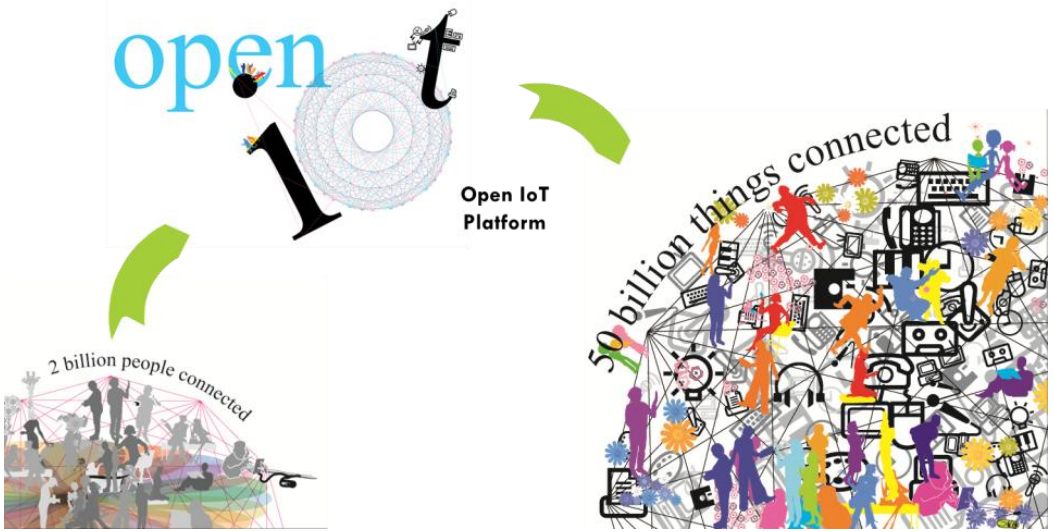
[사물인터넷 서비스 개발을 위한]

“개방형 사물인터넷 플랫폼 (Mobius, &Cube) 기반 서비스 개발 실습” 교육과정

-사물인터넷 디바이스 개발 과정-



전 자 부 품 연 구 원

목 차

1. 개 요

1.1 오픈소스 종류 및 oneM2M 표준 준수

1.2 &Cube: Thyme 활용 디바이스 개발 방법

1.3 &Cube 연동 구조

1.1 오픈소스 종류 및 oneM2M 표준 준수

■ 오픈소스 종류 별 oneM2M 표준 지원 여부

		AE	CSE			Framework
			ASN	MN	IN	
Mobius	Blue Octopus				√	Spring
	Yellow Turtle				√	Node.js
nCube	Rosemary			√		Java
				√		Node.js
	Lavender		√			Java
			√			Node.js
	Thyme	√				Java
		√				Node.js

■ Latest Release

	oneM2M Release 1				
	Software name	Framework	version	Publication date	Standard
Server(IN-CSE)	Mobius : Blue Octopus	Spring Framework	v1.1	2015-09-10	TS-0001 Functional Architecture v1.6.1 TS-0004 Service Layer Core Protocol v1.0.1 TS-0009 HTTP Protocol Binding v1.0.1 TS-0010 MQTT Protocol Binding v1.0.1
	Mobius : Yellow Turtle	Node.js	v1.1	2015-09-10	
Gateway(MN-CSE)	&Cube : Rosemary	Node.js	v1.0	2015-10	
Device(ASN-CSE)	&Cube : Lavender	Java	v1.0	2015-08-11	
Application(AE)	&Cube : Lavender	Node.js	v1.0	2015-10	

1.1 오픈소스 종류 및 oneM2M 표준 준수

■ Mobius: Blue Octopus

- Spring Framework 기반 IoT Server Platform (oneM2M IN-CSE)
- Java Virtual Machine 위에서 동작
- 시스템 요구사항

System requirements	Remarks
Operating System	WindowsX, Linux Redhat and CentOS
Java Virtual Machine	Java 7
Open Source Framework	Spring 3.06
Web Application Server	Spring MVC 3.0
Database	Tomcat 7
CoAP Framework	Mongo 2.6 Redis 2.8
CoAP Framework	Californium
MQTT Broker	Mosquitto 1.4.x

- 표준 리소스 지원
  - CSEBase, remoteCSE, AE, node, container, contentInstance, subscription, notification, mgmtCmd, execInstance, mgmtObj, group
- 표준 Primitive 지원
  - XML/Json 지원
  - Short-name 리소스 지원

1.1 오픈소스 종류 및 oneM2M 표준 준수

■ Mobius: Yellow Turtle

- Node.js Java Script 기반 IoT Server Platform (oneM2M IN-CSE)
- 경량/저용량 시스템으로 간편한 설치 가능
- 시스템 요구사항

System requirements	Remarks
Operating System	WindowsX, Linux Redhat and CentOS, Mac, Raspbian
Open Source Framework	Node.js
Web Application Server	Node.js
Database	MySQL
CoAP Framework	-
MQTT Broker	Mosquitto 1.4.x

- 표준 리소스 지원
  - CSEBase, remoteCSE, AE, container, contentInstance, subscription, notification, group
- 표준 Primitive 지원
  - XML/Json 지원
  - Long/Short-name 리소스 지원

1.1 오픈소스 종류 및 oneM2M 표준 준수

■ &Cube: Rosemary

- Node.js Java Script 기반 IoT Gateway Platform (oneM2M MN-CSE)
- 경량/저용량 시스템으로 간편한 설치 가능
- 시스템 요구사항

System requirements	Remarks
Operating System	WindowsX, Linux Redhat and CentOS, Mac, Raspbian
Open Source Framework	Node.js
Web Application Server	Node.js
Database	MySQL
CoAP Framework	-
MQTT Broker	Mosquitto 1.4.x

- 표준 리소스 지원
  - CSEBase, remoteCSE, AE, container, contentInstance, subscription, notification, group
- 표준 Primitive 지원
  - XML/Json 지원
  - Long/Short-name 리소스 지원

1.1 오픈소스 종류 및 oneM2M 표준 준수

■ &Cube: Lavender

- Java 기반 IoT Device Platform (oneM2M ASN-CSE)
- 경량/저용량 시스템으로 간편한 설치 가능
- Java Virtual Machine 위에서 동작
- 자체적으로 HTTP 서버를 소스 내부에 탑재하여 리소스 최소화
- 시스템 요구사항

System requirements	Remarks
Operating System	Windows, Linux, Mac OSX, Raspbian
Java Virtual Machine	Java 7 / 8
MQTT Broker	Mosquitto 1.4.x

- 표준 리소스 지원
  - CSEBase, remoteCSE, AE, node, container, contentInstance, mgmtCmd, execInstance, mgmtObj
- 표준 Primitive 지원
  - XML/Json 지원
  - Long/Short-name 리소스 지원

1.1 오픈소스 종류 및 oneM2M 표준 준수

■ &Cube: Thyme

- Node.js Java Script 기반 IoT Application (oneM2M AE)
- 경량/저용량 시스템으로 간편한 설치 가능
- 시스템 요구사항

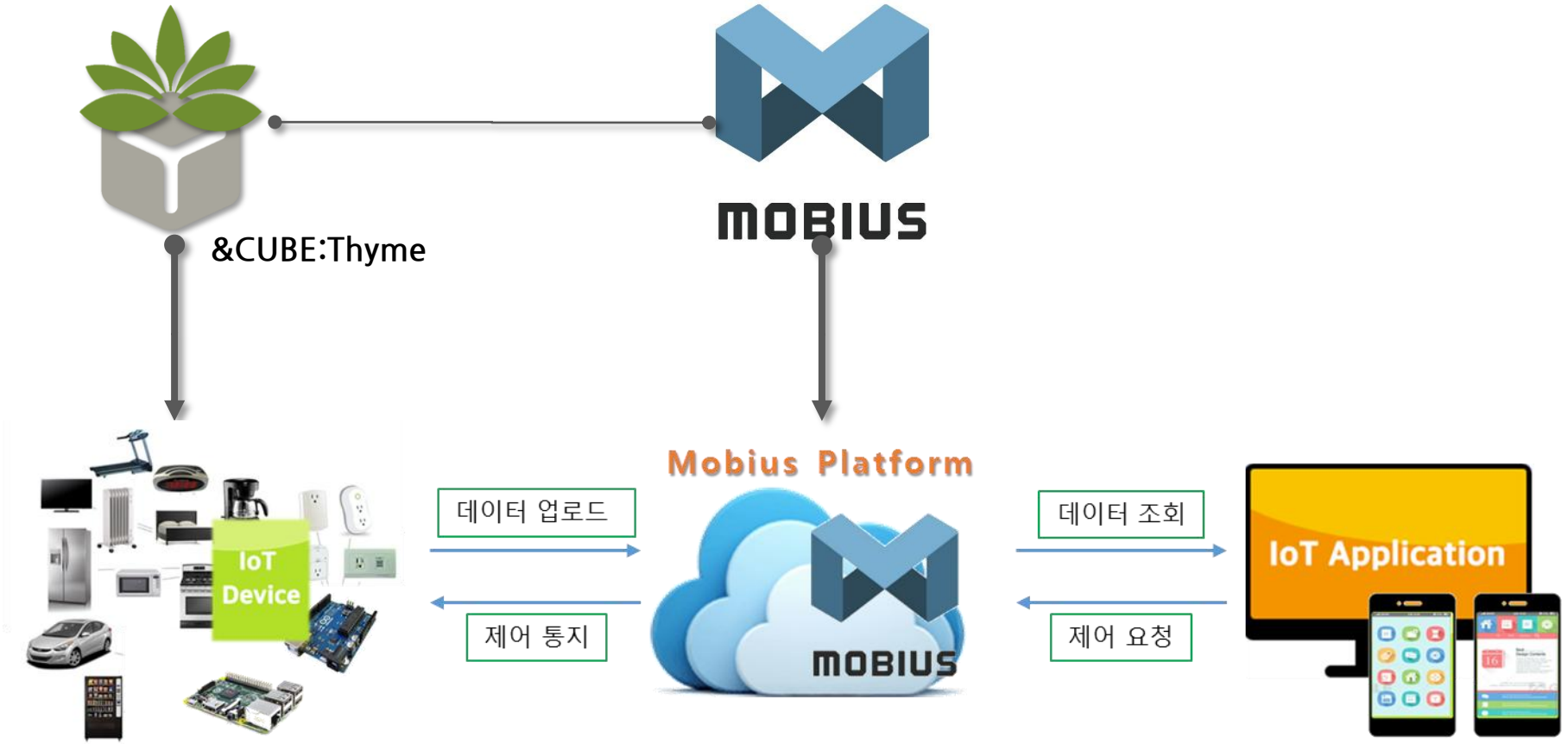
System requirements	Remarks
Operating System	WindowsX, Linux Redhat and CentOS, Mac, Raspbian
Open Source Framework	Node.js
Web Application Server	Node.js
Database	MySQL
CoAP Framework	-
MQTT Broker	Mosquitto 1.4.x

- 표준 리소스 지원
  - CSEBase, remoteCSE, AE, container, contentInstance, subscription, notification, group
- 표준 Primitive 지원
  - XML/Json 지원
  - Long/Short-name 리소스 지원

1. 개 요

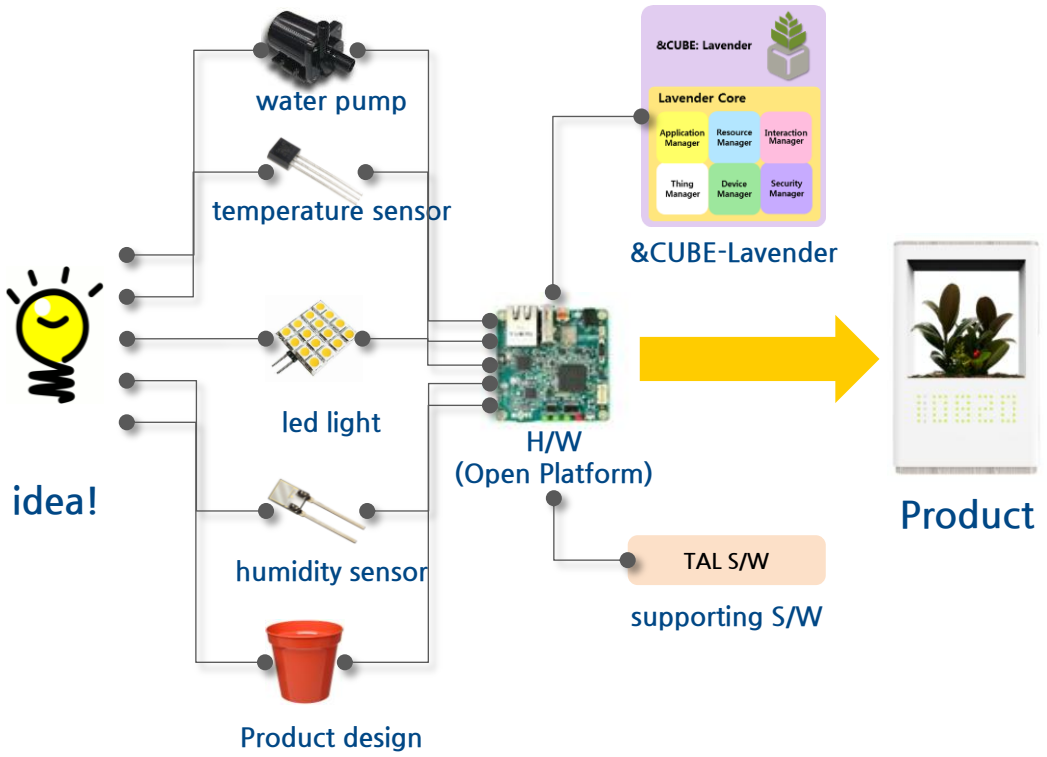
1.1 오픈소스 종류 및 oneM2M 표준 준수
1.2 &Cube: Thyme 활용 디바이스 개발 방법
1.3 &Cube 연동 구조

1.2 &Cube:Thyme 활용 디바이스 개발 방법



1.2 &Cube:Thyme 활용 디바이스 개발 방법

Product = Idea + H/W + &CUBE + TAS



IoT Server = Server(H/W) + Mobius

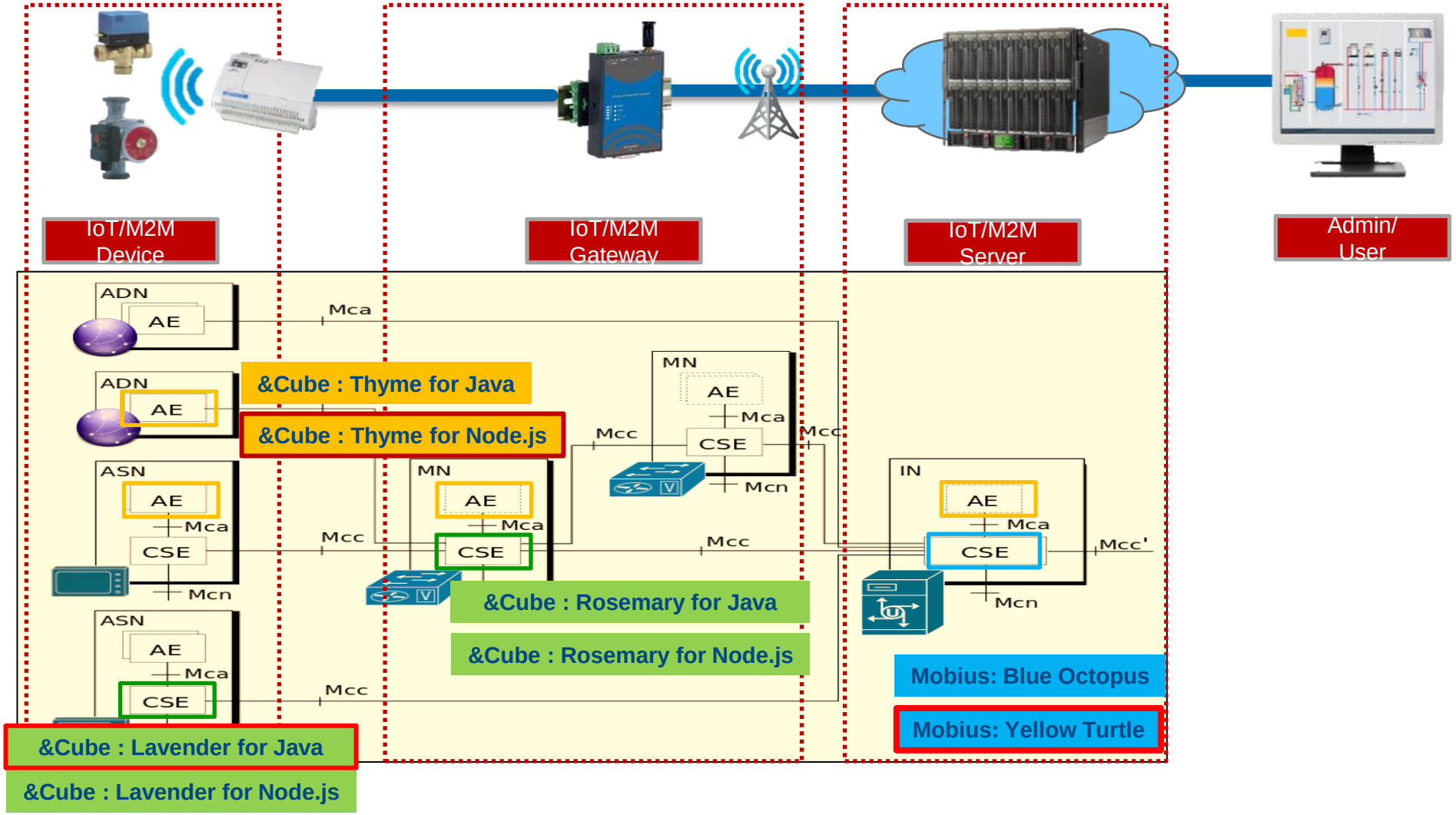
Yellow Turtle

Blue Octopus

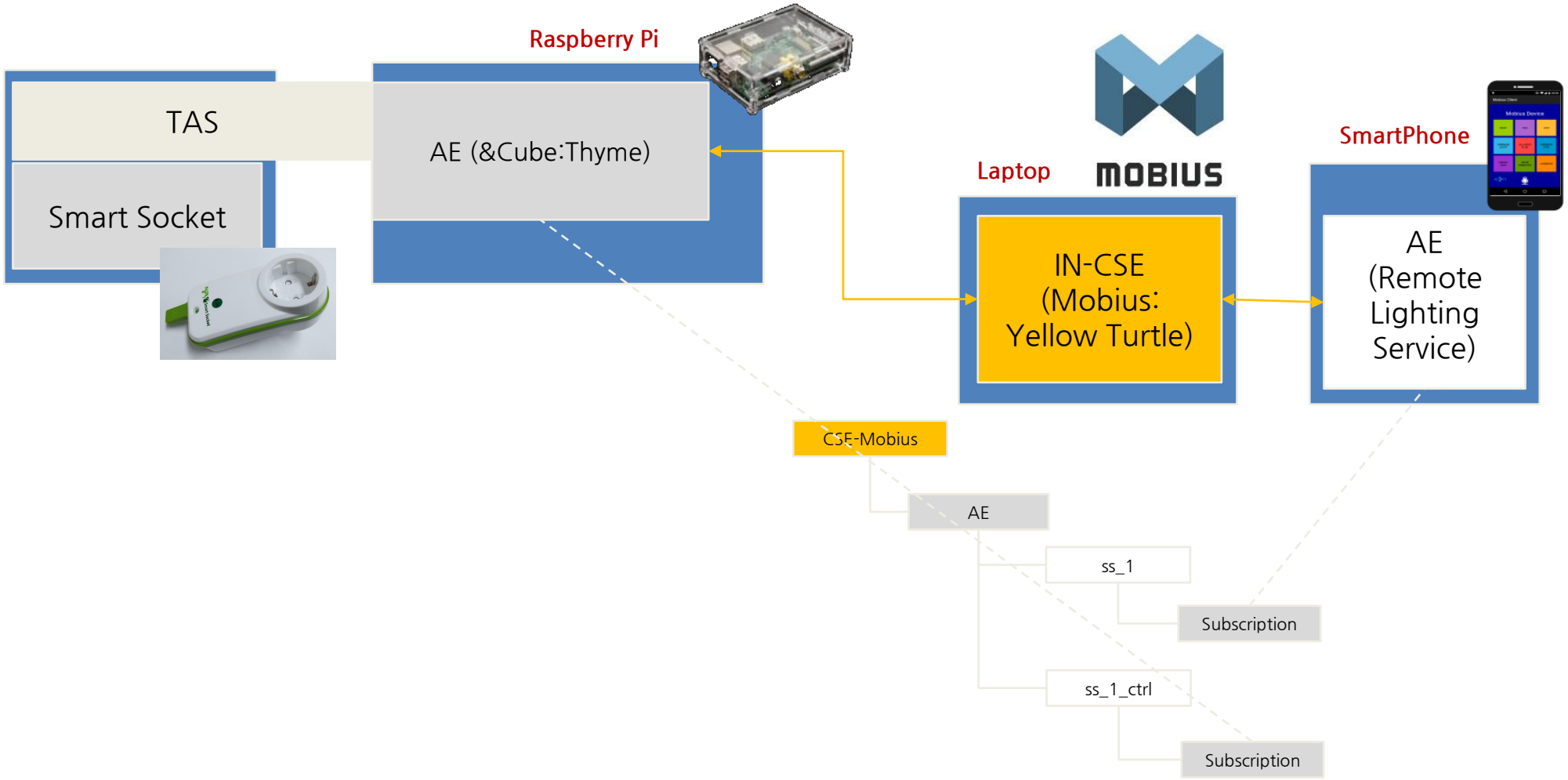
1. 개 요

1.1 오픈소스 종류 및 oneM2M 표준 준수
1.2 &Cube: Thyme 활용 디바이스 개발 방법
1.3 &Cube 연동 구조

1.3 &Cube 연동 구조


IoT/M2M Device
IoT/M2M Gateway
IoT/M2M Server
Admin/ User
ADN
AE
Mca
&Cube : Thyme for Java
&Cube : Thyme for Node.js
MN
CSE
Mcc
Mcn
ASN
IN
Mcc'
&Cube : Rosemary for Java
&Cube : Rosemary for Node.js
Mobius: Blue Octopus
Mobius: Yellow Turtle
&Cube : Lavender for Java
&Cube : Lavender for Node.js

1.4 &Cube 연동 구조


Raspberry Pi
TAS
Smart Socket
AE (&Cube:Thyme)
Laptop
MOBIUS
IN-CSE
(Mobius:
Yellow Turtle)
SmartPhone
AE
(Remote
Lighting
Service)
CSE-Mobius
AE
ss_1
Subscription
ss_1_ctrl
Subscription

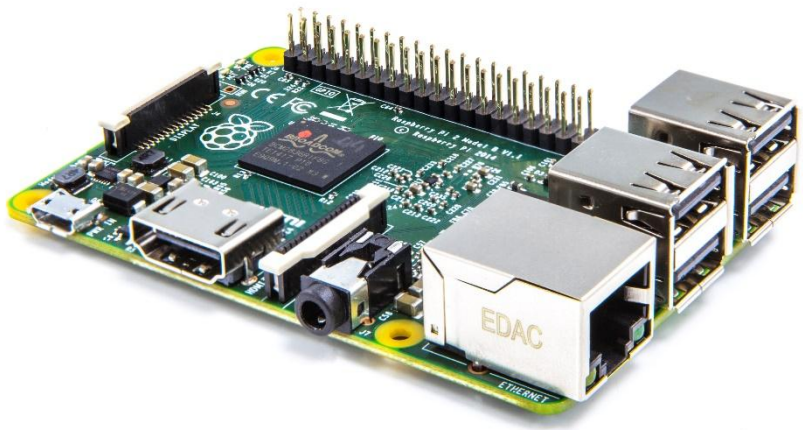
2. 사물인터넷 디바이스 개발환경 구축

2.1 사물인터넷 디바이스 Open H/W 소개
2.2 Open H/W 기반 개발환경 구축
2.3 &Cube: Thyme 구동환경 구축

2.1 사물인터넷 디바이스 Open H/W 소개

■ Raspberry Pi 2 Model B

- 영국의 라즈베리 파이 재단이 개발
- 기초 컴퓨터 과학 교육을 증진시키기 위해 만든 싱글 보드 컴퓨터
- Raspbian (Debian 계열 Linux) 운영체제 사용
- 기타 운영체제 포팅 가능
- 상세 정보는 http://www.raspberrypi.org/ 홈페이지에서 확인 가능



Item	Name	Feature
Processor	BCM2836	Broadcom BCM2836 SoC - 900Mhz ARM Cortex-A7 Quad core - Broadcom VideoCore IV Dual core - 1GByte Memory
Ethernet	LAN9514	SMSC LAN9514-jzx 10/100Mbps Ethernet Port - RJ-45 connector
USB	Host	USB 2.0 HS/FS/LS Host Port x 4
SD/MMC	microSD	1 microSD Socket (SDMMC2—BOOT)
Video Input	Connector	15Pin MIFI Camera interface connector x 2 - Raspberry Pi Camera - NoIR Camera
Video Output	HDMI	PAL/NTSC 640x350 ~ 1920x1200 resolution
	RCA	PAL/NTSC, Audio output
Audio Input	I2S	Inter-IC Sound, Integrated Interchip Sound
Audio Output	I2S	Inter-IC Sound, Integrated Interchip Sound - 3.5mm Audio jack - HDMI Digital Audio
Power	DC 5V	5V 800mA DC Input (Micro 5pin connector)
Expansion	Connector	GPIO 17 port, UART bus, I2C bus, SPI 2 port, I2S Audio port, +3.3V, +5V, Ground, HAT ID bus
Dimension	width x height	85.6 x 56.5 (mm)

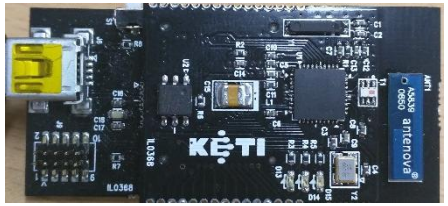
2.1 사물인터넷 디바이스 Open H/W 소개

■ Smart Socket

- 전력량 측정을 수행하고 ON/OFF 신호를 통해 전력차단이 가능한 디바이스
- 실제 전력 측정 및 전력차단을 수행하는 보드와 무선 통신으로 데이터 송신 및 제어 수신을 수행하는 보드의 2중 보드 형태
- TI CC2530을 이용한 ZigBee RF 무선 통신 수행
- 주기적으로 현재 전력량을 측정하여 Sink node로 전송
- Sink node로부터 제어 신호 수신 시 전원공급/차단 수행

■ Sink node

- TI CC2530
- CP2103 USB-to-Serial
- Mini USB 활용 연결 인터페이스 제공



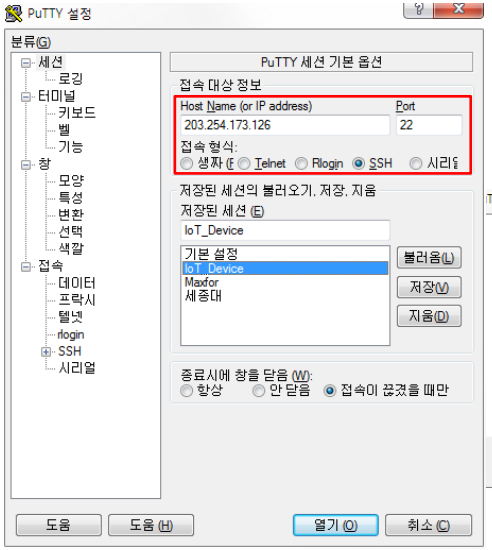
2. 사물인터넷 디바이스 개발환경 구축

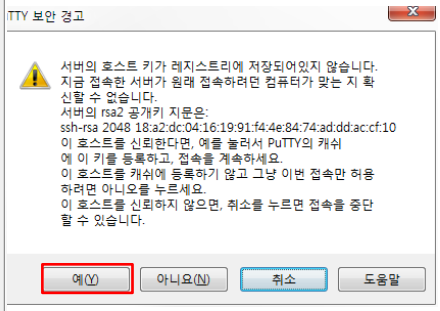
2.1 사물인터넷 디바이스 Open H/W 소개
2.2 Open H/W 기반 개발환경 구축
2.3 &Cube: Thyme 구동환경 구축

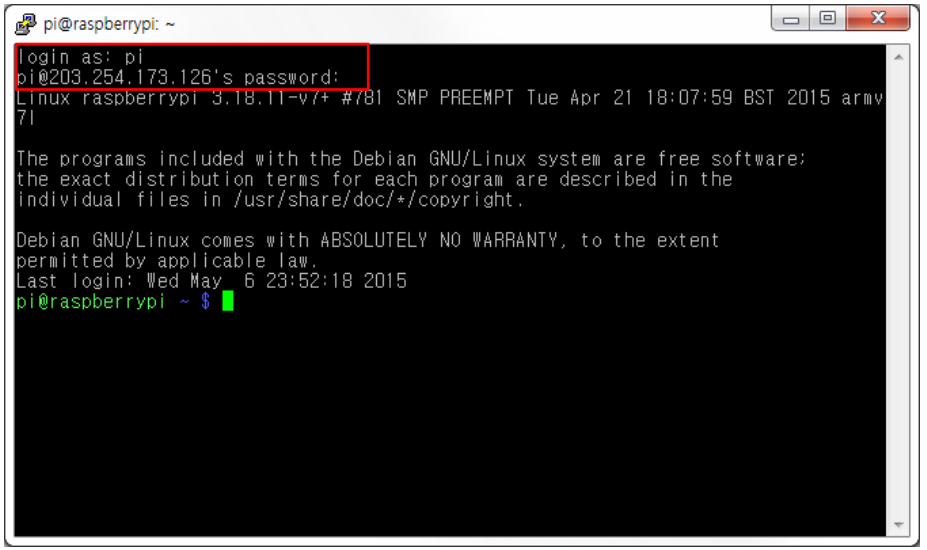
2.2 Open H/W 기반 개발환경 구축

■ 원격 접속환경 구축 (SSH client 사용)

- SSH client 프로그램 다운로드 및 설치 (PuTTY)
  - http://www.chiark.greenend.org.uk/~sgtatham/putty/
- PuTTY 프로그램 설정 및 연결
  - Raspberry-Pi의 IP address 입력을 통해 접속
  - 호스트 키 확인 메시지 출력 시 '예'를 눌러 캐쉬에 등록
  - 접속 완료 시 로그인 진행 (Raspberry-Pi 초기 설정 → ID : pi, PW : raspberry)







2.2 Open H/W 기반 개발환경 구축

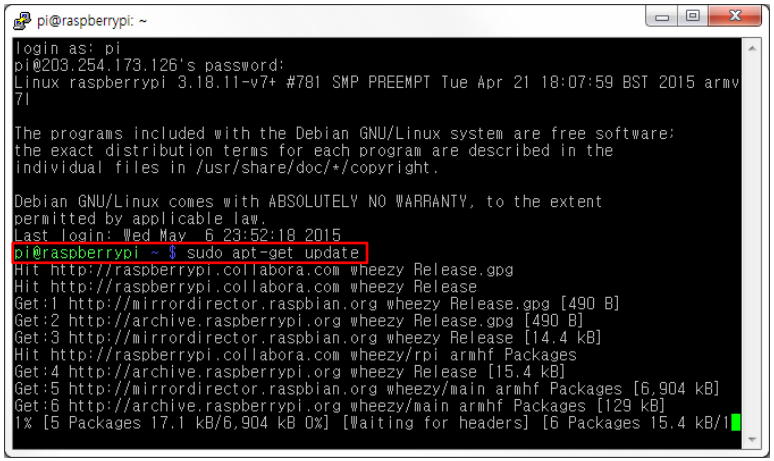
■ Samba 서버 구축

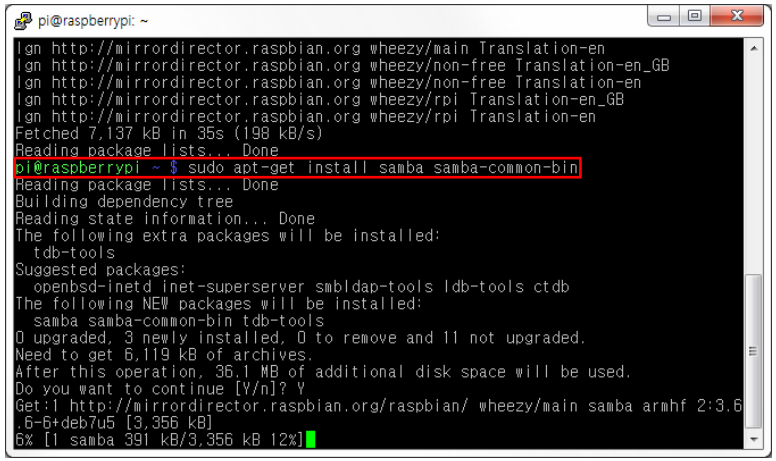
● Repository 업데이트

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

● Samba 서버 설치

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



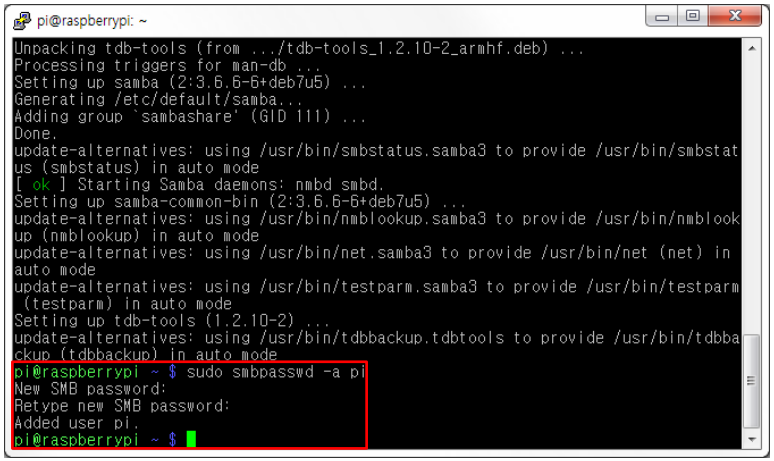


2.2 Open H/W 기반 개발환경 구축

■ Samba 서버 구축

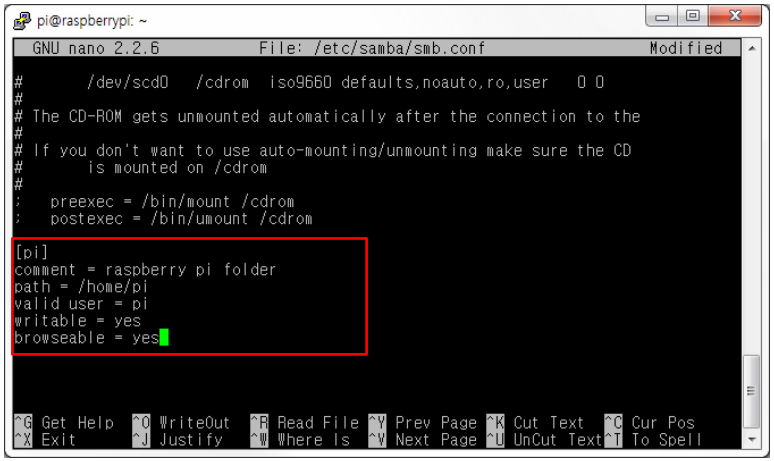
● Samba 서버 사용자 추가

```
pi@raspberrypi ~ $ sudo smbpasswd –a pi
New SMB password: (원하는 패스워드 입력)
Retype new SMB password: (원하는 패스워드 입력)
Added user pi.
```



● Samba 서버 사용자 설정

```
pi@raspberrypi ~ $ sudo nano /etc/samba/smb.conf
…… (가장 마지막 줄 밑에)
[pi]
comment = raspberry pi folder
path = /home/pi
valid user = pi
writable = yes
browseable = yes
<Ctrl>+<X> → Y → <Enter>
```

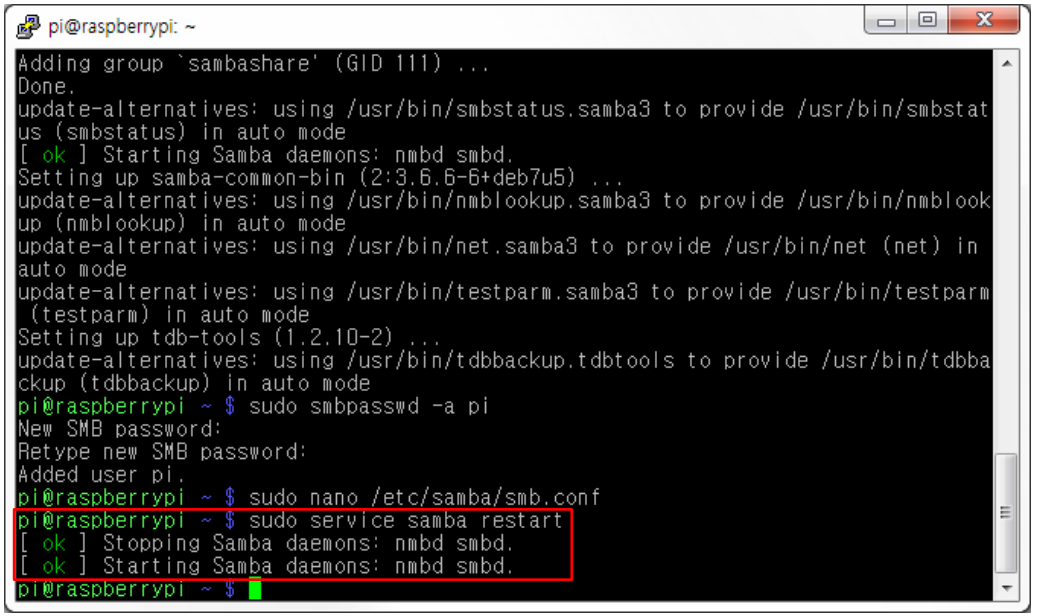


2.2 Open H/W 기반 개발환경 구축

■ Samba 서버 구축

● Samba 서버 재시작

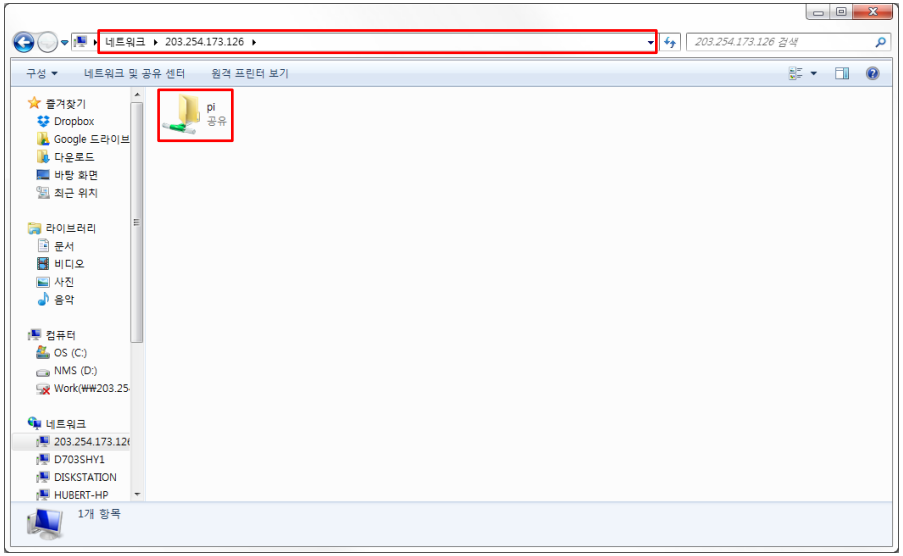
```
pi@raspberrypi ~ $ sudo service samba restart
[ ok ] Stopping Samba daemons: nmbd smbd.
[ ok ] Starting Samba daemons: nmbd smbd.
```

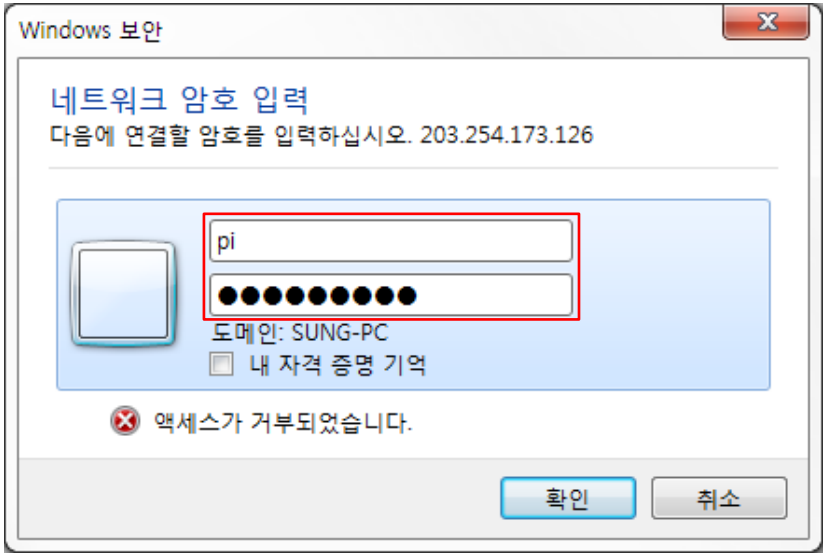


2.2 Open H/W 기반 개발환경 구축

■ Samba 폴더 연결

- Windows 탐색기 실행
- 주소입력 창에 ₩₩Raspberry-Pi IP 주소 입력
  - 예) ₩₩203.254.173.126
- pi 폴더 더블클릭
- 계정 및 패스워드 입력
  - 계정 : pi, 패스워드 : Samba서버 설치 시 지정한 패스워드





2. 사물인터넷 디바이스 개발환경 구축

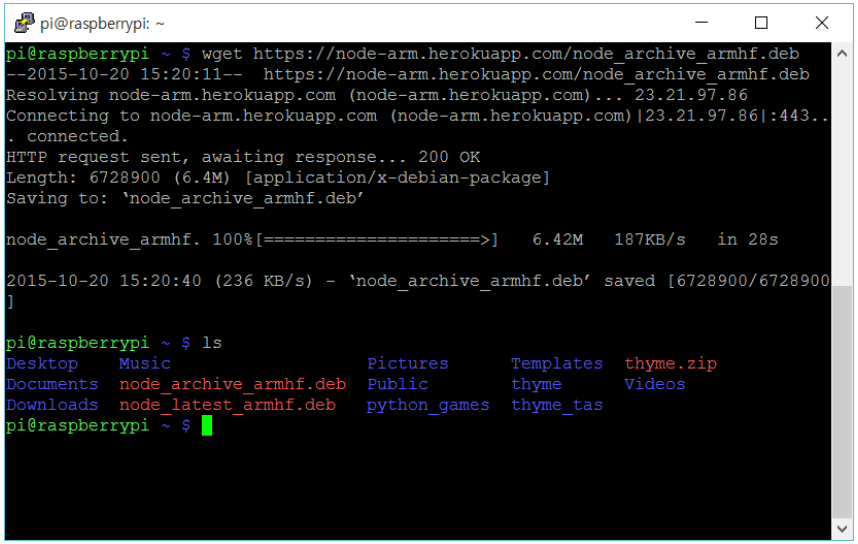
2.1 사물인터넷 디바이스 Open H/W 소개
2.2 Open H/W 기반 개발환경 구축
2.3 &Cube: Thyme 구동환경 구축

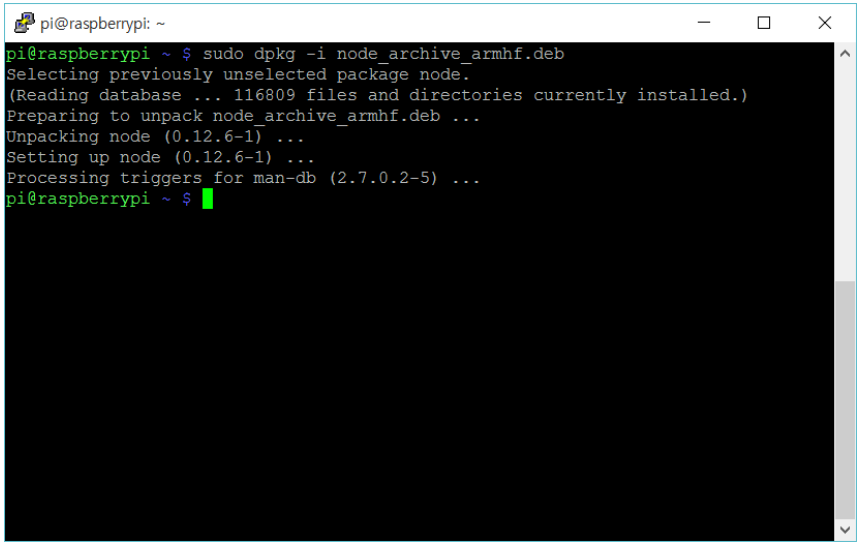
2.3 &Cube: Thyme 구동환경 구축

■ Node.js 패키지 설치

● Node.js 다운로드 및 설치

```
pi@raspberrypi ~ $ mkdir node
pi@raspberrypi ~ $ cd node
pi@raspberrypi ~/node $ sudo wget https://node-arm.herokuapp.com/node_archive_armhf.deb
pi@raspberrypi ~/node $ sudo dpkg –i node_archive_armhf.deb (패키지 설치 명령어)
pi@raspberrypi ~/node $ node –v (버전 확인 명령어)
pi@raspberrypi ~/node $ npm -v (추가 라이브러리 설치도구 버전 확인 명령어)
```

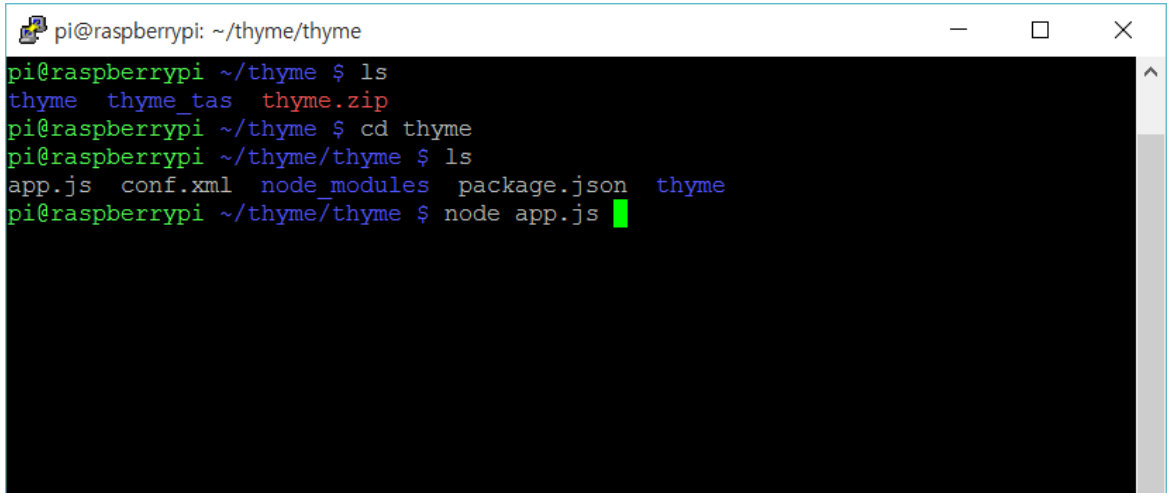




2.3 &Cube: Thyme 구동환경 구축

- &Cube: Thyme 샘플 다운로드 및 구동 테스트
  - www.iotocean.org 에서 &Cube:Thyme 다운로드
  - 구동 테스트 → &Cube: Thyme 실행

```
pi@raspberrypi ~/node/thyme $ sudo npm install (추가 라이브러리 설치 명령)
pi@raspberrypi ~/node/thyme $ node app.js (실행 명령)
```



3. 사물인터넷 디바이스 개발 실습

3.1 &Cube: Thyme 설정
3.2 &Cube: Thyme 및 Mobius 간 연동
3.3 &Cube: Thyme 코드 수정
3.4 Rest Client 활용 Mobius 데이터 확인 및 제어

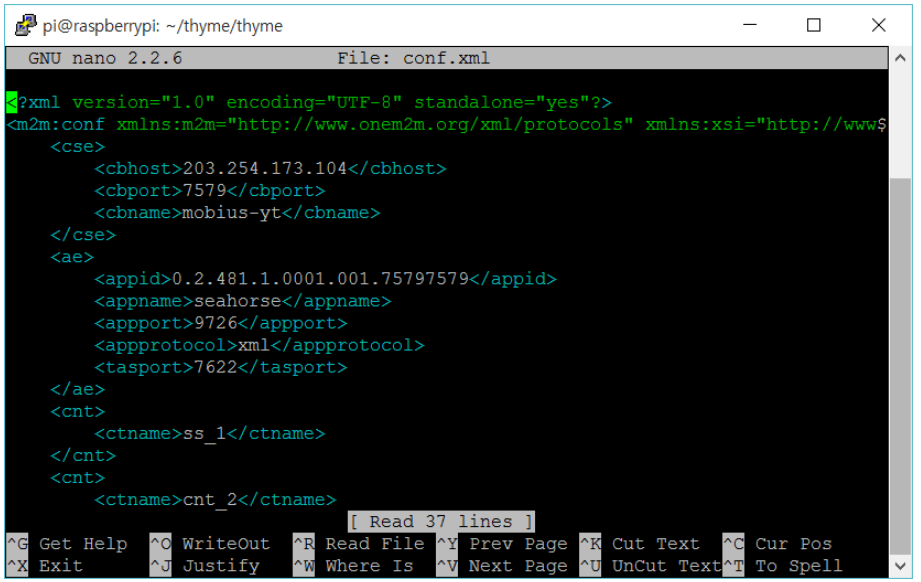
3.1 &Cube: Thyme 설정

■ &Cube: Thyme Configure file 설정

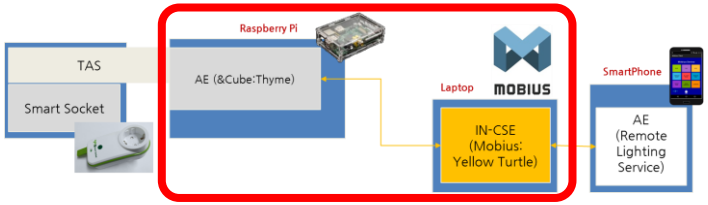
● Conf.xml 수정

```
pi@raspberrypi ~/node/thyme $ sudo nano conf.xml
```

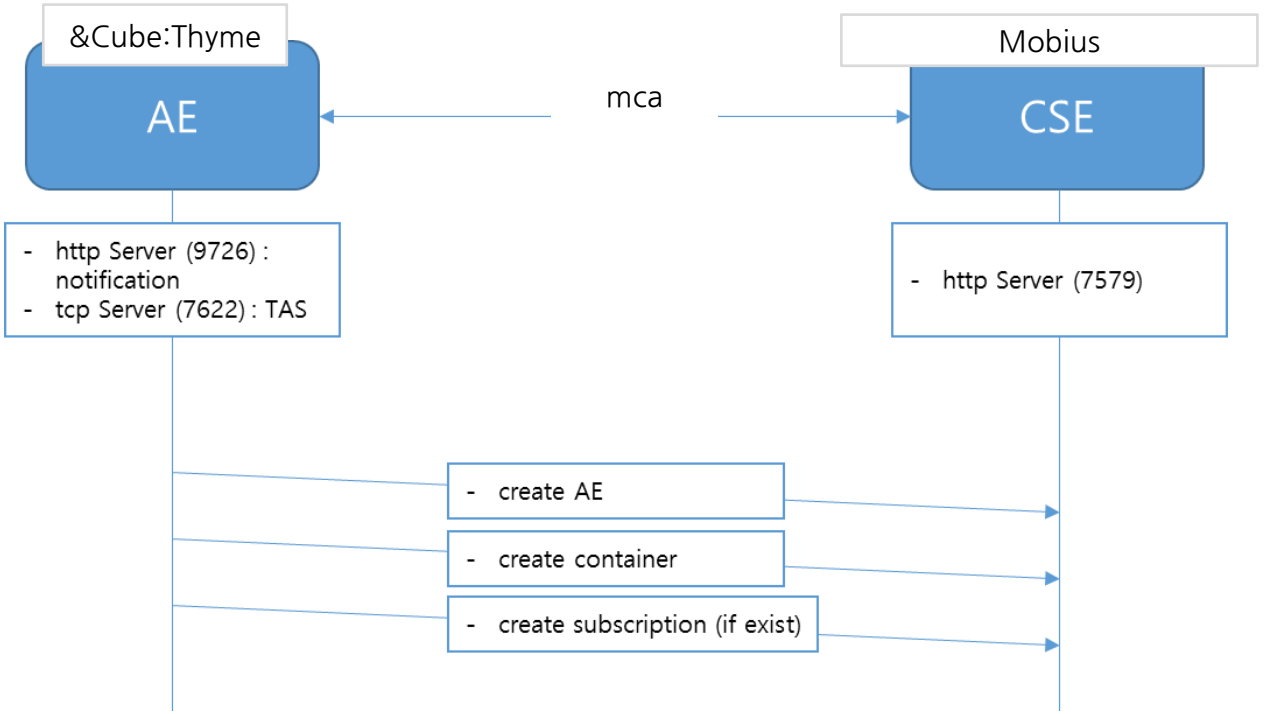
```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<conf xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <cse>
        <cbhost>localhost</cbhost>
        <cbport>7579</cbport>
        <cbname>mobius-yt</cbname>
    </cse>
    <ae>
        <appid>0.2.481.1.0001.001.75797579</appid>
        <appname>seahorse</appname>
        <appport>9726</appport>
        <appprotocol>xml</appprotocol>
        <tasport>7622</tasport>
    </ae>
    <cnt>
        <ctname>ss_1</ctname> % 업로드용
    </cnt>
    <cnt>
        <ctname>cnt_2</ctname> % 업로드용
    </cnt>
</conf>
```

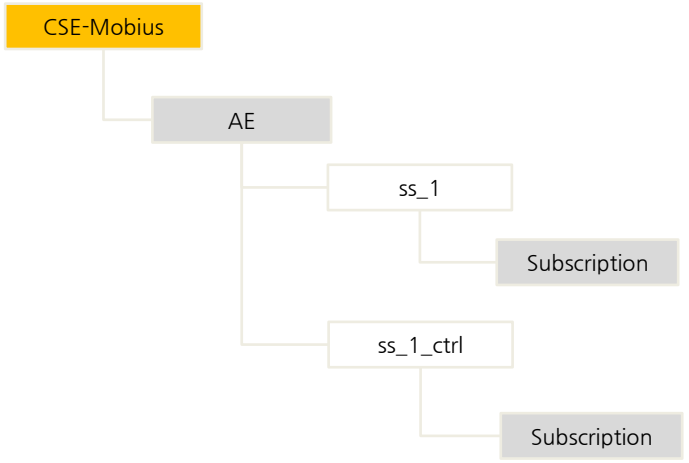


3.1 &Cube: Thyme 설정



■ &Cube: Thyme - Mobius (AE - CSE) MSC example





3.1 &Cube: Thyme 설정

■ &Cube: Thyme 설정 변경 확인

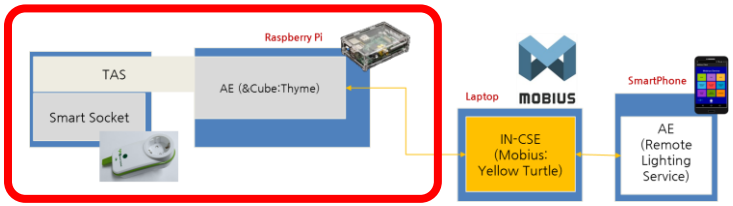
● &Cube: Thyme 재실행

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

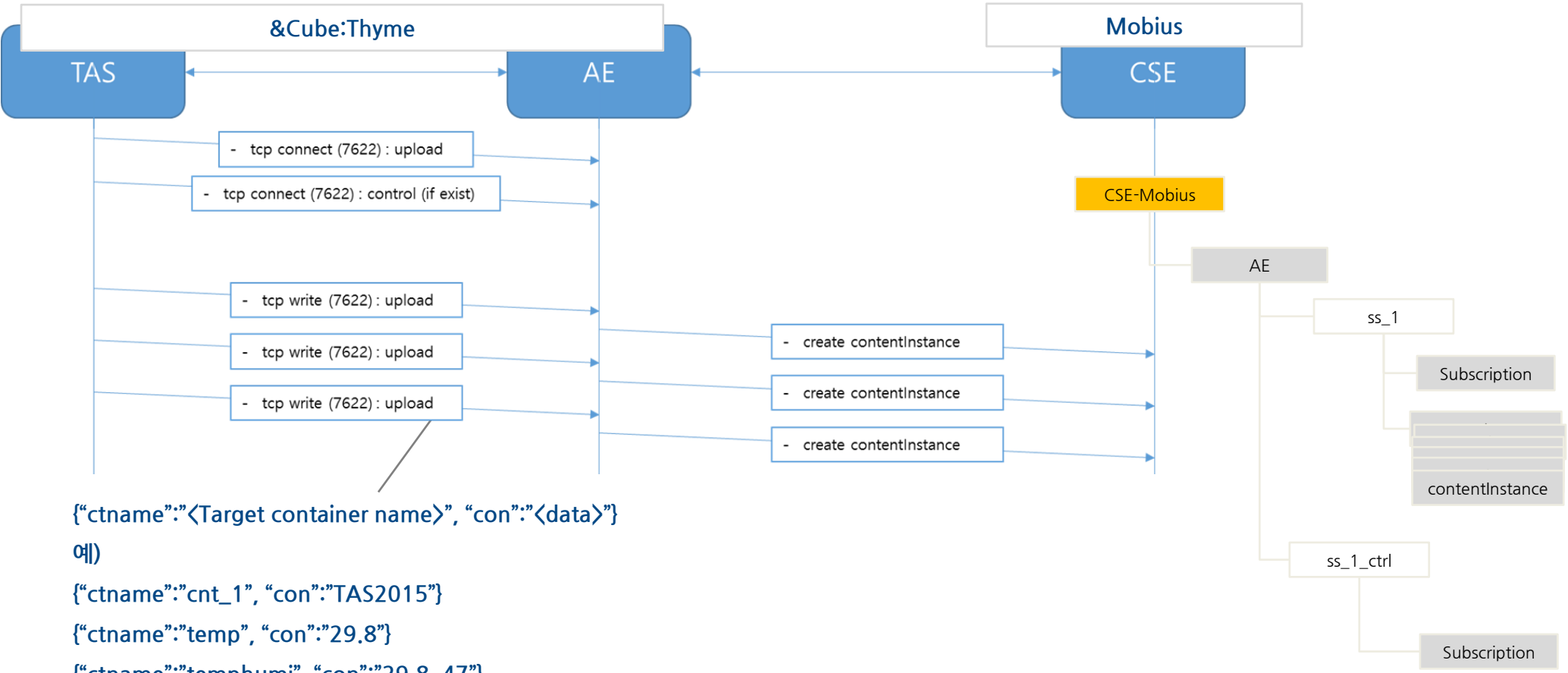
3. 사물인터넷 디바이스 개발 실습

3.1 &Cube: Thyme 설정
3.2 &Cube: Thyme 및 Mobius 간 연동
3.3 &Cube: Thyme 코드 수정
3.4 Rest Client 활용 Mobius 데이터 확인 및 제어

3.2 &Cube: Thyme 및 Mobius 간 연동



- &Cube: Thyme TAS 샘플 다운로드 (iotocean.org)
- TAS - &Cube: Thyme - Mobius (MSC example)



{"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"}

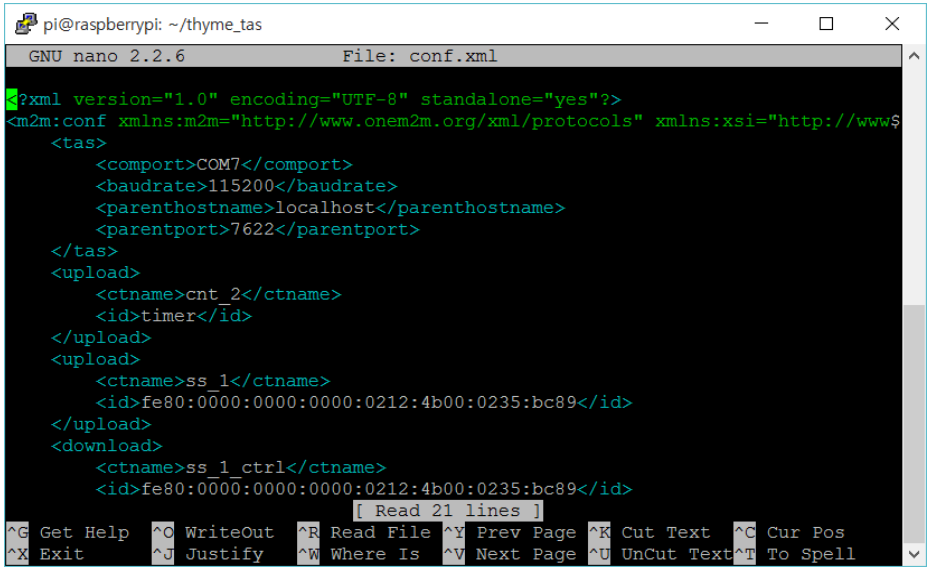
3.2 &Cube: Thyme 및 Mobius 간 연동

■ &Cube: Thyme TAS Configure file 설정

● Conf.xml 수정

```
pi@raspberrypi ~/node/thyme_tas $ sudo nano conf.xml
```

```
<?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>115200</baudrate>
        <parenthostname>localhost</parenthostname>
        <parentport>7622</parentport>
    </tas>
    <upload>
        <ctname>cnt_2</ctname>
        <id>timer</id>
    </upload>
    <upload>
        <ctname>ss_1</ctname>
        <id>fe80:0000:0000:0000:0212:4b00:0235:bc89</id>
    </upload>
    <download>
        <ctname>ss_1_ctrl</ctname>
        <id>fe80:0000:0000:0000:0212:4b00:0235:bc89</id>
    </download>
</m2m:conf>
```

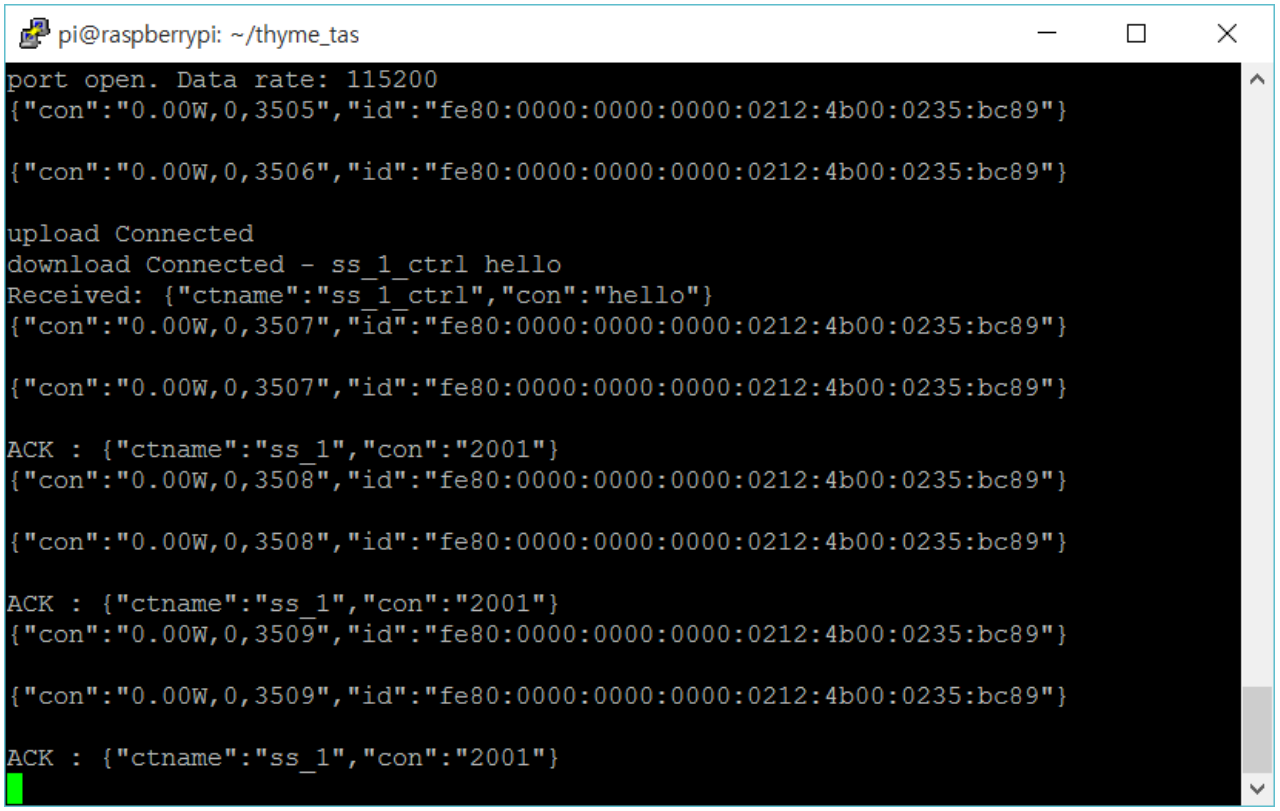


3.2 &Cube: Thyme 및 Mobius 간 연동

■ &Cube: Thyme TAS 샘플 구동 테스트

● &Cube: Thyme TAS 실행

pi@raspberrypi ~/node/thyme_tas $ node app.js (실행 명령)



3. 사물인터넷 디바이스 개발 실습

3.1 &Cube: Thyme 설정
3.2 &Cube: Thyme 및 Mobius 간 연동
3.3 &Cube: Thyme 코드 수정
3.4 Rest Client 활용 Mobius 데이터 확인 및 제어

3.3 &Cube: Thyme 코드 수정

- &Cube: Thyme 코드
  - TAS 데이터 파싱

```
net.createServer(function (socket) {
    console.log('socket connected');
    socket.on('data', function(data) {
        var data_arr = data.toString().split('}');
        for(var i = 0; i < data_arr.length-1; i++) {
            var line = data_arr[i];
            line += '}';
            var jsonObj = JSON.parse(line);
            var ctname = jsonObj.ctname;
            var content = jsonObj.con;

            socket_arr[ctname] = socket;

            console.log('got data from tas for [' + ctname + ']');

            if(jsonObj.con == 'hello') {
                socket.write(line);
            }
            else {
                if (sh_state == 'crtci') {
                    for (var j = 0; j < usectname.length; j++) {
                        if (usectname[j].ctname == ctname) {
                            console.log(line);
                            sh_adn.crtci(usecbhost, usecbport, usecbname, useappid, useappname, ctname, content, socket, function (status, ctname) {
                                if (status == 5106 || status == 2001 || status == 4015) {
                                    socket.write('{\"ctname\":\"' + ctname + '\",\"con\":\"' + status + '\"}');
                                }
                                else if (status == 5000) {
                                    sh_state = 'crtae';
                                    socket.write('{\"ctname\":\"' + ctname + '\",\"con\":\"' + status + '\"}');
                                }
                                else {
                                    socket.write('{\"ctname\":\"' + ctname + '\",\"con\":\"' + status + '\"}');
                                }
                            });
                        }
                    }
                }
            }
        }
    });
```

3.3 &Cube: Thyme 코드 수정

- &Cube: Thyme TAS 코드
  - 시리얼 데이터 파싱

```
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;
            }
        }
    }
});
```

```
function saveLastestData(data) {
    //var c = util.format('%s', data);
    //console.log(data);



    //g_sink_buf_index = g_sink_buf_index + 1;
    //console.log(g_sink_buf_index);

    if (data == '{') {
//        console.log('%s', 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 ==  '}') {
        //console.log('%s', 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;
        //g_sink_buf.toString('ascii');
    }
    else if (g_sink_buf_start == 1) {
        //       console.log('%s', data);
        cur_c = util.format('%s', data);
        g_sink_buf = pre_c + cur_c;
        pre_c = g_sink_buf;
    }
}
```

3. 사물인터넷 디바이스 개발 실습

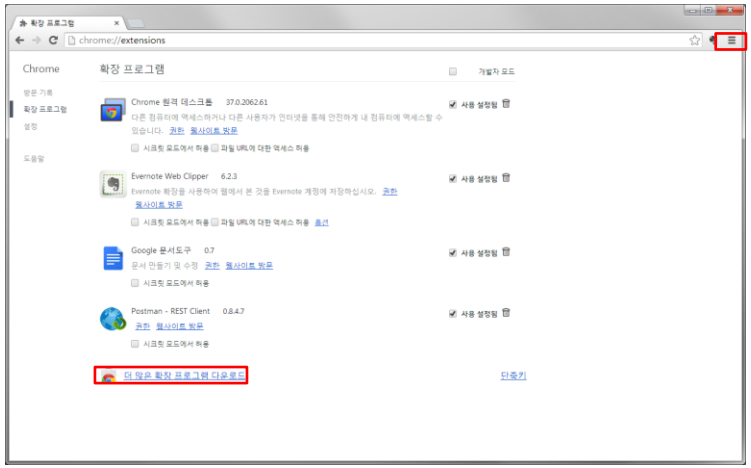
3.1 &Cube: Thyme 설정
3.2 &Cube: Thyme 및 Mobius 간 연동
3.3 &Cube: Thyme 코드 수정
3.4 Rest Client 활용 Mobius 데이터 확인 및 제어

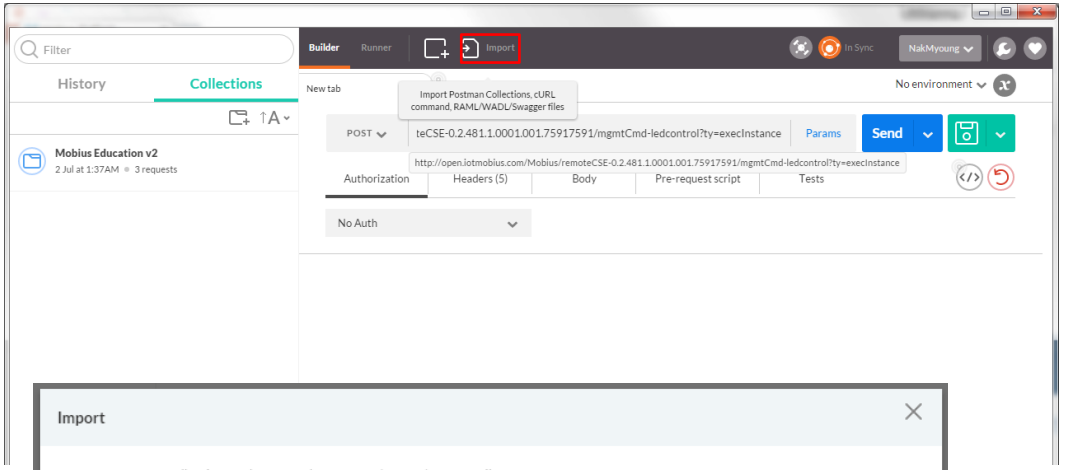
3.4 Rest Client 활용 Mobius 데이터 확인 및 제어

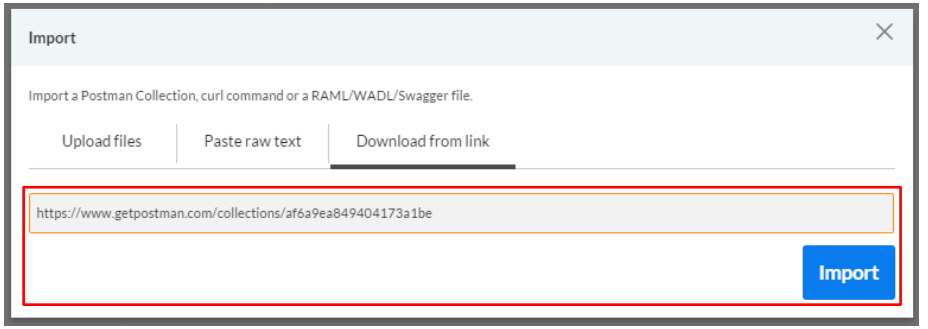
■ Postman Rest Client 설치 및 실행

● Postman 실행 및 Import

- Google Chrome Browser 설치 및 실행
- 우측 상단 메뉴 → 도구 → 확장 프로그램 → 더 많은 확장 프로그램 다운로드 → Postman - REST Client 실행
- Collections → import collection
- https://www.getpostman.com/collections/3ace2fbf0f44ec357afc 입력 및 import



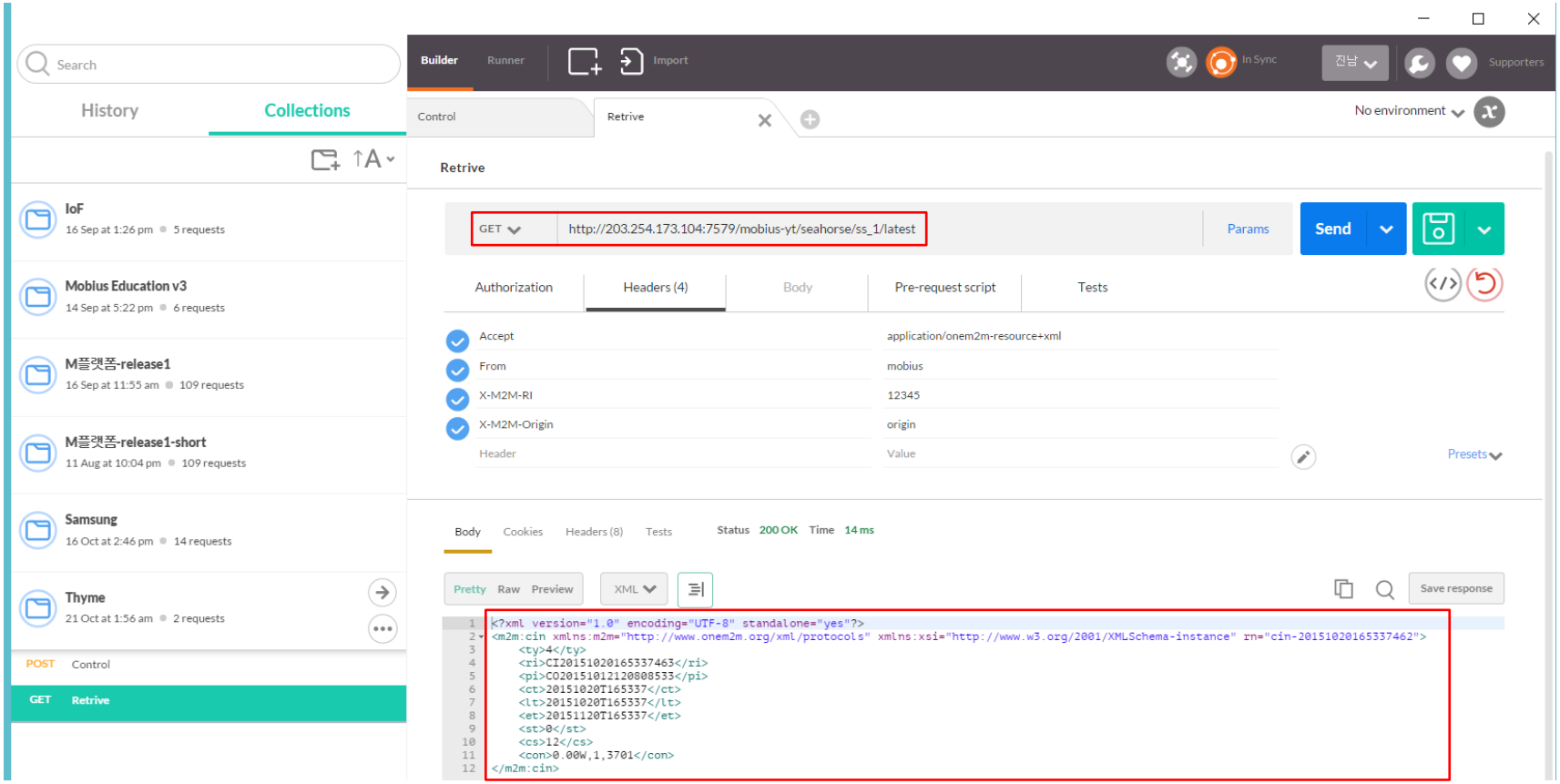




3.4 Rest Client 활용 Mobius 데이터 확인 및 제어

■ Postman Rest Client Data 조회 활용

- Postman 활용 Mobius Device Data 확인
  - Body 리턴 결과 업로드 된 데이터 확인

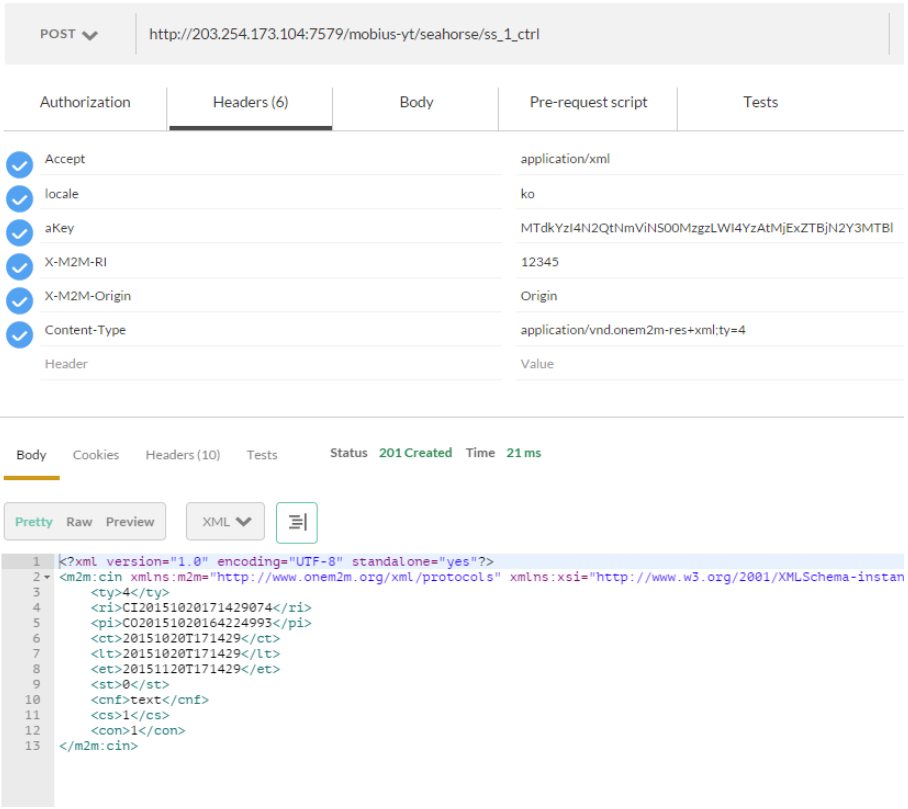


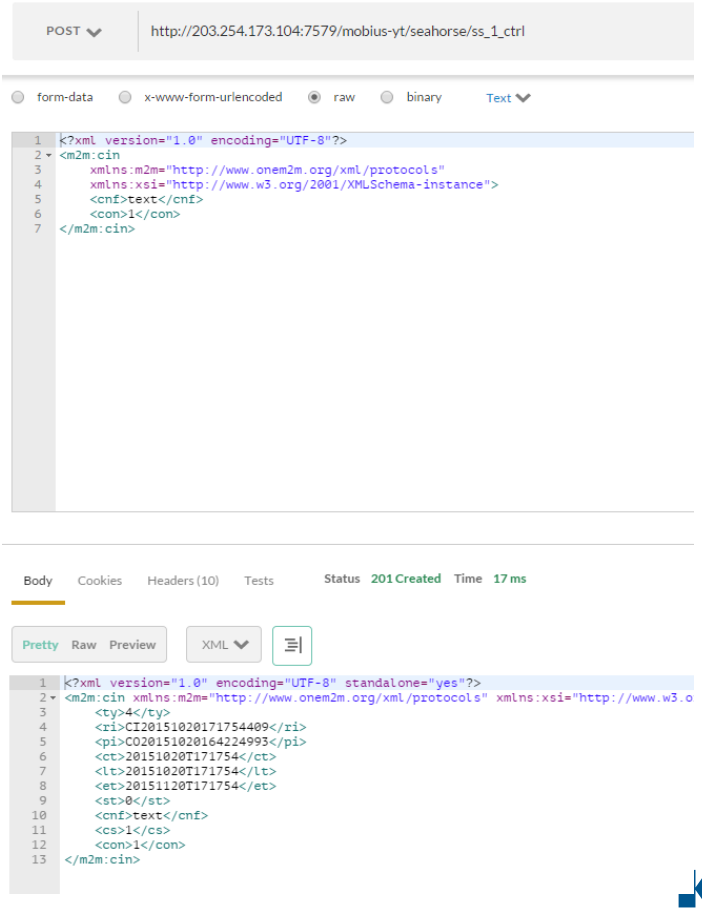
3.4 Rest Client 활용 Mobius 데이터 확인 및 제어

■ Postman Rest Client 제어 활용

● Postman 활용 Mobius Device 제어

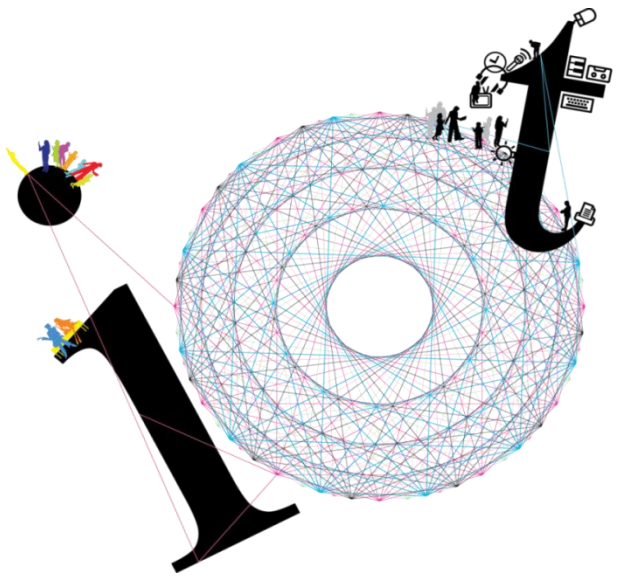
- POST 메소드 Body <con>에 제어 커맨드 입력
- PuTTY &Cube 플랫폼 및 TAS에서 제어 커맨드 수신 및 디바이스 제어 결과 확인



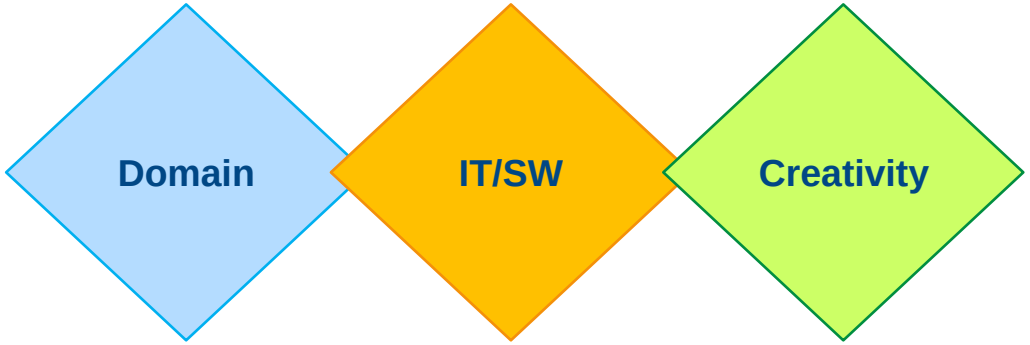


Conclusion

사용자 | Find, Connect, Control



Dream, Do, Realize | 개발자



Thank you

문 의 : 안일엽, 성낙명 {iyahn, nmsung}@keti.re.kr

031-789-7579, 7591