



## nCube-Thyme-nodejs

**Software Version: 2.5.0**

---

## TAS(Virtual Emulator) Guide v1.0.0

Document Release Date: Mar 2023

Software Release Date: Mar 2023



## Revision History

버전	변경일자	제·개정 내역
1.0.0	2023-03-10	<p>초안</p> <p>wonseok@keti.re.kr / 정원석 hyeonseo0128@keti.re.kr / 손현서 <a href="mailto:hdkyon96@keti.re.kr">hdkyon96@keti.re.kr</a> / 허대근</p>

# nCube-Thyme-Nodejs (TAS-Emulator)

---

## 1. 개요

본 문서는 TAS(Thing Adaptation Software)-Emulator에서 생성된 가상의 데이터를 오픈 소스 IoT device application entity인 **nCube-Thyme-Nodejs**를 활용하여 Mobius라는 오픈 소스 IoT 서버 플랫폼에 업로드하는 예제를 설명합니다.

본 문서의 목적과 과정은 다음과 같습니다.

- nCube-Thyme-Nodejs와 tas\_emulator를 이용하여 tas\_emulator에서 생성한 가상 데이터를 Mobius(IN-CSE)에 수집합니다.
  - TAS-Emulator에서 생성된 가상 데이터를 로컬 MQTT 프로토콜을 사용하여 nCube-Thyme-Nodejs로 전송하고, Mobius에 CIN(ContentInstance)로 업로드합니다.
  - Subscribe/Notify 기능을 통해 Mobius에 생성된 CIN 데이터를 nCube-Thyme-Nodejs에서 수신하여 TAS-Emulator에 전달함으로써 가상 LED를 제어합니다.

## 2. nCube-Thyme-Nodejs & TAS-Emulator 구조

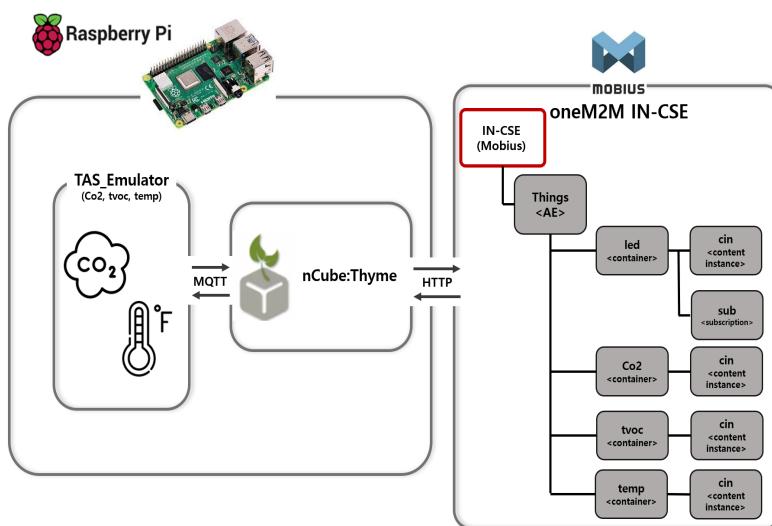


그림 1. nCube-Thyme-Nodejs & TAS-Emulator를 이용한 oneM2M 플랫폼 서비스 구조도

## 2. 준비물

---

### 2.1. H/W

- Raspberry Pi (3 시리즈 이상)
- Micro SD Card (8GB 이상)
- Optional
  - USB Keyboard
  - USB Mouse

## 2.2. S/W

- Raspberry Pi Raspbian (buster 이상)
- Node.js (16.x 버전 이상)
- [Mobius](#)
- [nCube-Thyme-Nodejs](#)
- MQTT-broker

## 3. 실습 환경 구축

### 3.1. Node.js 설치

아래의 명령어를 통해 패키지 저장소(Personal Package Archive)에 Node.js에 대한 패키지를 업데이트합니다.

```
$ curl -sL https://deb.nodesource.com/setup_16.x | sudo -E bash -
```

위 명령어의 16.x를 원하는 버전으로 변경하여 설치 가능합니다.

```
pi@raspberrypi:~ $ curl -sL https://deb.nodesource.com/setup_16.x | sudo -E bash -
## Installing the NodeSource Node.js 16.x repo ...
## Populating apt-get cache ...
+ apt-get update
Hit:1 http://raspbian.raspberrypi.org/raspbian bullseye InRelease
Hit:2 http://archive.raspberrypi.org/debian bullseye InRelease
Reading package lists ... Done
## Confirming "bullseye" is supported ...
+ curl -sLf -o /dev/null 'https://deb.nodesource.com/node_16.x/dists/bullseye/Release'
## Adding the NodeSource signing key to your keyring ...
+ curl -s https://deb.nodesource.com/gpgkey/nodesource.gpg.key | gpg --dearmor | tee /usr/share/keyrings/nodesource.gpg >/dev/null
## Creating apt sources list file for the NodeSource Node.js 16.x repo ...
+ echo 'deb [signed-by=/usr/share/keyrings/nodesource.gpg] https://deb.nodesource.com/node_16.x bullseye main' > /etc/apt/sources.list.d/nodesource.list
+ echo 'deb-src [signed-by=/usr/share/keyrings/nodesource.gpg] https://deb.nodesource.com/node_16.x bullseye main' >> /etc/apt/sources.list.d/nodesource.list
## Running `apt-get update` for you ...
+ apt-get update
Hit:1 http://archive.raspberrypi.org/debian bullseye InRelease
Hit:2 http://raspbian.raspberrypi.org/raspbian bullseye InRelease
Get:3 https://deb.nodesource.com/node_16.x bullseye InRelease [4,586 B]
Get:4 https://deb.nodesource.com/node_16.x bullseye/main armhf Packages [783 B]
Fetched 5,369 B in 2s (3,013 B/s)
Reading package lists ... Done
## Run `sudo apt-get install -y nodejs` to install Node.js 16.x and npm
## You may also need development tools to build native addons:
  sudo apt-get install gcc g++ make
## To install the Yarn package manager, run:
  curl -sL https://dl.yarnpkg.com/debian/pubkey.gpg | gpg --dearmor | sudo tee /usr/share/keyrings/yarnkey.gpg >/dev/null
  echo "deb [signed-by=/usr/share/keyrings/yarnkey.gpg] https://dl.yarnpkg.com/debian stable main" | sudo tee /etc/apt/sources.list.d/yarn.list
  sudo apt-get update & sudo apt-get install yarn
```

업데이트 된 패키지 버전의 Node.js를 설치합니다.

```
$ sudo apt-get install -y nodejs
```

```
pi@raspberrypi:~ $ sudo apt-get install -y nodejs
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following package was automatically installed and is no longer required:
  libfuse2
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  nodejs
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 24.2 MB of archives.
After this operation, 120 MB of additional disk space will be used.
Get:1 https://deb.nodesource.com/node_16.x bullseye/main armhf nodejs armhf 16.19.1-deb-1nodesource1 [24.2 MB]
Fetched 24.2 MB in 11s (2,107 kB/s)
Selecting previously unselected package nodejs.
(Reading database ... 106404 files and directories currently installed.)
Preparing to unpack .../nodejs_16.19.1-deb-1nodesource1_armhf.deb ...
Unpacking nodejs (16.19.1-deb-1nodesource1) ...
Setting up nodejs (16.19.1-deb-1nodesource1) ...
Processing triggers for man-db (2.9.4-2) ...
```

### 3.2. nCube-Thyme-Nodejs 다운로드 및 패키지 설치

GitHub에 배포된 nCube-Thyme-Nodejs를 git 명령어를 통해 다운로드합니다.

```
$ git clone https://github.com/IoTKETI/nCube-Thyme-Nodejs
```

```
pi@raspberrypi:~ $ git clone https://github.com/IoTKETI/nCube-Thyme-Nodejs
Cloning into 'nCube-Thyme-Nodejs' ...
remote: Enumerating objects: 4354, done.
remote: Counting objects: 100% (51/51), done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 4354 (delta 26), reused 35 (delta 13), pack-reused 4303
Receiving objects: 100% (4354/4354), 12.97 MiB | 4.05 MiB/s, done.
Resolving deltas: 100% (1031/1031), done.
```

다운로드한 nCube-Thyme-Nodejs 폴더로 이동하여 실행을 위한 패키지를 설치합니다.

```
$ cd nCube-Thyme-Nodejs
$ npm install
```

```
pi@raspberrypi:~ $ cd nCube-Thyme-Nodejs/
pi@raspberrypi:~/nCube-Thyme-Nodejs $ npm install
added 139 packages, and audited 140 packages in 18s
13 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
npm notice
npm notice New major version of npm available! 8.19.3 → 9.5.1
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.5.1
npm notice Run npm install -g npm@9.5.1 to update!
npm notice
```

### 3.3. Mosquitto MQTT broker 설치

Mosquitto 서명키(인증키)를 다운로드하고 추가합니다.

```
$ wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key  
$ sudo apt-key add mosquitto-repo.gpg.key
```

Mosquitto 저장소 패키지를 등록합니다. 아래 명령어에서 **buster**에 해당하는 부분은 Raspbian의 OS 버전에 따라 변경해야 합니다.

```
$ cd /etc/apt/sources.list.d/  
$ sudo wget http://repo.mosquitto.org/debian/mosquitto-buster.list
```

Mosquitto MQTT 브로커를 설치합니다.

```
$ sudo apt-get update  
$ sudo apt-get install -y mosquitto
```

## 4. nCube-Thyme-Nodejs & TAS-Emulator 설정

### 4.1. nCube-Thyme-Nodejs

#### 4.1.1. **conf.js** 설정 파일 수정

- Mobius 연동에 사용할 통신 프로토콜 설정, nCube-Thyme-Nodejs 자체 가상 데이터 사용 설정

변수명	설명
<b>conf.useprotocol</b>	사용할 프로토콜 선택
<b>conf.sim</b>	가상 데이터 사용여부 선택

- 연동할 Mobius 설정

CSE 변수명	설명
host	실행중인 Mobius의 주소
port	Mobius HTTP 연동 포트
name	Mobius 이름
id	Mobius id
mqttport	Mobius MQTT 연동 포트
wsport	Mobius 웹소켓 연동 포트

- AE 생성, Notification 포트 등 설정

AE 변수명	설명
ae_name	사용자가 설정할 AE의 이름
name	사용자가 설정할 AE의 이름
id	AE ID
parent	AE를 생성 경로
appid	어플리케이션 식별자
port	Notification을 받기 위한 포트
bodytype	Mobius와 주고받는 메시지의 body 형식(json/xml/cbor)
tasport	TAS와 소켓 통신하기 위한 포트

- AE 하위 리소스 생성을 위한 설정

CNT 변수명	설명
parent	CNT 생성 경로
name	생성할 CNT 이름

- Subscription 생성을 위한 설정

SUB 변수명	설명
parent	SUB 경로
name	SUB 이름
nu	Notification URL

아래 명령어를 통해 수정합니다.

```
$ cd nCube-Thyme-Nodejs
$ nano conf.js
```

```
conf.useprotocol = 'http'; // select one for 'http' or 'mqtt' or 'coap' or 'ws'

conf.sim = 'disable'; // enable or disable

// build cse
cse = {
  host : 'localhost', // CSE host ip ex) 192.168.1.1
  port : '7579',      // CSE http hosting port
  name : 'Mobius',
  id   : '/Mobius2',
```

```

    mqttport: '1883',           //CSE mqtt broker port
    wsport : '7577',
};

// build ae
let ae_name = 'device_DEMO';

ae = {
    name     : ae_name,        // AE name to create
    id       : 'S'+ae_name,   // AE-ID
    parent   : '/' + cse.name,
    appid   : 'measure_co2',
    port     : '9727',
    bodytype: 'json',
    tasport : '3105',
};

cnt_arr = [
{
    parent: '/' + cse.name + '/' + ae.name,
    name: 'temp',
},
{
    parent: '/' + cse.name + '/' + ae.name,
    name: 'humi',
},
{
    parent: '/' + cse.name + '/' + ae.name,
    name: 'led',
},
];
];

// build sub

sub_arr = [
{
    parent: cnt_arr[2].parent + '/' + cnt_arr[2].name,
    name: 'sub1',
    nu: 'mqtt://' + cse.host + ':' + cse.mqttport + '/' + ae.id + '?ct=json',
// 'http://' + ip.address() + ':' + ae.port + '/noti?ct=json',
},
];
];

let tas = {
    client: {
        connected: false,
    },
    connection: {
        host: 'localhost', // nCube MQTT broker ip
        port: 1883,         // MQTT broker port
        endpoint: '',
        clean: true,
        connectTimeout: 4000,
    }
};

```

```

        reconnectPeriod: 4000,
        clientId: 'thyme_' + nanoid(15),
        username: 'keti_thyme',
        password: 'keti_thyme',
    },
};


```

#### 4.1.2. `thyme_tas.js` TAS 연동

- TAS-Emulator로 부터 받은 데이터를 파싱하고 TAS로부터 데이터를 받기 위한 로컬 MQTT Topic을 설정하고 TAS로부터 받은 데이터를 Mobius에 CIN으로 생성합니다.

```

let getDataTopic = {           // sub topic
    temp: '/thyme/temp',
    humi: '/thyme/humi'
};

let setDataTopic = {           // pub topic
    led: '/led/set',
};

...

conf.tas.client.on('message', (topic, message) => {
    let content = null;
    let parent = null;
    /* USER CODES */
    let act = null;
    if(topic === getDataTopic.temp) {
        parent = conf.cnt[1].parent + '/' + conf.cnt[0].name;
        let curTime = moment().format();
        let curVal = parseFloat(message.toString()).toFixed(1);
        content = {
            t: curTime,
            v: curVal
        };
    }
    else if(topic === getDataTopic.humi) {
        parent = conf.cnt[1].parent + '/' + conf.cnt[1].name;
        let curTime = moment().format();
        let curVal = parseFloat(message.toString()).toFixed(1);
        content = {
            t: curTime,
            v: curVal
        };
    }

    if((content !== null) && (topic === getDataTopic.humi) && (content.v > 50)){
        act = 1;
        console.log("light type set humi :", content.v);
    }
}


```

```

        else if((content !== null) && (topic === getDataTopic.humi) && (content.v <
50)){
            act = 3;
            console.log("light type set humi :", content.v);
        }

        if(content !== null) {
            onem2m_client.create_cin(parent, 1, JSON.stringify(content), this,
function (status, res_body, to, socket) {
            console.log('x-m2m-rsc : ' + status + ' -----');
        });
        if(act !== null){
            parent = conf.cnt[1].parent + '/' + conf.cnt[2].name;
            let curTime = moment().format();
            let act_content ={
                t: curTime,
                v: act
            };
            onem2m_client.create_cin(parent, 1, JSON.stringify(act_content), this,
function (status, res_body, to, socket) {
            console.log('x-m2m-rsc : ' + status + ' -----');
        });
        }
    }
    /* */
});

```

#### 4.1.3. [app.js](#) Notification 연동

- 4.1.1.에서 설정한 SUB을 통해 Notification을 받아 발생한 이벤트에 대해 처리를 하고 TAS에 전달하기 위한 설정을 합니다.

```

onem2m_client.on('notification', function (source_uri, cinObj) {

    console.log(source_uri, cinObj);

    var path_arr = source_uri.split('/');
    var event_cnt_name = path_arr[path_arr.length-2];
    var content = cinObj.con;

    /* ***** USER CODE ***** */
    if(event_cnt_name === 'led') {
        // send to tas
        thyme_tas.send_to_tas(event_cnt_name, content.v);
    }
    /* */
});

```

## 4.2. TAS-Emulator

#### 4.2.1. `tas_emulator.js` 가상 데이터 생성 및 nCube-Thyme-Nodejs 연동

- 가상 데이터를 생성하고 설정한 Topic에 따라 로컬 MQTT를 통해 생성한 가상 데이터를 nCube-Thyme-Nodejs로 전달합니다.

```
let sendDataTopic = {
    co2: '/thyme/co2',
    tvoc: '/thyme/tvoc',
    temp: '/thyme/temp',
};

let recvDataTopic = {
    led: '/led/set',
};

...

tas.client.on('message', (topic, message) => {
    let content = null;

    /* USER CODES */
    if(topic === recvDataTopic.led) {
        // LED 제어
    }
    /* */
});

...

/* USER CODE */

setInterval(() => {
    let val = (Math.random() * 50).toFixed(1);
    doPublish(sendDataTopic['tvoc'], val);
}, 3000); // sec

setInterval(() => {
    let val = (Math.random() * 50).toFixed(1);
    doPublish(sendDataTopic['co2'], val);
}, 5000); // sec

setInterval(() => {
    let val = (Math.random() * 50).toFixed(1);
    doPublish(sendDataTopic['temp'], val);
}, 7000); // sec

/* */
```

## 5. nCube-Thyme-Nodejs 구동 실습

## 5.1. 실습 1

TAS-Emulator - 가상 데이터를 생성하여 로컬 MQTT broker를 통해 nCube-Thyme-Nodejs로 데이터를 전달하는 실습

1. nCube-Thyme-Nodejs를 설치한 디렉토리로 이동 후 thyme.js 파일을 실행합니다.

```
$ cd nCube-Thyme-Nodejs
$ node thyme.js
```

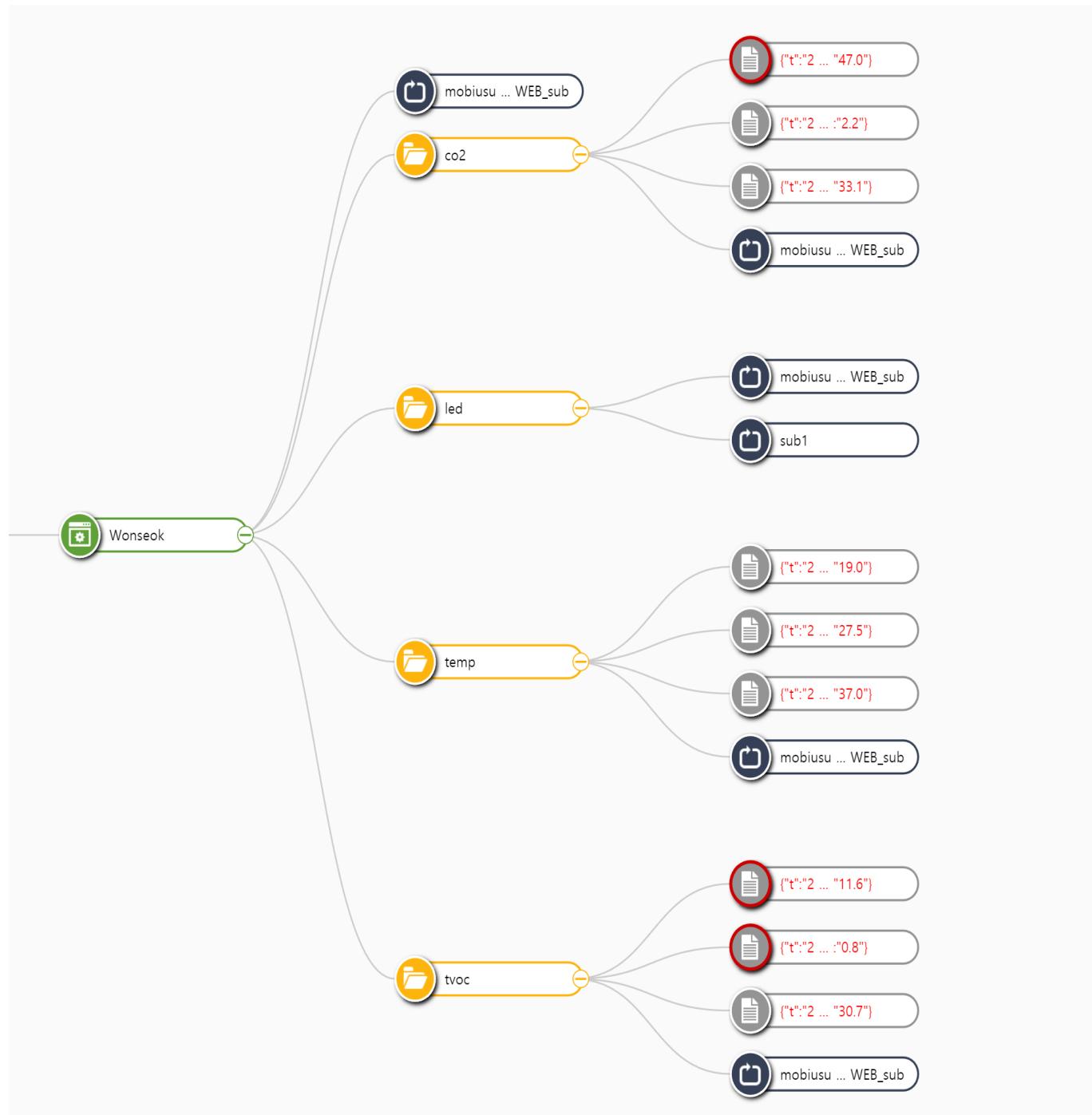
nCube-Thyme-Nodejs가 실행되면 AE 생성 - AE 조회 - CNT 생성 - SUB 삭제 - SUB 생성 - CIN 생성의 절차를 거쳐 TAS에서 데이터를 받기 위한 상태가 됩니다.

```
[status] : crtae
/Mobius
{ 'm2m:dbg': 'resource is already exist' }
x-m2m-rsc : 4105 <----
[status] : rtvae
/Mobius/Wonseok
x-m2m-rsc : 2000 - SWonseok <----
[status] : crtct
{"m2m:cnt":{"rn":"tvoc","lbl":["tvoc"]}}
/Mobius/Wonseok
0 - /Mobius/Wonseok/tvoc - x-m2m-rsc : 4105 <----
{ 'm2m:dbg': 'resource is already exist' }
{"m2m:cnt":{"rn":"co2","lbl":["co2"]}}
/Mobius/Wonseok
1 - /Mobius/Wonseok/co2 - x-m2m-rsc : 4105 <----
{ 'm2m:dbg': 'resource is already exist' }
{"m2m:cnt":{"rn":"temp","lbl":["temp"]}}
/Mobius/Wonseok
2 - /Mobius/Wonseok/temp - x-m2m-rsc : 4105 <----
{ 'm2m:dbg': 'resource is already exist' }
{"m2m:cnt":{"rn":"led","lbl":["led"]}}
/Mobius/Wonseok
3 - /Mobius/Wonseok/led - x-m2m-rsc : 4105 <----
{ 'm2m:dbg': 'resource is already exist' }
[status] : delsub
/Mobius/Wonseok/led/sub1
0 - /Mobius/Wonseok/led/sub1 - x-m2m-rsc : 2002 <----
{
  'm2m:sub': {
    pi: '3-20230308085012025379',
    ri: '23-20230308095427756320',
    ty: 23,
    ct: '20230308T095427',
    rn: 'sub1',
    lt: '20230308T095427',
    et: '20250308T095427',
    enc: { net: [Array] },
    exc: 100,
    nu: [ "mqtt://localhost:1883/SWonseok?ct=json" ],
    nct: 2,
    cr: 'SWonseok'
  }
}
[status] : crtsub
{"m2m:sub":{"rn":"sub1","enc":{"net":[1,2,3,4]},"nu":["mqtt://localhost:1883/SWonseok?ct=json"],"nct":2}}
/Mobius/Wonseok/led
[mqtt_connect] noti_topic : /oneM2M/req/+/SWonseok/#
0 - /Mobius/Wonseok/led/sub1 - x-m2m-rsc : 2001 <----
{"m2m:sub":{"rn":"sub1","ty":23,"pi":"3-20230308085012025379","ri":"23-2023030809541549709","ct":"20230308T095441","lt":"20230308T095441","et":"20250308T095441","nu":["mqtt://localhost:1883/SWonseok?ct=json"],"enc":{"net":[1,2,3,4]},"exc":100,"nct":2,"cr":"SWonseok"}}
localhost Connection succeeded!
Subscribe to topics ( /thyme/co2 )
Subscribe to topics ( /thyme/tvoc )
Subscribe to topics ( /thyme/temp )
[status] : crtci
```

2. 이후 다른 터미널에서 tas\_emulator.js를 실행하여 co2, tvoc, temp에 대해 가상 데이터를 생성하고, 이를 nCube-Thyme-Nodejs에 전달하여 Mobius의 CIN으로 데이터를 전송합니다.

```
$ cd tas_sample/tas_emulator  
$ node tas_emulator.js
```

CSE 모니터링을 통해 각 센서 값 CNT에 데이터가 cin으로 적축



## 6.2. 실습 2

nCube-Thyme-Nodejs을 통해 Subscription을 생성하고 Mobius로부터 Notification을 수신하여 TAS-Emulator 까지 전달을 확인하는 실습

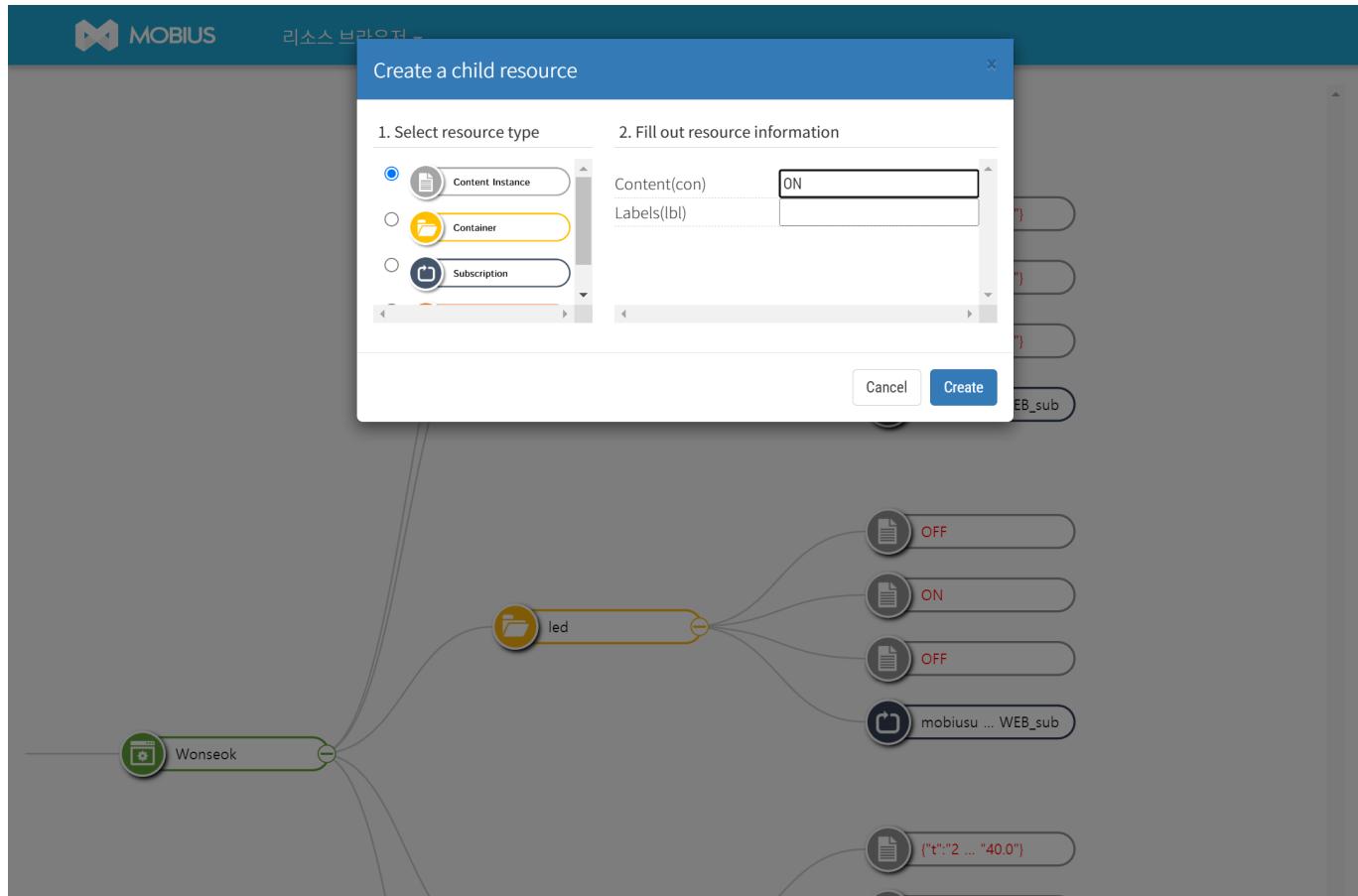
1. nCube-Thyme-Nodejs를 설치한 디렉토리로 이동 후 thyme.js 파일을 실행합니다.

```
$ cd nCube-Thyme-Nodejs  
$ node thyme.js
```

2. 다른 터미널에서 tas\_emulator.js를 실행하여 [/led/set](#) Topic에 Subscription을 생성하고, Mobius에 CIN 데이터가 업로드 되면 Mobius로 부터 Notification을 전달받습니다.

```
$ cd tas_sample/tas_emulator  
$ node tas_LED.js
```

```
localhost Connection succeeded!  
Subscribe to topics ( /led/set )  
Received ON from /led/set  
Received OFF from /led/set  
Received ON from /led/set  
Received OFF from /led/set  
Received ON from /led/set  
Received OFF from /led/set  
[]
```



## 6. 결론

본 문서에서 실제 센서를 대신하여 TAS-Emulator를 실행하여 co2, tvoc, temp에 대해 가상 데이터를 생성하고 데이터별 Topic을 정의하여 로컬 MQTT로 전송하였으며, Mobius와 연동되는 nCube-Thyme-Nodejs는 conf.js 파일에서 설정된 리소스 정보에 따라 AE, CNT, SUB을 생성하였고 TAS로부터 가상 데이터를 전달받아 파싱 후 CIN 생성을 통해 Mobius로 수집하였습니다. 또한, led라는 CNT에 ON/OFF 명령을 포함하는 CIN을 생성하였고 nCube-Thyme-Nodejs는 이를 Notification을 통해 전달받아 TAS-Emulator로 전송하였음을 확인해보았습니다.