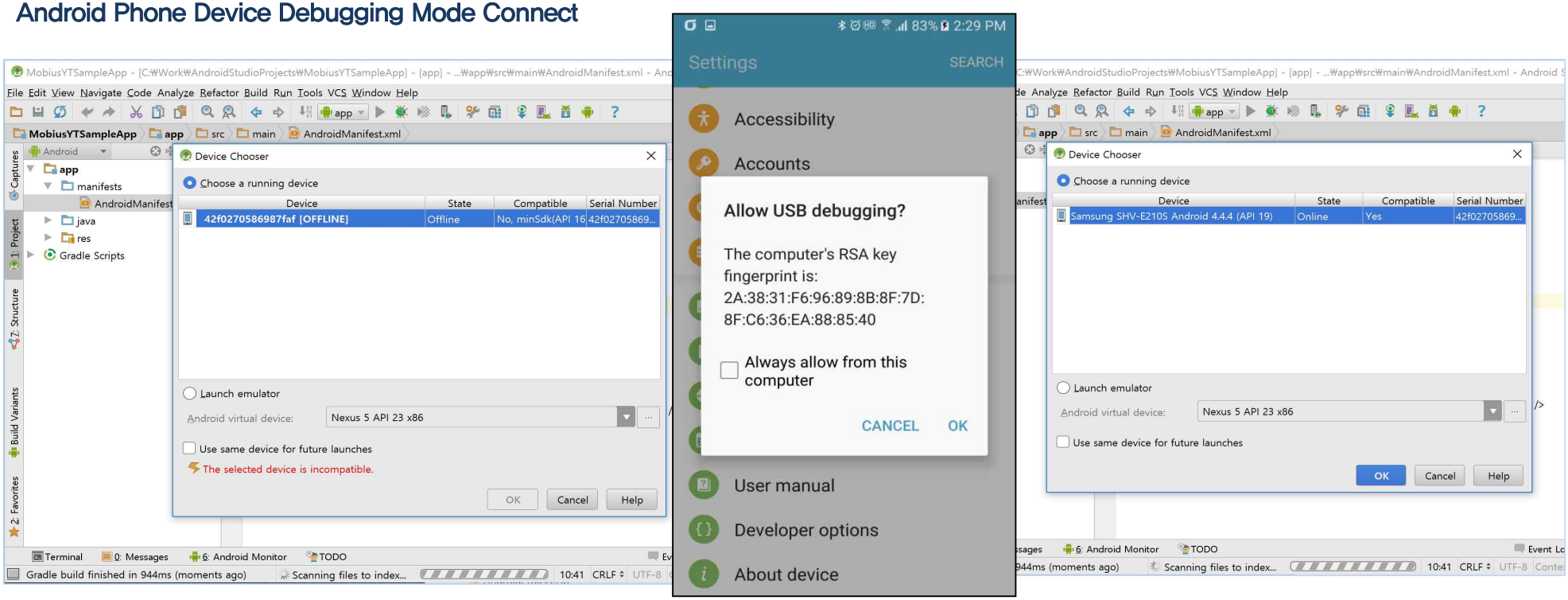


Click "Build number" continue

12.4 Practice

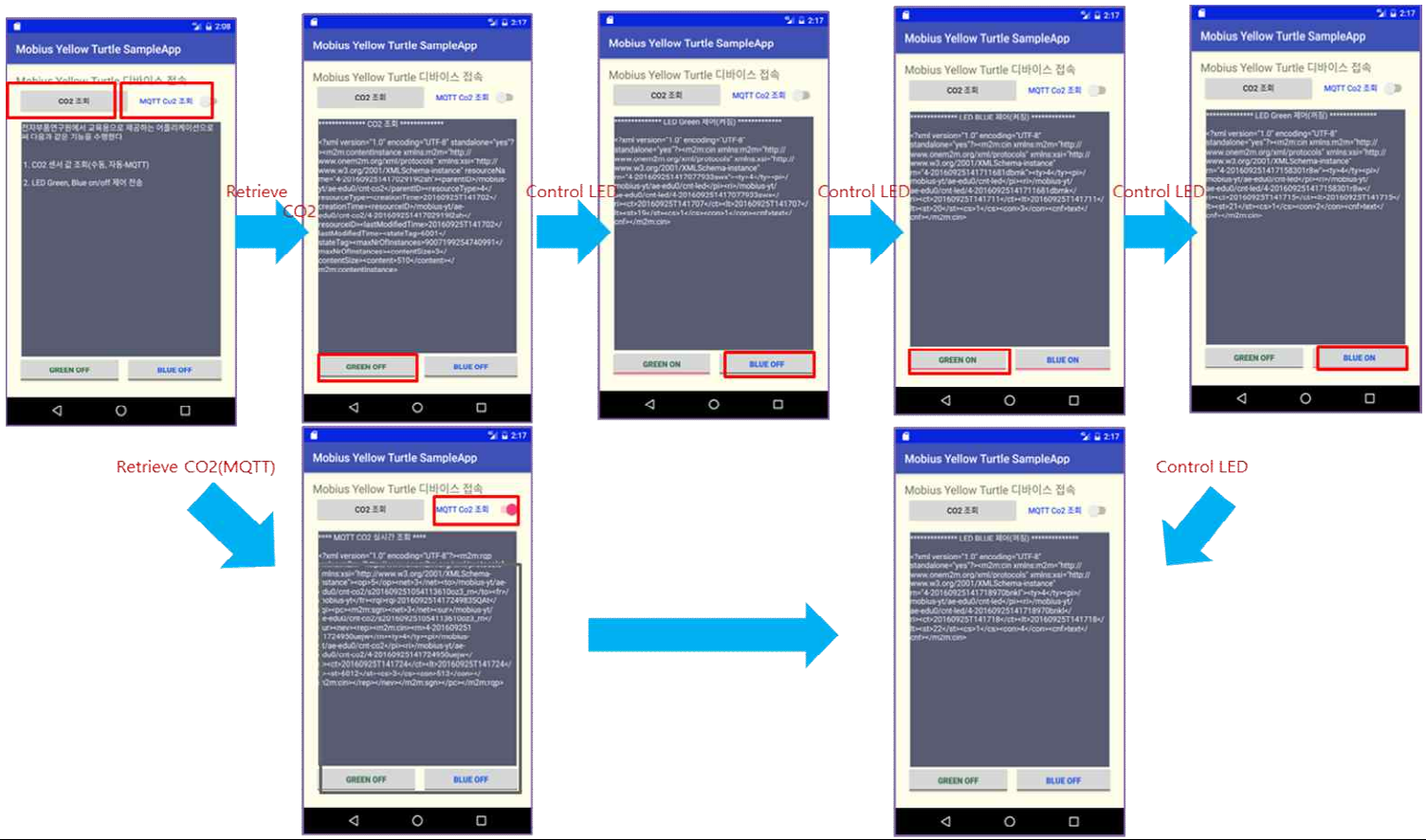
- Mobius and APP connectivity
  - App Running

Android Phone Device Debugging Mode Connect



12.4 Practice

- Mobius and APP connectivity
  - App Running



12.4 Practice

App Code Review

Internet Permission Setting [MQTT]

AndoridMainifest.xml

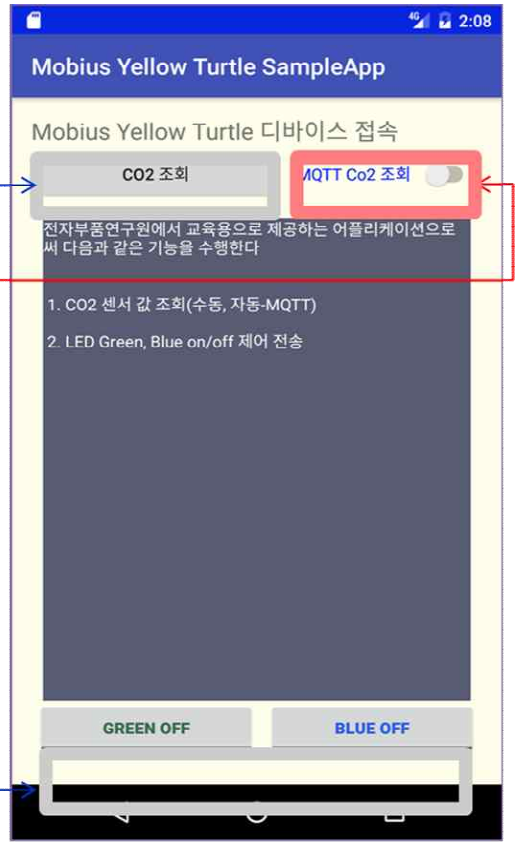
```
<?xml version="12.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="keti.com.mobiusytsampleapp" >

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme" >
        <activity android:name=".MainActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <service android:name="org.eclipse.paho.android.service.MqttService" />
    </application>

</manifest>
```

12.4 Practice

App Code Review

Code Structure Review

```
// Main
public MainActivity() { handler = new Handler(); }
/* onCreate */
protected void onCreate(Bundle savedInstanceState) {...}
/* AE Create for Androdi AE */
public void GetAEInfo() {...}
/* Switch - Get Co2 Data With MQTT */
public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {...}
/* MQTT Subscription */
public void MQTT_Create(boolean mtqqStart) {...}
/* MQTT Listener */
private IMqttActionListener mainIMqttActionListener = new IMqttActionListener() {...};
/* MQTT Broker Message Received */
private MqttCallback mainMqttCallback = new MqttCallback() {...};
@Override
public void onClick(View v) {...}
@Override
public void onStart() {...}
@Override
public void onStop() {...}


/* Response callback Interface */
public interface IReceived {...}

/* Retrieve Co2 Sensor */
class RetrieveRequest extends Thread {...}
/* Request Control LED */
class ControlRequest extends Thread {...}
/* Request AE Creation */
class aeCreateRequest extends Thread {...}
/* Retrieve AE-ID */
class aeRetrieveRequest extends Thread {...}
/* Subscribe Co2 Content Resource */
class SubscribeResource extends Thread {...}
```

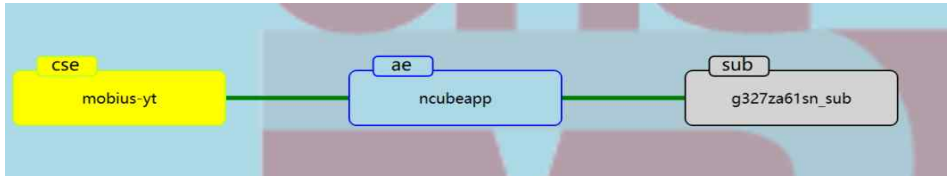


12.4 Practice

App Code Review

Code Structure Review [AE Create or AE Retrieve …]

```
/* Request AE Creation */
class aeCreateRequest extends Thread {
    private final Logger LOG = Logger.getLogger(aeCreateRequest.class.getName());
    String TAG = aeCreateRequest.class.getName();
    private IReceived receiver;
    int responseCode=0;
    public ApplicationEntityObject applicationEntity;
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }
    public aeCreateRequest(){
        applicationEntity = new ApplicationEntityObject();
        applicationEntity.setResourceName(ae.getappName());
    }
    @Override
    public void run() {...}
}
/* Retrieve AE-ID */
class aeRetrieveRequest extends Thread {
    private final Logger LOG = Logger.getLogger(aeCreateRequest.class.getName());
    private IReceived receiver;
    int responseCode=0;

    public aeRetrieveRequest() {...}
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }

    @Override
    public void run() {...}
}
```




```
<?xml version="1.0" encoding="utf-16" standalone="yes"?>
<m2m:ae xmlns:m2m="http://www.oneM2M.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="ncubeapp">
  <pi>/mobius-yt</pi>
  <ty>2</ty>
  <ct>20160928T161212</ct>
  <ri>/mobius-yt/ncubeapp</ri>
  <lt>20160928T161212</lt>
  <api>0.2.4812.2.00012.0012.0001114</api>
  <aei>xxxxxx</aei>
  <rr>true</rr>
</m2m:ae>
```

Note: In oneM2M standard Android Application is mapping with Application Entity(AE). It should use the AE-ID as request “origin” to do other resource operation(Create, Retrieve, Update, Delete).

12.4 Practice

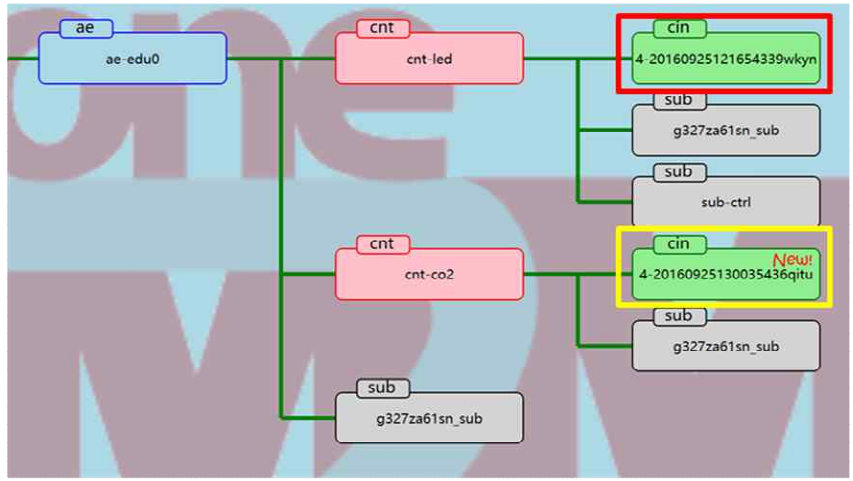
App Code Review

Code Structure Review [CO2 Retrieve, LED Control]

```
/* Retrieve Co2 Sensor */
class RetrieveRequest extends Thread {
    private final Logger LOG = Logger.getLogger(RetrieveRequest.class.getName());
    private IReceived receiver;
    private String ContainerName = "cnt-co2";

    public RetrieveRequest(String containerName) {
        this.ContainerName = containerName;
    }
    public RetrieveRequest() {}

    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }

    @Override
    public void run() {...}
}
```

```
/* Request Control LED */
class ControlRequest extends Thread {
    private final Logger LOG = Logger.getLogger(ControlRequest.class.getName());
    private IReceived receiver;
    private String container_name = "cnt-led";

    public ContentInstanceObject contentinstance;

    public ControlRequest(String comm) {
        contentinstance = new ContentInstanceObject();
        contentinstance.setContent(comm);
    }

    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }

    @Override
    public void run() {...}
}
```

```
<?xml version="1.0" encoding="utf-16" standalone="yes"?><m2m:cin
xmlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="4-
20160925121654339wkyn"><pi>/mobius-yt/ae-edu0/cnt-
led</pi><ty>4</ty><ct>20160925T121654</ct><ri>/mobius-yt/ae-edu0/cnt-
led/4-
20160925121654339wkyn</ri><lt>20160925T121654</lt><st>18</st><mni>90
07199254740991</mni><cs>1</cs><cnf>text</cnf><con>4</con></m2m:cin>
```



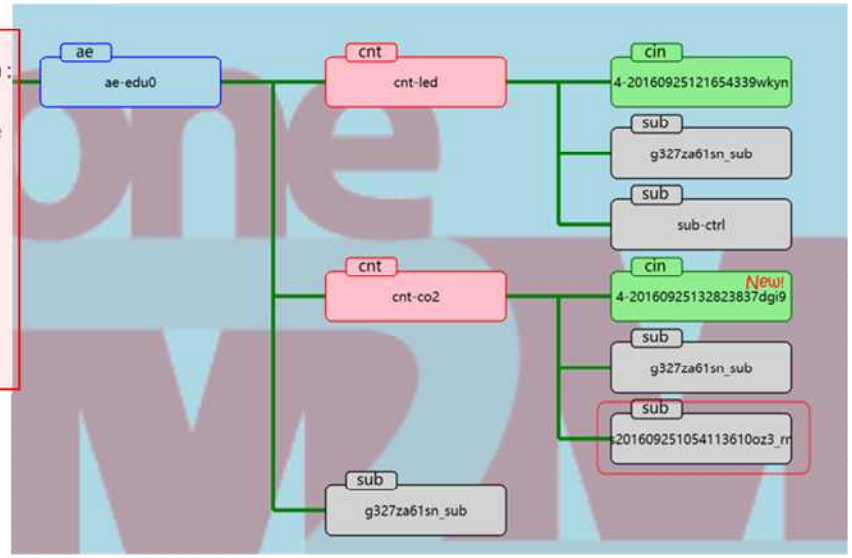

```
<?xml version="1.0" encoding="utf-16" standalone="yes"?><m2m:cin
xmlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="4-
20160925130205542urps"><pi>/mobius-yt/ae-edu0/cnt-
co2</pi><ty>4</ty><ct>20160925T130205</ct><ri>/mobius-yt/ae-edu0/cnt-co2/4-
20160925130205542urps</ri><lt>20160925T130205</lt><st>3756</st><mni>90071
99254740991</mni><cs>3</cs><con>719</con></m2m:cin>
```

12.4 Practice

App Code Review

Code Structure Review [Subscription create for MQTT]

```
/* Subscribe Co2 Content Resource */
class SubscribeResource extends Thread {
    private final Logger LOG = Logger.getLogger(SubscribeResource.class.getName());
    private IReceived receiver;
    private String container_name = "cnt-co2"; //change to control container name

    public ContentSubscribeObject subscribeInstance;
    public SubscribeResource() {...}
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }

    @Override
    public void run() {...}
}
```




```
<?xml version="1.0" encoding="utf-16" standalone="yes"?><m2m:sub
xmlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
rn="s20160925105411361oz3_rn"><pi>/mobius-yt/ae-edu0/cnt-
co2</pi><ty>23</ty><ct>20160925T133651</ct><ri>/mobius-yt/ae-edu0/cnt-
co2/s20160925105411361oz3_rn</ri><lt>20160925T133651</lt><st>0</st><en
c><net>3</net></enc><nu>mqtt://192.168.25.47/S20160925105411361oz3_su
b</nu><nct>2</nct><cr>S20160925105411361oz3</cr></m2m:sub>
```

12.4 Practice

App Code Review

Code Structure Review [Button Click Event]

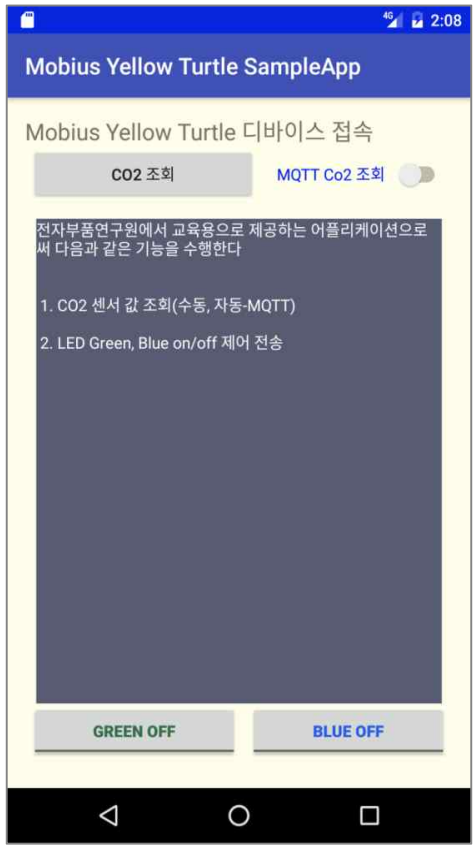
```
case R.id.btnControl_Green: {
                if (((ToggleButton) v).isChecked()) {
                    ControlRequest req = new ControlRequest("1");
                    req.setReceiver(new IReceived() {
                        @Override
                        public void getResponseBody(final String msg) {
                            handler.post(new Runnable() {
                                @Override
                                public void run() {
                                    textViewData.setText("********** LED Green 제어(켜짐).." );
                                }
                            });
                        }
                    });
                    req.start();
                } else {
                    ControlRequest req = new ControlRequest("2");
                    req.setReceiver(new IReceived() {
                        @Override
                        public void getResponseBody(final String msg) {
                            handler.post(new Runnable() {
                                @Override
                                public void run() {
                                    textViewData.setText("********** LED Green 제어(꺼짐).." );
                                }
                            });
                        }
                    });
                    req.start();
                }
                break;
            }
```

```
public void onClick(View v) {

    switch (v.getId()) {

        case R.id.btnRetrieve: {
            RetrieveRequest req = new RetrieveRequest();
            textViewData.setText("");
            req.setReceiver(new IReceived() {
                @Override
                public void getResponseBody(final String msg) {
                    handler.post(new Runnable() {
                        @Override
                        public void run() {
                            textViewData.setText("********** CO2 조회 ***;
                        }
                    });
                }
            });
            req.start();
            break;
        }
    ...

                ..
                .
```

12.4 Practice

▪ App Code Review

▪ Code Structure Review [Switch onCheckChanged Event]

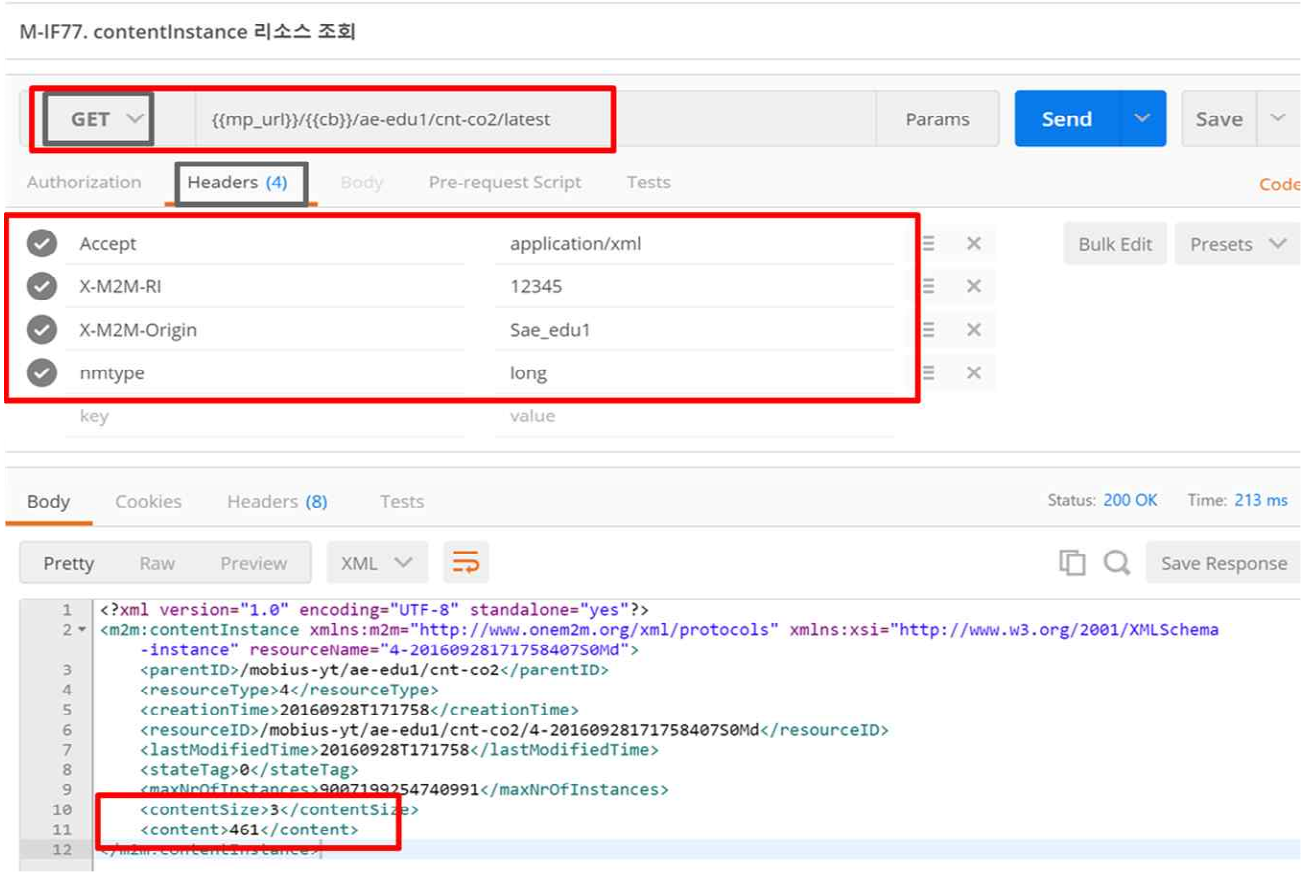



```
public void onCheckedChanged(CompoundButton buttonView, boolean isChecked)
if (isChecked) {
        Log.d(TAG, "MQTT Create");
        MQTT_Create(true);
    } else {
        Log.d(TAG, "MQTT Close");
        MQTT_Create(false);
    }
}

/* MQTT Subscription */
public void MQTT_Create(boolean mtqqStart) {
    if (mtqqStart && mqttClient == null) {
        /* Subscription Resource Create to Yellow Turtle */
        SubscribeResource subcribeResource = new SubscribeResource();
        subcribeResource.setReceiver(new IReceived() {
            public void getResponseBody(final String msg) {
                handler.post(new Runnable() {
                    public void run() {   ···    }
                });
            }
        });
        subcribeResource.start();

        /* MQTT Subscribe */
        mqttClient = new MqttAndroidClient(this.getApplicationContext(), "tcp://" + csebase.getHost() + ":" + csebase.getMQTTPort(),
MqttClient.generateClientId());
        mqttClient.setCallback(mainMqttCallback);
        try {
            IMqttToken token = mqttClient.connect();
            token.setActionCallback(mainIMqttActionListener);
        } catch (MqttException e) {
            e.printStackTrace();
        }
    } else {
        /* MQTT unSubscribe or Client Close */
        mqttClient.setCallback(null);
        mqttClient.close();
        mqttClient = null;
    }
}
```

12.4 Practice

App Code Review

Retrieve by POSTMAN [CO2 Sensing Data]



12.4 Practice

- App Code Review
  - Retrieve by Java Code
    [CO2 Sensing Data]

```
private String ServiceAEName = "ae-edu1";

...

public void GetAEInfo() {

    csebase.setInfo("192.168.25.47","7579","mobius-yt","1883"); //Sample IP
    ...

    ...
}

/* Retrieve Co2 Sensor */
class RetrieveRequest extends Thread {
    private final Logger LOG = Logger.getLogger(RetrieveRequest.class.getName());
    private IReceived receiver;
    private String ContainerName = "cnt-co2";

    public RetrieveRequest(String containerName) {
        this.ContainerName = containerName;
    }
    public RetrieveRequest() {}
    public void setReceiver(IReceived handler) { this.receiver = handler; }
```

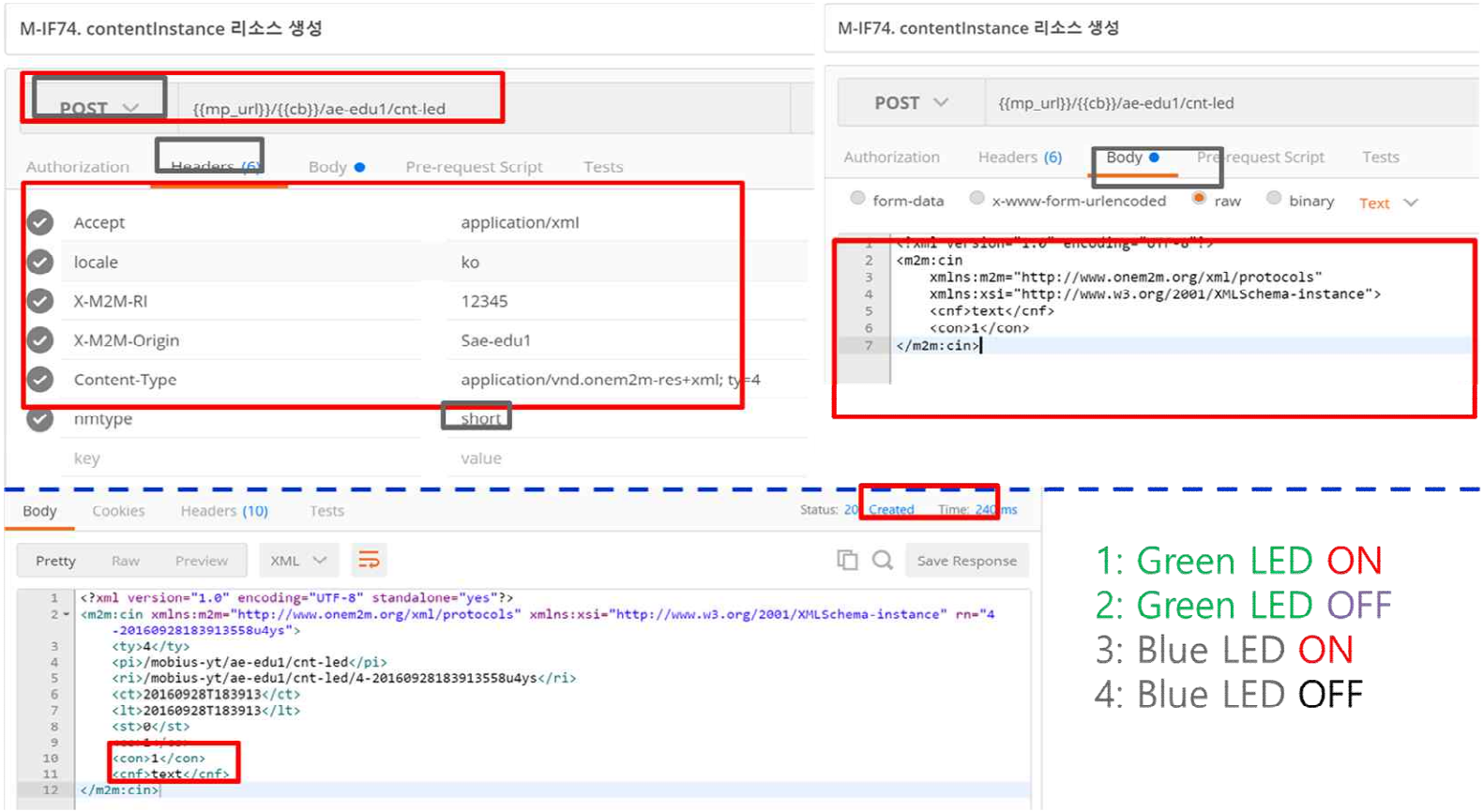
```
@Override
public void run() {
    try {
        String sb = csebase.getServiceUrl() + "/" + ServiceAEName + "/" + ContainerName + "/" + "latest";

        URL mUrl = new URL(sb);

        HttpURLConnection conn = (HttpURLConnection) mUrl.openConnection();
        conn.setRequestMethod("GET");
        conn.setDoInput(true);
        conn.setDoOutput(false);

        conn.setRequestProperty("Accept", "application/xml");
        conn.setRequestProperty("X-M2M-RI", "12345");
        conn.setRequestProperty("X-M2M-Origin", ae.getAEid() );
        conn.setRequestProperty("nmtype", "long");
        conn.connect();

        String strResp = "";
        BufferedReader in = new BufferedReader(new InputStreamReader(conn.getInputStream()));

        String strLine= "";
        while ((strLine = in.readLine()) != null) {
            strResp += strLine;
        }

        if ( strResp != "" ) {
            receiver.getResponseBody(strResp);
        }
        conn.disconnect();

    } catch (Exception exp) {
        LOG.log(Level.WARNING, exp.getMessage());
    }
}
}
```

```
<?xml version="1.0" encoding="UTF-8" standalone="
<m2m:contentInstance xmlns:m2m="http://www.onem2m
    -instance" resourceName="4-20160928171758407S
    <parentID>/mobius-yt/ae-edu1/cnt-co2</parentI
    <resourceType>4</resourceType>
    <creationTime>20160928T171758</creationTime>
    <resourceID>/mobius-yt/ae-edu1/cnt-co2/4-2016
    <lastModifiedTime>20160928T171758</lastModifi
    <stateTag>0</stateTag>
    <maxNrOfInstances>9007199254740991</maxNrOfIn
    <contentSize>3</contentSize>
    <content>461</content>
</m2m:contentInstance>
```

12.4 Practice

App Code Review

Control by POSTMAN [LED Control]



12.4 Practice

- App Code Review
  - Control by Java Code
    [LED Control]

```
private String ServiceAEName = "ae-edu1";

...

public void GetAEInfo() {

    csebase.setInfo("192.168.25.47","7579","mobius-yt","1883");
    ...

    ...
}
```

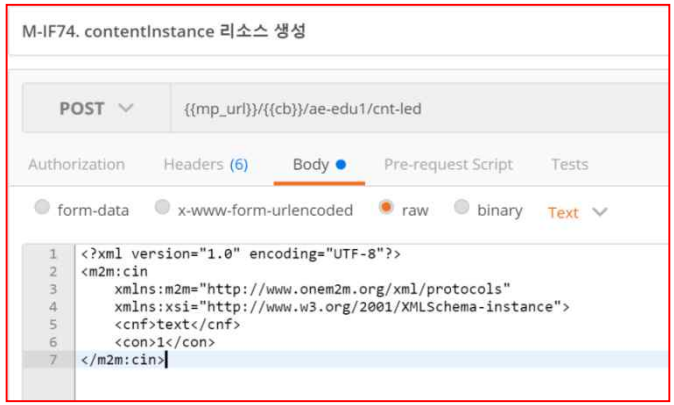
```
class ControlRequest extends Thread {
    private final Logger LOG = Logger.getLogger(ControlRequest.class.getName());
    private IReceived receiver;
    private String container_name = "cnt-led";

    public ContentInstanceObject contentinstance;
    public ControlRequest(String comm) {
        contentinstance = new ContentInstanceObject();
        contentinstance.setContent(comm);
    }
    public void setReceiver(IReceived handler) { this.receiver = handler; }
```

```
@Override
    public void run() {
        try {
            String sb = csebase.getServiceUrl() +"/" + ServiceAEName + "/" + container_name;

            URL mUrl = new URL(sb);

            HttpURLConnection conn = (HttpURLConnection) mUrl.openConnection();
            conn.setRequestMethod("POST");
            conn.setDoInput(true);
            conn.setDoOutput(true);
            conn.setUseCaches(false);
            conn.setInstanceFollowRedirects(false);

            conn.setRequestProperty("Accept", "application/xml");
            conn.setRequestProperty("Content-Type", "application/vnd.oneM2M-res+xml;ty=4");
            conn.setRequestProperty("locale", "ko");
            conn.setRequestProperty("X-M2M-RI", "12345");
            conn.setRequestProperty("X-M2M-Origin", ae.getAEid() );

            String reqContent = contentinstance.makeXML();
            conn.setRequestProperty("Content-Length", String.valueOf(reqContent.length()));

            DataOutputStream dos = new DataOutputStream(conn.getOutputStream());
            dos.write(reqContent.getBytes());
            dos.flush();
            dos.close();

            BufferedReader in = new BufferedReader(new InputStreamReader(conn.getInputStream()));

            String resp = "";
            String strLine="";
            while ((strLine = in.readLine()) != null) {
                resp += strLine;
            }
            if (resp != "") {
                receiver.getResponseBody(resp);
            }
            conn.disconnect();

        } catch (Exception exp) {
            LOG.log(Level.SEVERE, exp.getMessage());
        }
    }
}
```

12.4 Practice

App Code Review

Content Instance Creation Java Code



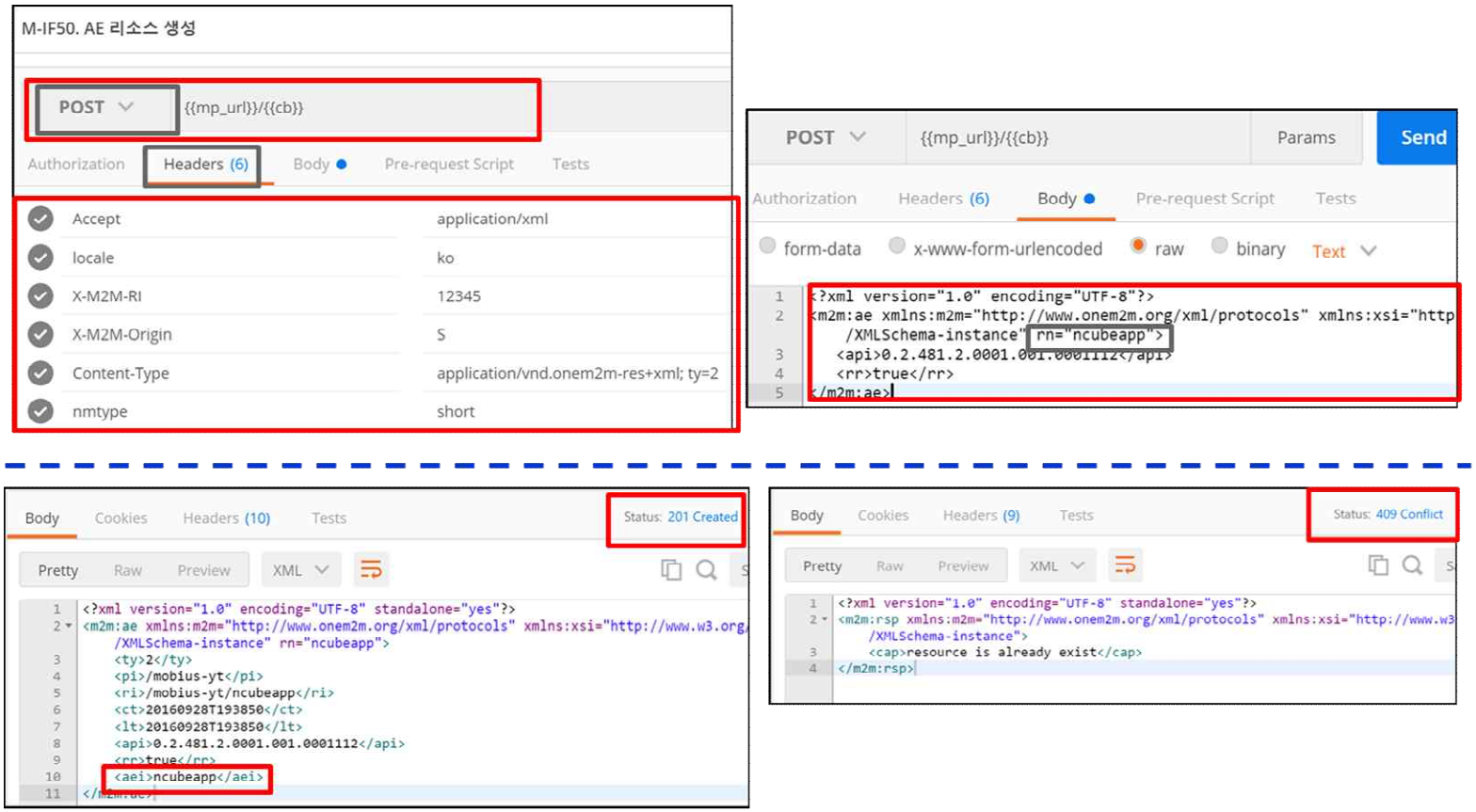

```
public class ContentInstanceObject {
    private String content = "";
    public void setContent(String contentValue) {
        this.content = contentValue; }

    public String makeXML() {
        String xml = "";

        xml += "<?xml version=₩"12.0₩" encoding=₩"UTF-8₩"?>";
        xml += "<m2m:cin ";
        xml += "xmlns:m2m=₩"http://www.oneM2M.org/xml/protocols₩" ";
        xml += "xmlns:xsi=₩"http://www.w3.org/2001/XMLSchema-instance₩">";
        xml += "<cnf>text</cnf>";
        xml += "<con>" + content + "</con>";
        xml += "</m2m:cin>";

        return xml;
    }
}
```

```
public String makeXML(){
        String xml = "";

        xml += "<?xml version=₩"12.0₩" encoding=₩"UTF-8₩"?>";
        xml += "<m2m:contentInstance ";
        xml += "xmlns:m2m="http://www.oneM2M.org/xml/protocols₩"  ";
        xml += "xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance₩">";
        xml += "<contentInfo>text</contentInfo>";
        xml += "<content>" + content + "</content>";
        xml += "</m2m:contentInstance>";

        return xml;
    }
```

12.4 Practice

App Code Review

AE Create by POSTMAN



12.4 Practice

- App Code Review
  - AE Create by Java Code

```
private String ServiceAEName = "ae-edu1";

...

public void GetAEInfo() {

    csebase.setInfo("192.168.25.47","7579","mobius-yt","1883");
    ae.setAppName("ncubeapp");

    ...

    ...
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<m2m:ae xmlns:m2m="http://www.oneM2M.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="ncubeapp">
<api>0.2.4812.2.00012.0012.0001114</api>
<rr>true</rr>
</m2m:ae>
```

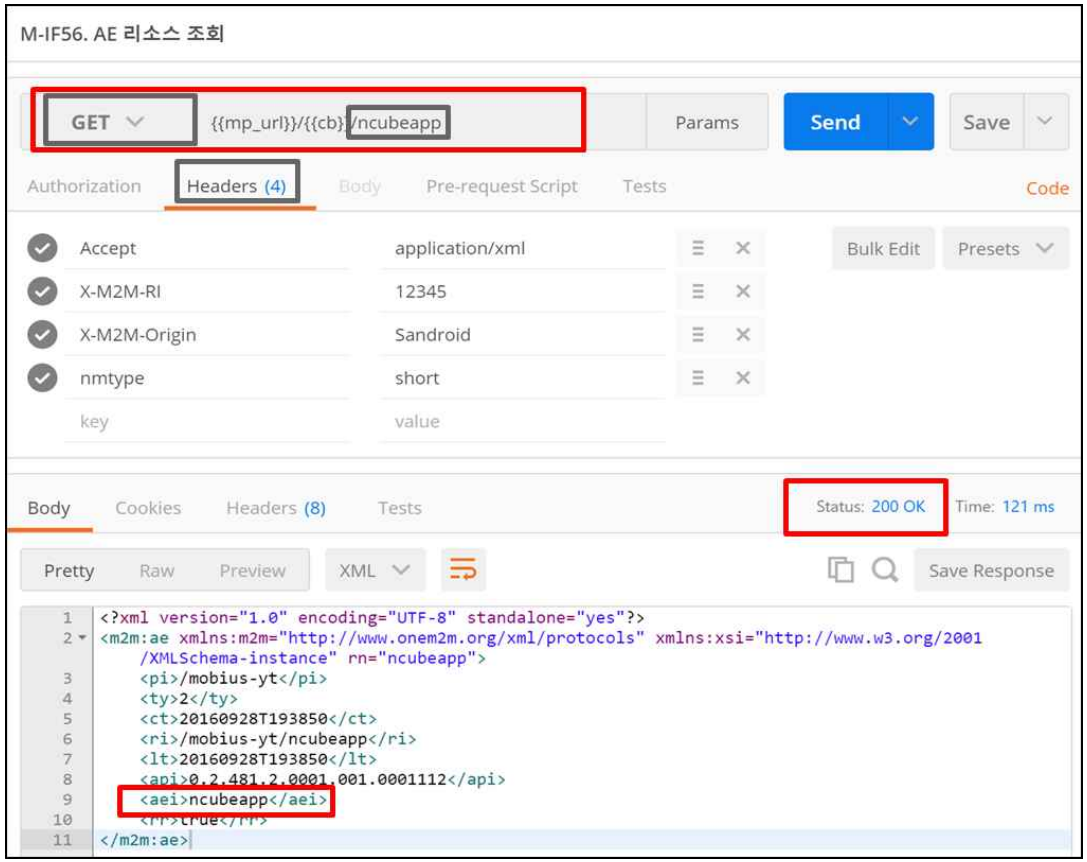
```
Class aeCreateRequest extends Thread {
    private final Logger LOG = Logger.getLogger(aeCreateRequest.class.getName());
    String TAG = aeCreateRequest.class.getName();
    private IReceived receiver;
    int responseCode=0;
    public ApplicationEntityObject applicationEntity;
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }
    public aeCreateRequest(){
        applicationEntity = new ApplicationEntityObject();
        applicationEntity.setResourceName(ae.getappName());
    }
```

```
public void run() {
    try {
        String sb = csebase.getServiceUrl();
        URL mUrl = new URL(sb);
        HttpURLConnection conn = (HttpURLConnection) mUrl.openConnection();
        conn.setRequestMethod("POST");
        conn.setDoInput(true);
        conn.setDoOutput(true);
        conn.setUseCaches(false);
        conn.setInstanceFollowRedirects(false);
        conn.setRequestProperty("Content-Type", "application/vnd.oneM2M-res+xml;ty=2");
        conn.setRequestProperty("Accept", "application/xml");
        conn.setRequestProperty("locale", "ko");
        conn.setRequestProperty("X-M2M-Origin", "S");
        conn.setRequestProperty("X-M2M-RI", "12345");
        conn.setRequestProperty("X-M2M-NM", ae.getappName() );
        String reqXml = applicationEntity.makeXML();
        conn.setRequestProperty("Content-Length", String.valueOf(reqXml.length()));
        DataOutputStream dos = new DataOutputStream(conn.getOutputStream());
        dos.write(reqXml.getBytes());
        dos.flush();
        dos.close();
        responseCode = conn.getResponseCode();
        BufferedReader in = null;
        String aei = "";
        if (responseCode == 201) {
            // Get AEID from Response Data
            in = new BufferedReader(new InputStreamReader(conn.getInputStream()));
            String resp = "";
            String strLine;
            while ((strLine = in.readLine()) != null) {
                resp += strLine;
            }
            ParseElementXml pxml = new ParseElementXml();
            aei = pxml.GetElementXml(resp, "aei");
            ae.setAEid( aei );
            Log.d(TAG, "Create Get AEID[" + aei + "]");
            in.close();
        }
        if (responseCode != 0) {
            receiver.getResponseBody( Integer.toString(responseCode) );        }
        conn.disconnect();
    } catch (Exception exp) {
        LOG.log(Level.SEVERE, exp.getMessage());
    }
```

```
<?xml version="1.0" encoding="UTF-8" s
<m2m:ae xmlns:m2m="http://www.onem2m.o
    -instance" rn="ncubeapp">
    <pi>/mobius-yt</pi>
    <ty>2</ty>
    <ct>20160928T193850</ct>
    <ri>/mobius-yt/ncubeapp</ri>
    <lt>20160928T193850</lt>
    <api>0.2.481.2.0001.001.0001112</a
    <aei>ncubeapp</aei>
    <rr>true</rr>
</m2m:ae>
```

12.4 Practice

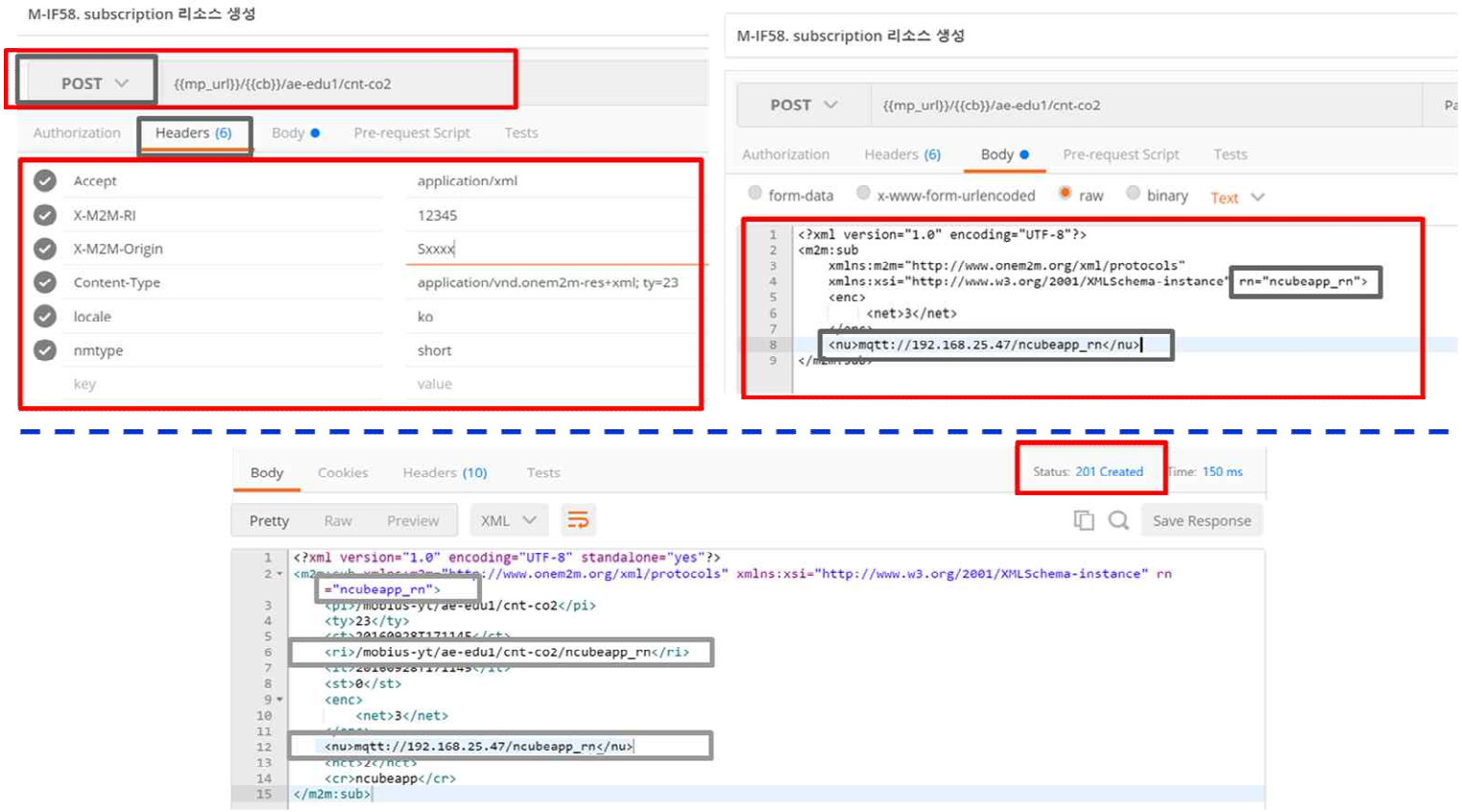
- App Code Review
  - AE Retrieve POSTMAN [AE-ID]



12.4 Practice

App Code Review

Subscription Creation by POSTMAN



12.4 Practice

App Code Review

Subscription Creation by Java Code

```
private String ServiceAEName = "ae-edu1";

public void GetAEInfo() {

    csebase.setInfo("192.168.25.47","7579","mobius-yt","1883");
    ae.setAppName("ncubeapp");
    ...
}

/* MQTT Subscription */
public void MQTT_Create(boolean mtqqStart) {
    if (mtqqStart && mqttClient == null) {
        /* Subscription Resource Create to Yellow Turtle */
        SubscribeResource subcribeResource = new SubscribeResource();
        subcribeResource.setReceiver(new IReceived() {
            public void getResponseBody(final String msg) {
                handler.post(new Runnable() {
                    public void run() {
                        ...
                    }
                });
            }
        });
        subcribeResource.start();
        ...
}
 /* Subscribe Co2 Content Resource */
class SubscribeResource extends Thread {
    private IReceived receiver;
    private String container_name = "cnt-co2"; //change to control container name

    public ContentSubscribeObject subscribeInstance;
    public SubscribeResource() {
        subscribeInstance = new ContentSubscribeObject();
        subscribeInstance.setUrl(csebase.getHost());
        subscribeInstance.setResourceName(ae.getAEid()+"_rn");
        subscribeInstance.setPath(ae.getAEid()+"_sub");
        subscribeInstance.setOrigin_id(ae.getAEid());
    }
    public void setReceiver(IReceived handler) { this.receiver = handler; }
```

```
private String ServiceAEName = "ae-edu1";

public void GetAEInfo() {

    csebase.setInfo("192.168.25.47","7579","mobius-yt","1883");
    ae.setAppName("ncubeapp");
    ...
}

/* MQTT Subscription */
public void MQTT_Create(boolean mtqqStart) {
    if (mtqqStart && mqttClient == null) {
        /* Subscription Resource Create to Yellow Turtle */
        SubscribeResource subcribeResource = new SubscribeResource();
        subcribeResource.setReceiver(new IReceived() {
            public void getResponseBody(final String msg) {
                handler.post(new Runnable() {
                    public void run() {
                        ...
                    }
                });
            }
        });
        subcribeResource.start();
        ...
}

 /* Subscribe Co2 Content Resource */
class SubscribeResource extends Thread {
    private IReceived receiver;
    private String container_name = "cnt-co2"; //change to control container name

    public ContentSubscribeObject subscribeInstance;
    public SubscribeResource() {
        subscribeInstance = new ContentSubscribeObject();
        subscribeInstance.setUrl(csebase.getHost());
        subscribeInstance.setResourceName(ae.getAEid()+"_rn");
        subscribeInstance.setPath(ae.getAEid()+"_sub");
        subscribeInstance.setOrigin_id(ae.getAEid());
    }
    public void setReceiver(IReceived handler) { this.receiver = handler; }
```

```
<?xml version="1.0" encoding="UTF-8"?>
<m2m:sub
    xmlns:m2m="http://www.onem2m.org/xml/protocols"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="ncubeapp_rn">
    <enc>
        <net>3</net>
    </enc>
    <nu>mqtt://192.168.25.47/ncubeapp_rn</nu>
</m2m:sub>
```

12.4 Practice

App Code Review

MQTT Client Subscription, Listener, Callback, Response Java Code

```
private String ServiceAEName = "ae-edu1";

public void GetAEInfo() {

    csebase.setInfo("192.168.25.47","7579","mobius-yt","1883");
    ae.setAppName("ncubeapp");
    ...
    MQTT_Req_Topic = "/oneM2M/req/:mobius-yt/"+ae.getAEid()+"_sub"+"/xml";
    MQTT_Resp_Topic = "/oneM2M/resp/:mobius-yt/"+ae.getAEid()+"_sub"+"/xml";
}

/* MQTT Subscription */
public void MQTT_Create(boolean mtqqStart) {
    if (mtqqStart && mqttClient == null) {
        /* Subscription Resource Create to Yellow Turtle */
        SubscribeResource subcribeResource = new SubscribeResource();
        subcribeResource.setReceiver(new IReceived() {
        ...
        subcribeResource.start();

        /* MQTT Subscribe */
        mqttClient = new MqttAndroidClient(this.getApplicationContext(), "tcp://" + csebase.getHost() + ":" +
csebase.getMQTTPort(), MqttClient.generateClientId());
        mqttClient.setCallback(mainMqttCallback);
        try {
            IMqttToken token = mqttClient.connect();
            token.setActionCallback(mainIMqttActionListener);
        } catch (MqttException e) {
            e.printStackTrace();
        }
    } else {
        /* MQTT unSubscribe or Client Close */
        mqttClient.setCallback(null);
        mqttClient.close();
        mqttClient = null;
    }
}
```

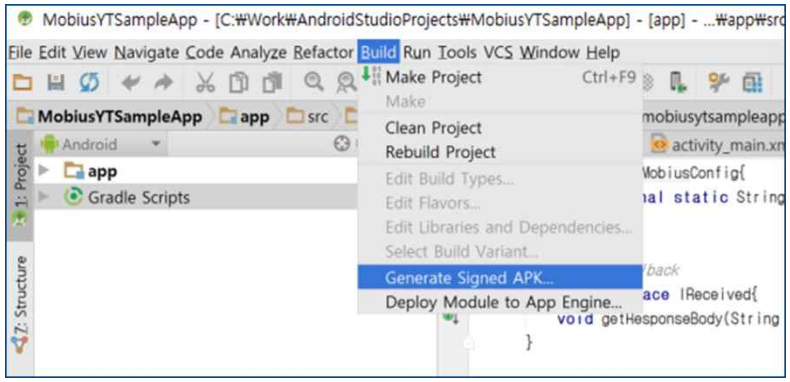
```
/* MQTT Listener */
private IMqttActionListener mainIMqttActionListener = new IMqttActionListener() {
    @Override
    public void onSuccess(IMqttToken asyncActionToken) {
        String payload = "";
        int mqttQos = 1; /* 0: NO QoS, 1: No Check , 2: Each Check */

        MqttMessage message = new MqttMessage(payload.getBytes());
        try {
            mqttClient.subscribe(MQTT_Req_Topic, mqttQos);
        } catch (MqttException e) {
            e.printStackTrace();
        }
    }
    ...
};
/* MQTT Broker Message Received */
private MqttCallback mainMqttCallback = new MqttCallback() {
    @Override
    public void connectionLost(Throwable cause) {  Log.d(TAG, "connectionLost"); }
    @Override
    public void messageArrived(String topic, MqttMessage message) throws Exception {
        Log.d(TAG, "messageArrived");
        textViewData.setText("");
        textViewData.setText("**** MQTT CO2 실시간 조회 ****₩r₩n₩r₩n" + message.toString().replaceAll("₩t|₩n", ""));
        ArrayList<String> mqttMessage = new ArrayList<String>();
        mqttMessage = MqttClientRequestParser.notificationParse(message.toString());
        String responseMessage = MqttClientRequest.notificationResponse(mqttMessage);

        /* Make xml for MQTT Response Message */
        MqttMessage resmessage = new MqttMessage(responseMessage.getBytes());

        try {
            mqttClient.publish(MQTT_Resp_Topic, resmessage);
        } catch (MqttException e) { e.printStackTrace(); }
    }
    @Override
    public void deliveryComplete(IMqttDeliveryToken token) {
        Log.d(TAG, "deliveryComplete");
    }
};
```

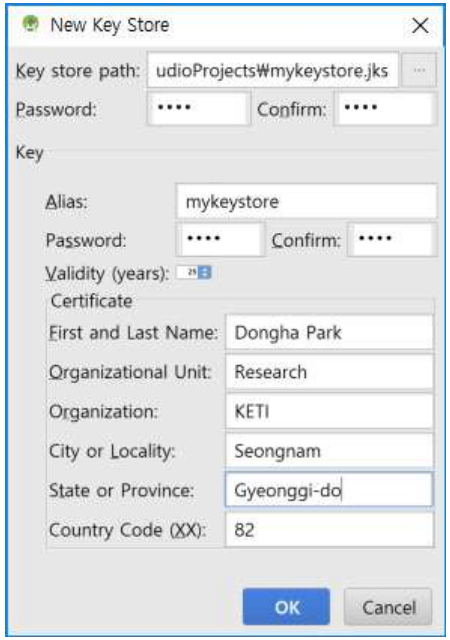
12.4 Practice

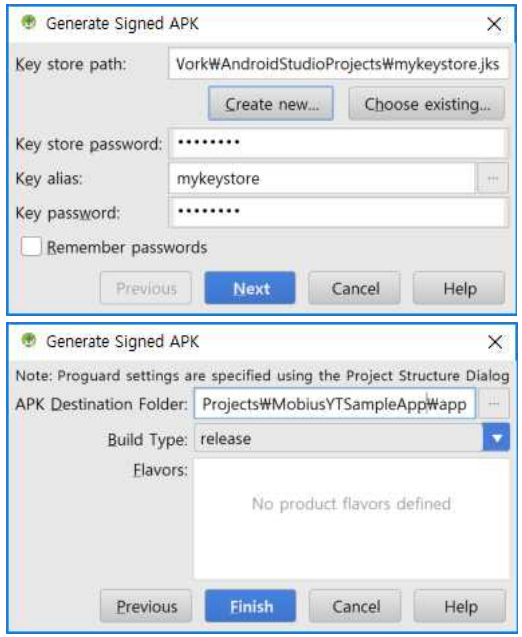
▪ App Code Review

▪ Android Studio Generate APK

- Build –〉 Generate Signed APK...
- If have Key store, Select “Choose existing”
- If do not have Key store, Select “Create new”
- Save location and input file name, Password, Alias, Certification information.
- Select key file
- Locate the APK path.





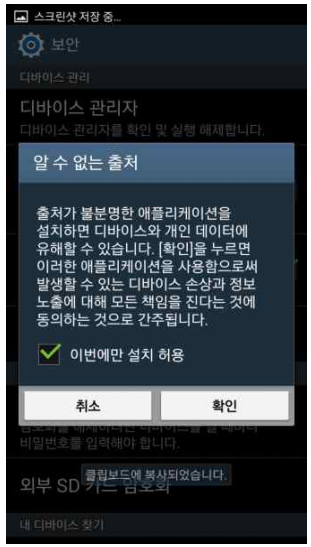


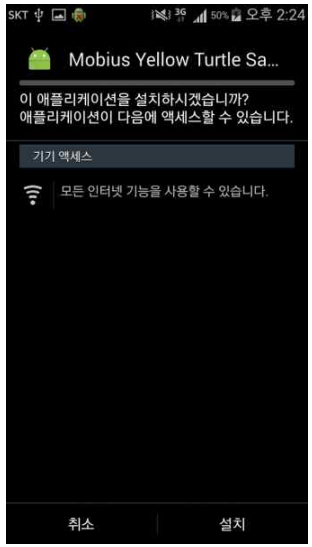
12.4 Practice

▪ App Code Review

▪ Android Studio Generate APK Install

- Move the APK file to Android Phone Device
- Use the File Manager to locate the APK file and install it





Assignment

- Assignment 12

실습 예제를 활용하여 Co2수치에 따라서 LED를
on/off 시키는 서비스 어플리케이션을 **개발해 보시오.**

Mobius 플랫폼을 응용하여 개발 할 수 있는 서비스 예제를
구상해 보시오.

Reference

[12.1] Run &Cube and TAS

&Cube configuration

그림 출처: http://www.iotocean.org/main/

[12.2] Extension

SK ThingPlug (Jun. 2015)

그림 출처: https://sandbox.sktiot.com/

Busan Smart City

그림 출처: http://www.iotocean.org/main/

ConnecThing Demo

동영상 출처1 : https://youtu.be/FngoyncRhCM

동영상 출처2 : https://youtu.be/zHfIQwLBPsM

13장. oneM2M 플랫폼기반 서비스 응용 예제 실습

Chapter 13. oneM2M Platform based Service Application

13.1 서비스 개요 & 요구 하드웨어
13.2 Raspberry Pi 센서 연동
13.3 TAS 작성 & Mobius 서버 연동

- Assignment
- Reference

강의 개요

- **강의 목표**

  Raspberry Pi 와 센서 디바이스간의 연동 방법을 배우고

  TAS 코드를 분석하여 센서 디바이스 컨트롤을 학습한다.

- **강의 내용**

  - 요구 하드웨어 확인
  - 서비스 모델 계획
  - Raspberry Pi 와 센서 디바이스를 연동
  - TAS 코드 분석
  - 센서 디바이스 조회

13.1 서비스 개요 & 요구 하드웨어

- 상용 서비스중인 B사 스마트 플러그

- 스마트폰을 사용한 스마트 플러그 제어 서비스
- 샘플코드를 사용하여 간단한 스마트 플러그 서비스 구현
- 목표 구현 기능
  - 스마트플러그 원격 상태 확인
  - 원격 전원 온/오프 제어

13.1 서비스 개요 & 요구 하드웨어

Raspberry Pi2 model B

Comm. Module

Device

- Main Device – Raspberry Pi2 model B
- Module – KETI Smart Plug
- Comm.Module – TI CC2530, CP2103 USB-to-Serial

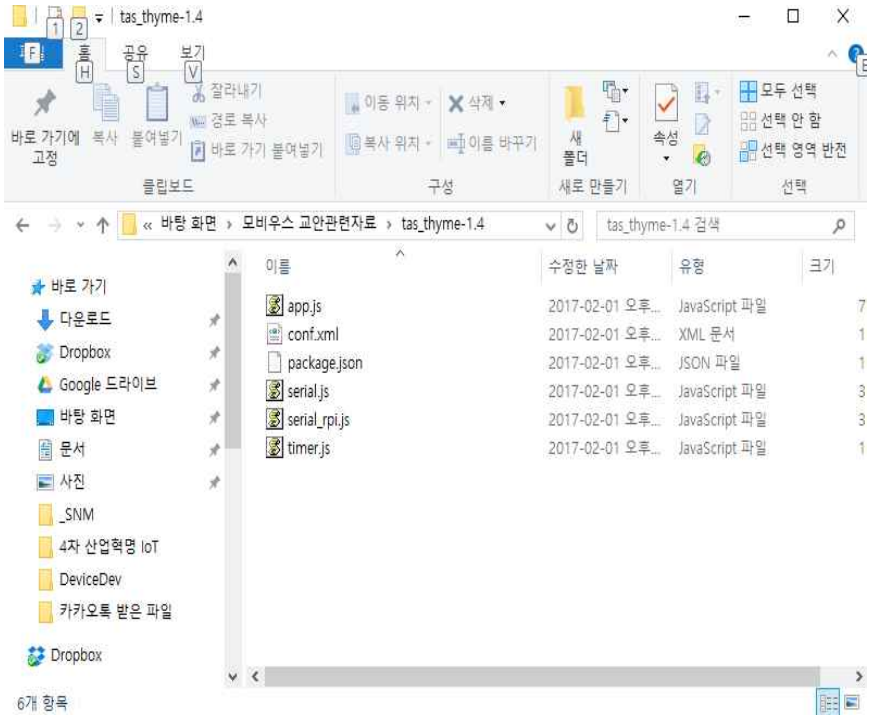
13.2 Raspberry Pi

- 모듈연결 구조

- 위 사진과 같이 통신 모듈을 Raspberry Pi에 USB로 연결
- 스마트 플러그는 통신 모듈과 무선(ZigBee) RF 무선 통신

13.3 TAS 작성 & Mobius 서버 연동

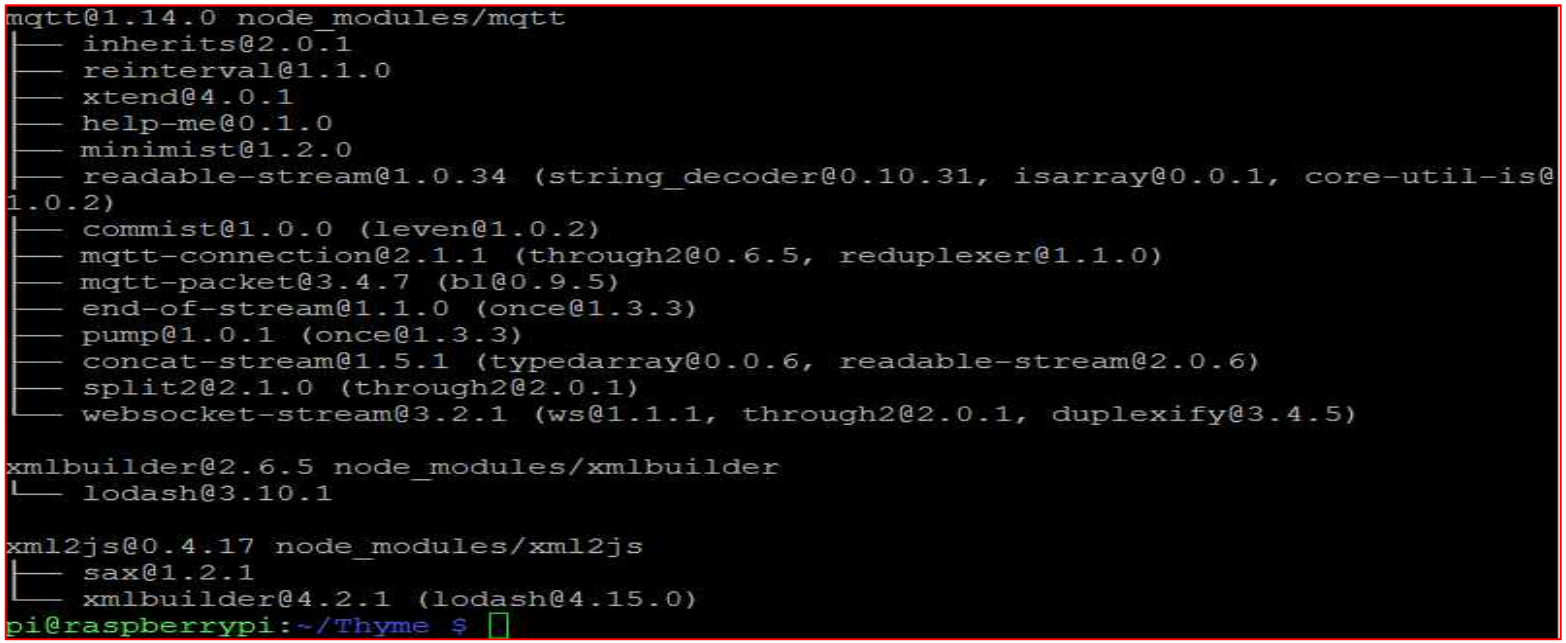
- Node.js TAS 샘플코드 파일 리스트



- 동작기능
  - 모듈로부터 데이터 수신 대기
  - 주기적으로 수신된 데이터 전송
  - 사용자에 따른 온/오프 제어
- App.js : 메인 동작 코드
- serial.js : 통신모듈 연동 코드
- Conf.xml : 각 종 설정 값
- Package.json : 라이브러리 리스트 설치 설정 파일

13.3 TAS 작성 & Mobius 서버 연동

- 라이브러리 설치 성공 화면



- node/thyme_tas 디렉토리로 이동하여 npm을 통해 라이브러리 설치
- sudo npm install

13.3 TAS 작성 & Mobius 서버 연동

- 기본 셋업 단계

〈app.js〉

〈중략〉
```
var sh_timer = require('./timer');
var sh_serial = require('./serial');
```
〈생략〉

〈serial.js〉
```
var util = require('util');
var os = require('os');
var serialport = require('serialport');

var SerialPort;
var myPort;

exports.open = function(portname, baudrate) {
    SerialPort = serialport.SerialPort;

    myPort = new SerialPort(portname, {
        baudRate : baudrate,
        buffersize : 1
        //parser : serialport.parsers.readline("₩r₩n")
    });

    myPort.on('open', showPortOpen);
    myPort.on('data', saveLastestData);
    myPort.on('close', showPortClose);
    myPort.on('error', showError);
};
```

- 통신 모듈 연동을 위한 serial.js 코드 활용
- 통신 모듈로부터 데이터를 수신하고 제어 메시지를 전송하기 위한 라이브러리 로드
- Serial 포트 활용 통신 모듈 연결을 위한 초기화

13.3 TAS 작성 & Mobius 서버 연동

- 스마트 플러그 제어

〈app.js〉

```
upload_client.on('data', function(data) {

                    〈중략〉

for (j = 0; j < download_arr.length; j++) {
                        if (download_arr[j].ctname == sink_obj.ctname) {
                            cin = JSON.stringify({id: download_arr[i].id, con:
sink_obj.con});
                            sh_serial.g_down_buf = cin;
                            sh_serial.serial_event.emit('down');
                            break;
                        }
                    }

                    〈후략〉
```

〈serial.js〉

```
exports.serial_event.on('down', function () {
    console.log(exports.g_down_buf);
    myPort.write(exports.g_down_buf);
});
```

- 컨테이너 네임의 유효성 판단
- 사용자의 메시지를 통해 Serial 코드 호출로 제어명령 전송

13.3 TAS 작성 & Mobius 서버 연동

▪ 스마트 플러그 데이터 수신 및 업로드 (1/2)

〈serial.js〉

```
function saveLastestData(data) {
    if (data == '{') {
        g_sink_buf_index = 0;
        g_sink_buf_start = 1;
        pre_c = '';
        cur_c = util.format('%s', data);
        g_sink_buf = pre_c + cur_c;
        pre_c = g_sink_buf;
    }
    else if (data ==  '}') {
        cur_c = util.format('%s', data);
        g_sink_buf = pre_c + cur_c;
        pre_c = g_sink_buf;

        exports.g_sink_buf = g_sink_buf;
        exports.serial_event.emit('up');
        exports.myPort = myPort;

        g_sink_buf_start = 0;
        g_sink_ready = 1;
    }
    else if (g_sink_buf_start == 1) {
        cur_c = util.format('%s', data);
        g_sink_buf = pre_c + cur_c;
        pre_c = g_sink_buf;
    }
}
```

- 통신모듈로부터 스마트 플러그 데이터 수신 시 데이터 파싱을 통해 업로드 데이터 생성

13.3 TAS 작성 & Mobius 서버 연동

▪ 스마트 플러그 데이터 수신 및 업로드 (2/2)

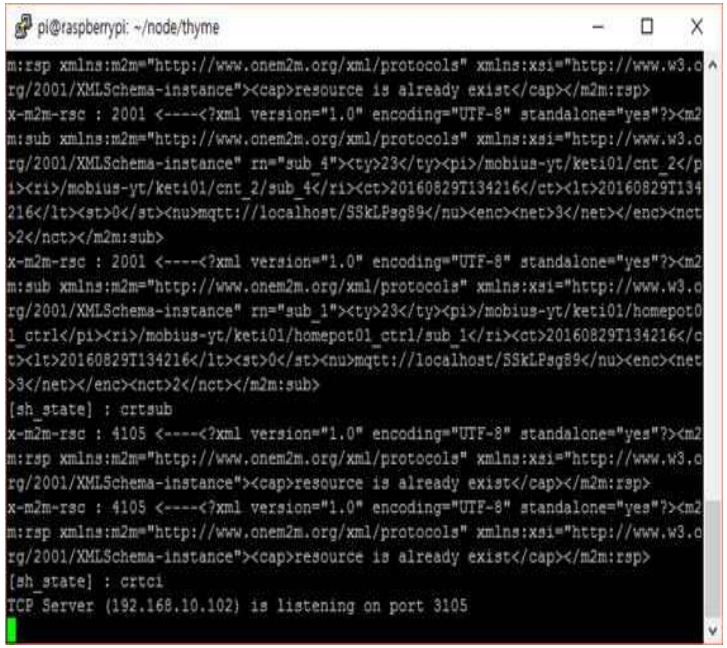
〈app.js〉

〈 중략 〉

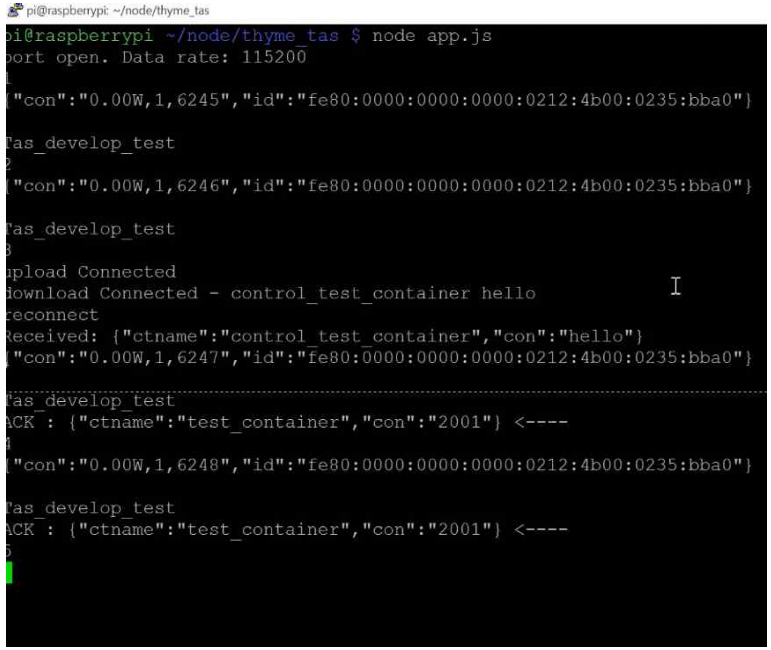
```
sh_serial.serial_event.on('up', function () {
    if(tas_state == 'upload') {
        console.log(sh_serial.g_sink_buf);

        // parsing sensor data, manage id according with ctname
        var sink_str = util.format('%s', sh_serial.g_sink_buf);
        var sink_obj = JSON.parse(sink_str);

        for(var i = 0; i < upload_arr.length; i++) {
            if(upload_arr[i].id == sink_obj.id) {
                var cin = {ctname: upload_arr[i].ctname, con: sink_obj.con};
                upload_client.write(JSON.stringify(cin));
                break;
            }
        }
    }
});
```

- 서버로 업로드 시 ctname에 해당하는 트리 구조에 스마트 플러그 정보 저장

13.3 TAS 작성 & Mobius 서버 연동



nCube 동작화면

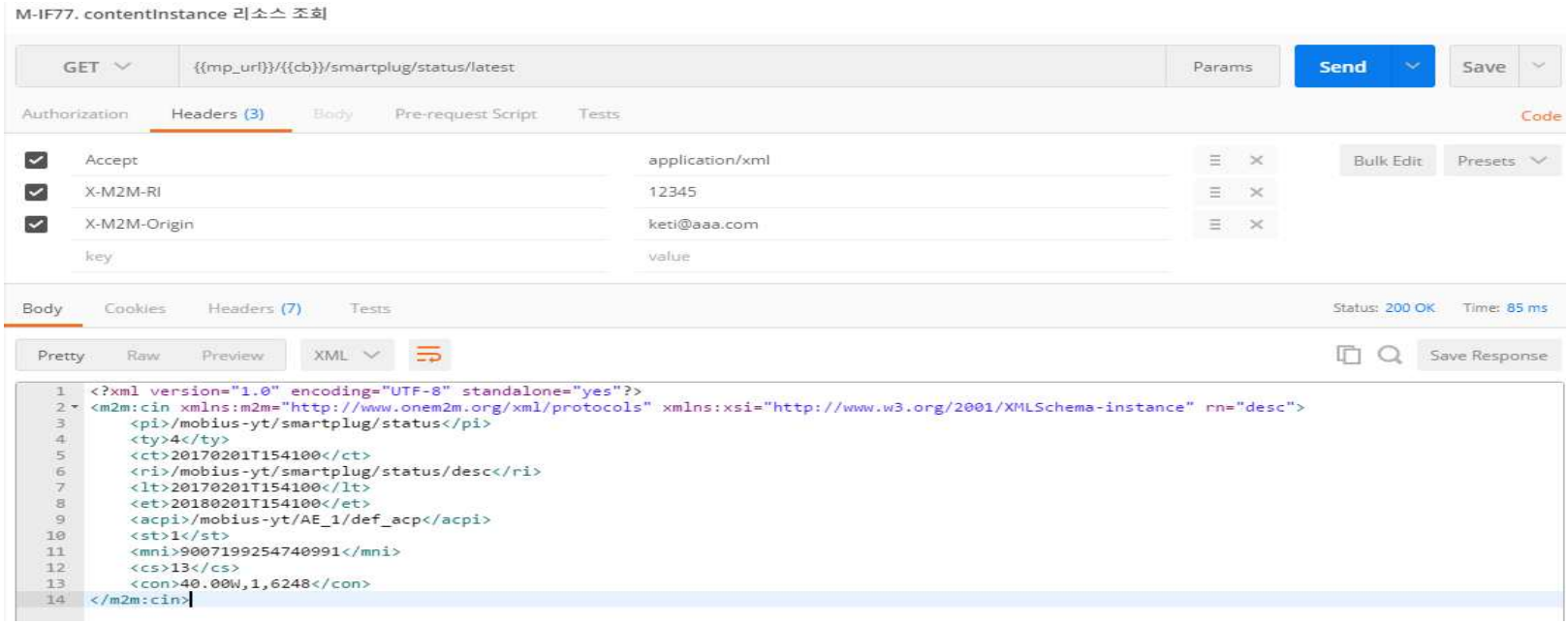


TAS 동작화면

- Node/thyme 디렉토리 내에서 nCube 실행 (sudo node thyme.js)
- Node/thyme_tas 디렉토리의 샘플코드 원본 또는 수정본 TAS 실행 (sudo node app.js)
- Putty 화면 내에 정상 동작 확인

13.3 TAS 작성 & Mobius 서버 연동

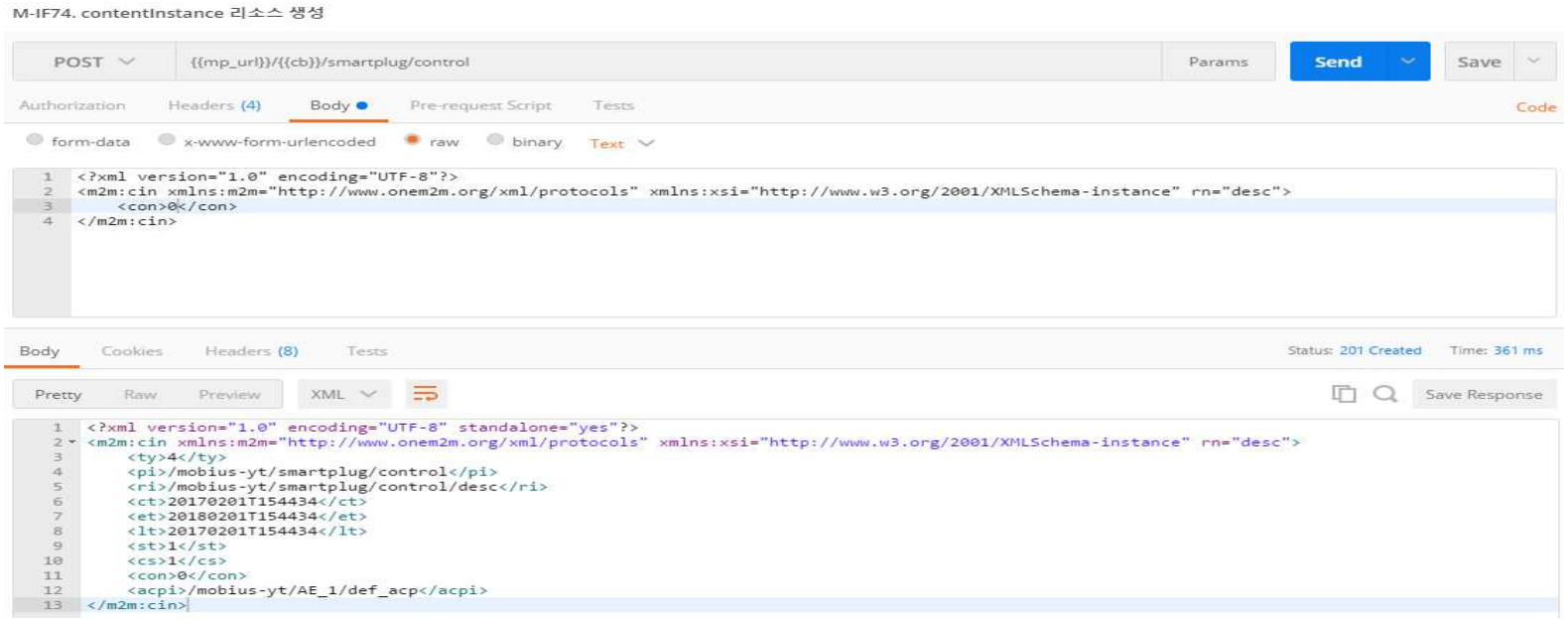
- Postman REST 조회 과정



- Mobius 서버와의 연동을 정상 동작 확인
- http://Mobius 서버주소/mobius-yt/스마트 플러그 조회 폴더트리/latest
- Chrome의 postman 앱을 통해 스마트 플러그 정보 조회

13.3 TAS 작성 & Mobius 서버 연동

- Postman REST 제어 과정



- Mobius 서버와의 연동을 정상 동작 확인
- http://Mobius 서버주소/mobius-yt/스마트플러그 제어 폴더트리
- Chrome의 postman 앱을 통해 스마트 플러그 제어

Assignment

- Assignment 13

제시된 TAS code를 사용하여 스마트 플러그 정보를 조회하기 위한
안드로이드 응용 서비스 어플리케이션을 개발해 보시오.

Reference

[13.1] 서비스 개요 & 요구 하드웨어

상용 서비스중인 B사 스마트 플러그 P5

그림 출처: http://www.belkin.com/kr/Products/c/WSPWR/

14장. oneM2M 플랫폼 기반 서비스 응용 예제 실습 II

Chapter 14. oneM2M Platform based Service Application Example II

14.1 서비스 개요 & 요구 하드웨어
14.2 Raspberry Pi 센서 연동
14.3 TAS 작성 & Mobius 서버 연동

- Assignment
- Reference

강의 개요

▪ 강의 목표

Raspberry Pi 와 센서 디바이스간의 연동 방법을 배우고

TAS 코드를 분석하여 센서 디바이스 컨트롤을 학습한다.

▪ 강의 내용

- 요구 하드웨어 확인
- 서비스 모델 계획
- Raspberry Pi 와 센서 디바이스 연동
- TAS 코드 분석
- 센서 디바이스 조회

14.1 서비스 개요 & 요구 하드웨어

- 상용 서비스 중인 L사 가스락(Gas Lock)

- 스마트폰을 사용한 가스밸브 제어 서비스
- 샘플코드를 사용하여 간단한 스마트 가스락 서비스 (Smart Gas Lock Service) 구현
- 목표 구현 기능
  - 가스밸브 원격 확인/잠금
  - 외출 시 경고 알림

14.1 서비스 개요 & 요구 하드웨어

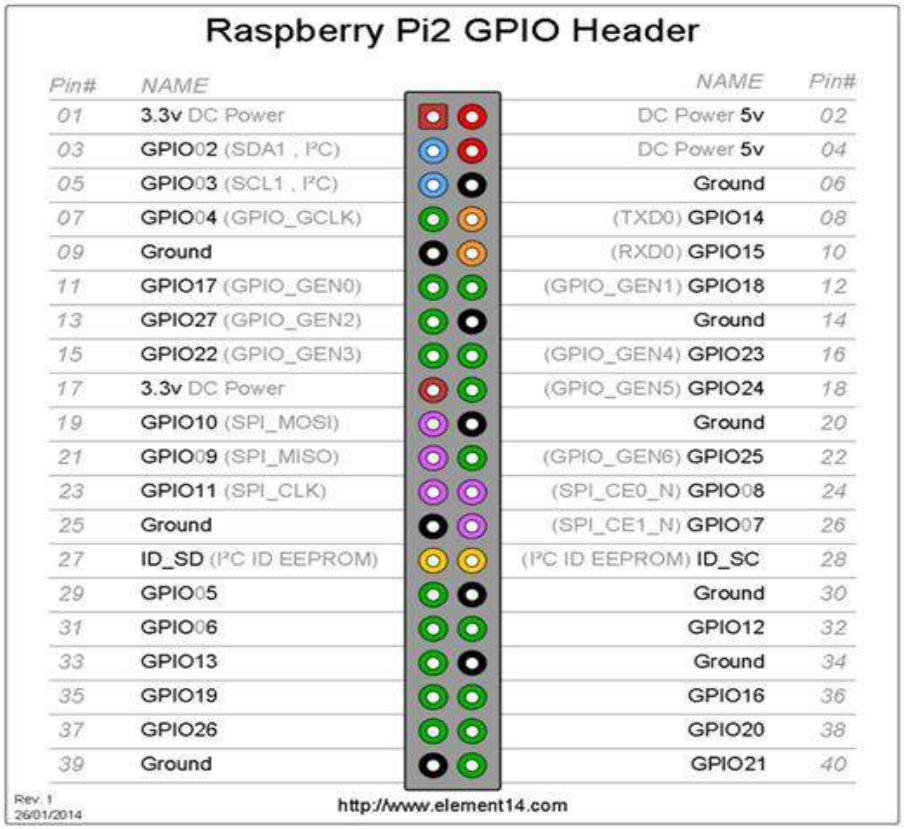
Raspberry Pi2 model B

Main Module

- Main Device – Raspberry Pi2 model B
- Module – Mini Servo Motor (SG-90), Gas Sensor (MQ-6), ADC(PCF8591), Buzzer
- ETC – breadboard, wire, 5V power×2, UTP cable, I2C connector

14.2 Raspberry Pi 센서 연동

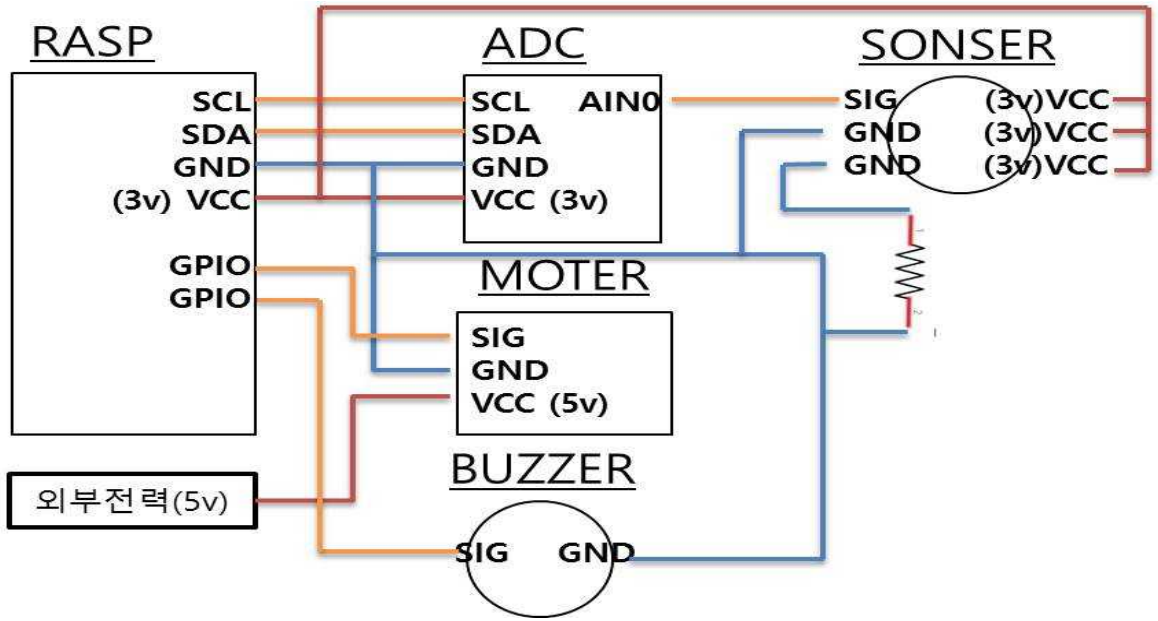
- RaspberryPi2 GPIO Header



- Mini Servo Motor
  - VCC
  - GND
  - SIG
- Gas Sensor
  - VCC
  - GND
  - SIG
- ADC
  - VCC
  - GND
  - SCL
  - SDA
- Buzzer

14.2 Raspberry Pi 센서 연동

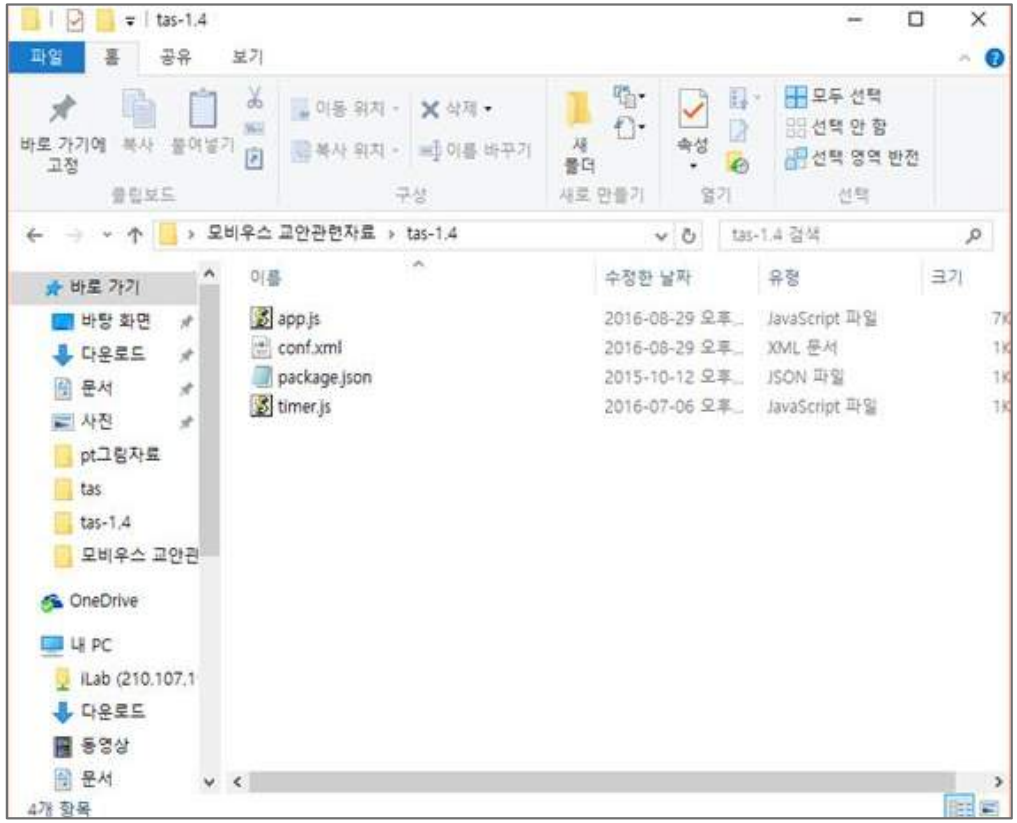
- 모듈 연결 회로도



- 위 그림과 같이 각 모듈을 Raspberry Pi에 연결, 모터는 별도의 외부 전원(5V) 사용
- Raspberry Pi – ADC – Gas Sensor간의 연결 (I2C 통신)
- Raspberry Pi – Motor, Raspberry Pi – Buzzer 연결 (GPIO 통신)

14.3 TAS 작성 & Mobius 서버 연동

- Node.js TAS 샘플 코드 파일 리스트

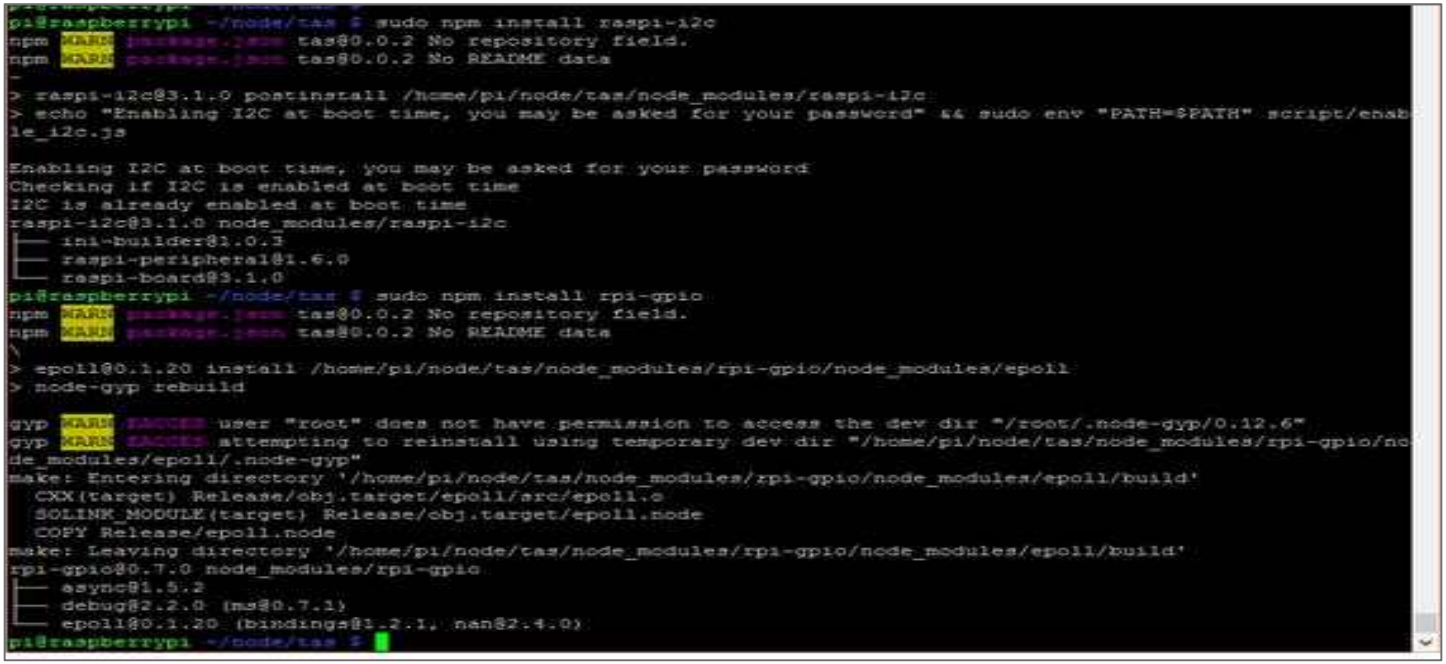


- 동작기능
  - 2초 주기로 가스 값 조회
  - 사용자에 따른 모터 제어
  - 가스 값에 따른 버저 제어

- App.js : 메인 동작 코드
- Conf.xml : 각 종 설정 값
- Package.json : 라이브러리 리스트 설치 설정 파일

14.3 TAS 작성 & Mobius 서버 연동

- 라이브러리 설치 성공 화면



- node/tas 디렉토리로 이동하여 npm를 통해 raspi, raspi-i2c, raspi-pwm, rpi-gpio 설치
- sudo npm install raspi; sudo npm install raspi-i2c;
- sudo npm install rpi-gpio; sudo npm install raspi-pwm;

14.3 TAS 작성 & Mobius 서버 연동

- 기본 셋업 단계

```
var net = require('net');
var util = require('util');
var fs = require('fs');
var xml2js = require('xml2js');
var gpio = require('rpi-gpio');
var raspi = require('raspi');
var PWM = require('raspi-pwm').PWM;
var I2C = require('raspi-i2c').I2C;
var i2c;
var pwm;

var sh_timer = require('./timer');

var usecomport = '';
var usebaudrate = '';
var useparentport = '';
var useparenthostname = '';

var upload_arr = [];
var download_arr = [];
raspi.init(function() {
                    i2c = new I2C();
                    pwm = new PWM();
});
```

- 센서 모듈을 제어 및 조회하기 위한 라이브러리 로드
- 가스 센서의 I2C 및 모터 모듈의 PWM 셋업

14.3 TAS 작성 & Mobius 서버 연동

▪ 부저 셋업 및 컨트롤

```
fs.readFile('conf.xml', 'utf-8', function (err, data) {
                    〈 중 략 〉
                }
                gpio.setup(16, gpio.DIR_OUT, write);

            }
        });
    }
});

function write(mode) {

                if(mode == true) {
                    gpio.write(16, true, function(err) {
                        if (err) throw err;
                        console.log('[JW LOG] Buzzer HIGH');
                    });
                } else {
                                gpio.write(16, false, function(err) {
                        if (err) throw err;
                        console.log('[JW LOG] Buzzer LOW');
                    });
                }
}
```

- GPIO 핀을 사용하여 부저 핀을 셋업
- 셋업된 콜백함수 정의
- Mode 변수를 사용하여 부저의 on, off 값을 제어

14.3 TAS 작성 & Mobius 서버 연동

- Servo 모터 제어

```
upload_client.on('data', function(data) {
                    〈중략〉

// Mobius -〉 nCube -〉 Tas (downlink message)
 for (j = 0; j 〈 download_arr.length; j++) {
    if (download_arr[j].ctname == sink_obj.ctname) {
       cin = JSON.stringify({id: download_arr[i].id, con: sink_obj.con});
        if(sink_obj.con == 'lock') { // control message
                pwm.write(180);
                console.log("[JW LOG]lock control message received - "
                            + sink_obj.con);
         } else if(sink_obj.con == 'ulck') {
               pwm.write(10);
               console.log("[JW LOG]lock control message received -  "
                            + sink_obj.con);
            }
            break;
          }
        }
      }
    }
  }
});
```

- 컨테이너 네임의 유효성 판단
- 사용자의 메시지인 ‘lock’, ‘ulck’를 통해 Servo 모터를 제어

14.3 TAS 작성 & Mobius 서버 연동

▪ 가스 조회 및 업로드 (1/2)

```
sh_timer.timer.on('tick', function() {
    tick_count++;
    if((tick_count % 2) == 0) {
        if (tas_state == 'upload') {
                var gasValue = i2c.readByteSync(0x48);
                var con = '[GasSensing]' + gasValue;
                var leakStatus = false;
                if(gasValue > 200) {
                        write(true);
                        pwm.write(180);
                        leakStatus = true;
                } else {
                        write(false);
                        pwm.write(10);
                }
            for (var i = 0; i < upload_arr.length; i++) {
                if(leakStatus == true) {
                        if (upload_arr[i].ctname == 'warn') {
                                con = '[LeakWarn]leak'
                    var cin = {ctname: upload_arr[i].ctname, con: con};
                    console.log(JSON.stringify(cin) + ' ----->');
                    upload_client.write(JSON.stringify(cin));
                    break;
                }
< 계속 >
```

- 2초마다 I2C 통신을 통하여 가스 센서 값 조회 후 서버로 업로드
- 가스 센서 값이 200을 넘어가면 자동으로 Servo Motor를 동작시켜 잠금

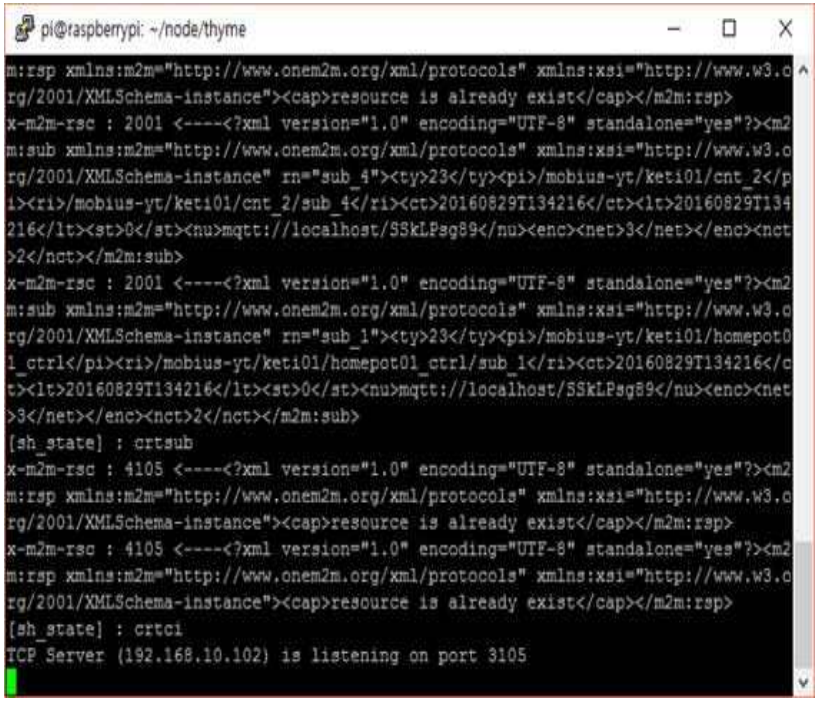
14.3 TAS 작성 & Mobius 서버 연동

- 가스 조회 및 업로드 (1/2)

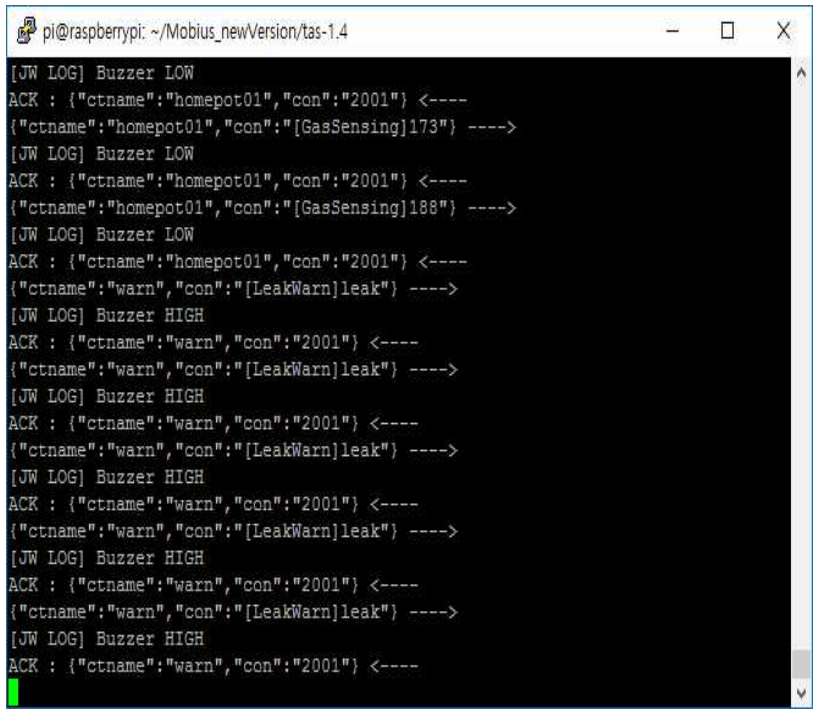
```
                } else {
                        if (upload_arr[i].ctname == 'homepot01') {
                    var cin = {ctname: upload_arr[i].ctname, con: con};
                    console.log(JSON.stringify(cin) + ' ----->');
                    upload_client.write(JSON.stringify(cin));
                    break;
                }
                }

            }
        }
    }

    if((tick_count % 3) == 0) {
        if(tas_state == 'connect' || tas_state == 'reconnect') {
            upload_client.connect(useparentport, useparenthostname,
function() {
                console.log('upload Connected');
                tas_man_count = 0;
                for (var i = 0; i < download_arr.length; i++) {
                    console.log('download Connected - ' +
download_arr[i].ctname + ' hello');
                    var cin = {ctname: download_arr[i].ctname, con: 'hello'};
                    upload_client.write(JSON.stringify(cin));
                }
```

- 서버로 업로드 시 homepot01의 트리 구조에 가스 센서 값 저장

14.3 TAS 작성 & Mobius 서버 연동



nCube 동작화면

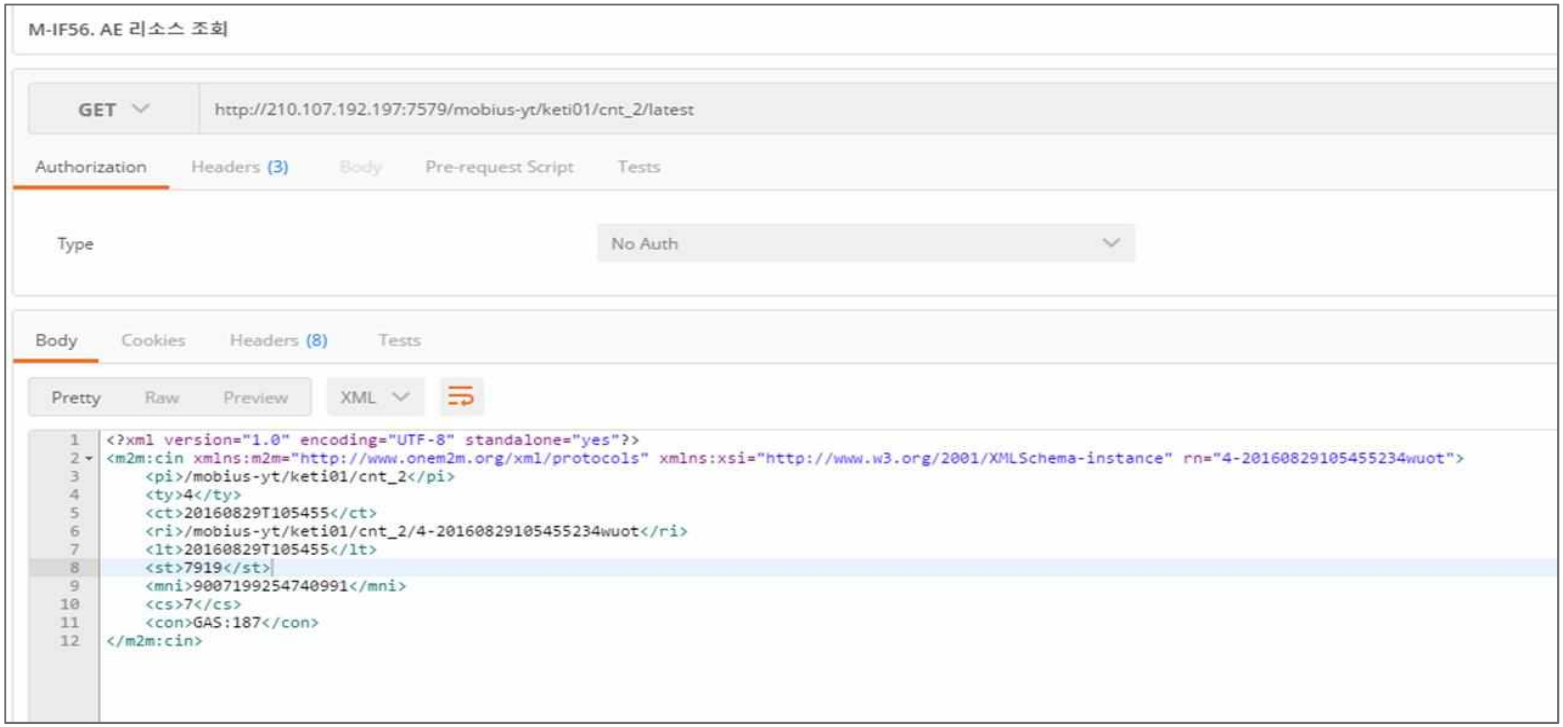


TAS 동작화면

- Node/thyme 디렉토리 내에서 nCube 실행 (sudo node thyme.js)
- Node/tas 디렉토리의 샘플코드 원본 또는 수정본 TAS 실행 (sudo node app.js)
- Putty 화면 내에 정상 동작 확인

14.3 TAS 작성 & Mobius 서버 연동

- Postman REST 조회 과정



- Mobius 서버와의 연동을 정상 동작 확인
- http://Mobius 서버주소/mobius-yt/센서 폴더트리/latest
- Chrome의 postman 앱을 통해 GAS 센서 값 조회

Assignment

- Assignment 14

제시된 TAS code를 사용하여 모터를 제어하기 위한 안드로이드 응용 서비스 어플리케이션을 개발해 보시오.

Reference

[14.1] 서비스 개요 & 요구 하드웨어

상용 서비스중인 L사 가스락

-출처 : http://www.uplus.co.kr/ent/iot/lotgasApp.hpi

THANK YOU

for **attending**