IoTTi -projekti

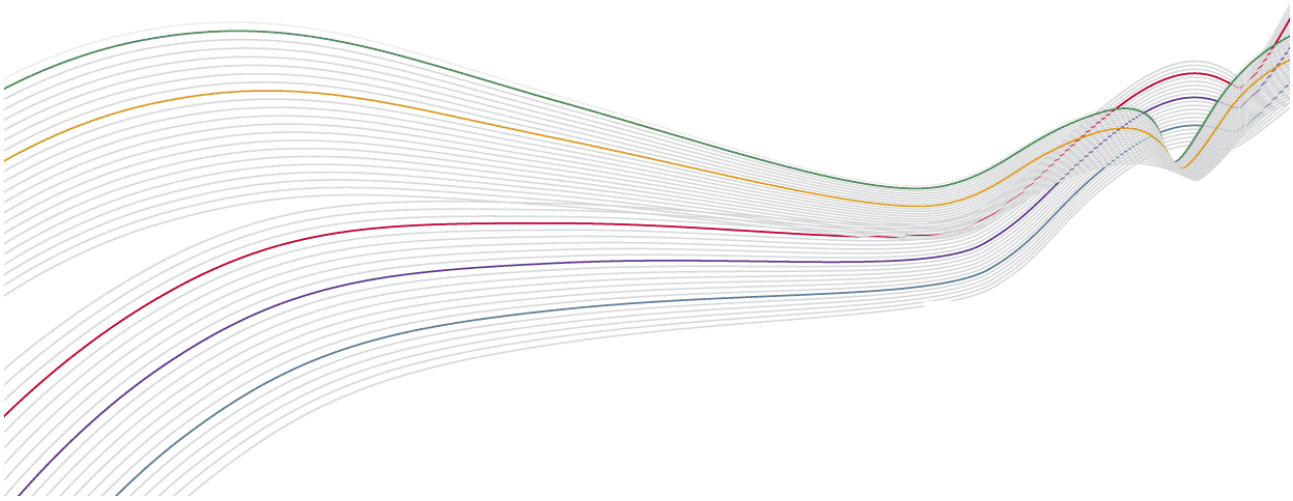
Euroopan unioni
Euroopan sosiaalirahasto

Metadata -dokumentti

1.0

Jukka Karlström

OAMK



Vipuvoimaa EU:lta 2014–2020

Euroopan unioni
Euroopan sosiaalirahasto

Sisällysluettelo

Vipuvoimaa EU:lta 2014–2020

Euroopan unioni
Euroopan sosiaalirahasto

1 Johdanto

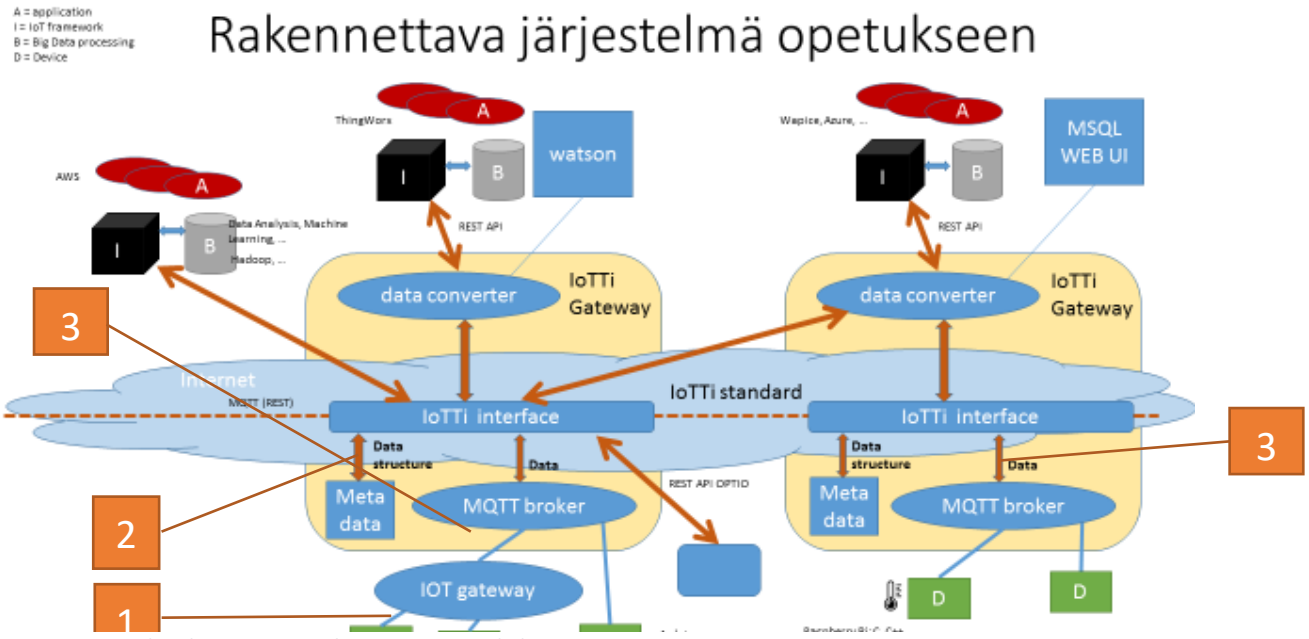
Dokumentissa kuvataan neljän korkeakoulun (Tamk, Hamk, Seamk ja Oamk) IoTTi –hankkeessa rakennetun IoT –tietoa tarjoavan järjestelmän toimintaa. Hankkeen yhdeksi tavoitteeksi on kirjattu ”toteutetaan verkostomallissa tarvittavat tekniset valmiudet kaikkien verkoston toimijoiden IoTympäristöjen hyödyntämiseen verkoston avulla toteutettavissa koulutuksissa”. Tämän dokumentin tarkoitus on määritellä mitä mainitut tekniset valmiudet ovat ja mitä vaatimuksia niiden on täytettävä.

Metadatalla tarkoitetaan sitä tietoa, jota tarvitaan jotta voidaan hyödyntää verkoston eri osapuolten tuottamaa dataa tai tarkemmin IoT:iin liittyvää anturitietoa. Metadata -dokumentti kertoo miten metadataa käytetään, miten anturidataa voidaan tuottaa, missä muodossa anturidata liikkuu ja on osapuolten käytettävissä. Metadata -dokumentissa kuvataan

1) yleinen toimintamalli
2) osapuolen palvelimella olevan metadatan esitystapa
3) lähetettävien sanomien formaatit laitteelta IOT-Gatewayhin

2 Yleinen rakenne

Jokaisella osallistujalla (korkeakoulu) on oma ns. IoTTi –liityntä (kuvassa IoTTi interface) Internettiin. IoTTi –liityntä on rajapinta, jonka kautta päästään korkeakoulun ylläpitämään MQTT –palvelimeen ja siihen liittyvään Metadata –tietokantaan. Metadata –tietokannan sisältönä on tieto siitä mitä tietoja ja missä muodossa on saatavilla ko. korkeakoulun MQTT –palvelimelta.



Kuva 1 Kari Naakan kuvaus järjestelmästä, jossa on kolme sanomavirtaa.

Vipuvoimaa EU:lta 2014–2020

Euroopan unioni
Euroopan sosiaalirahasto

Järjestelmä muodostuu siis osapuolten IoTTi -liitynnöillä varustetuista MQTT -palvelimista, jotka ovat keskenään yhteydessä Internetin kautta noudattaen MQTT -protokollaa. Järjestelmässä on oletuksena että vain IoT –laitteilta tulevia anturitietoja liikkuu palvelimien välillä eikä antureille tai ohjainlaitteille meneviä ohjaavia sanomia. Tieto siis kulkee vain antureilta poispäin.

MQTT –palvelimen toiminta perustuu tilaaja/julkaisija (subscriber/publisher) –malliin. Tilaaja tilaa MQTT –palvelimelta jonkin aiheen (topic) ja sen jälkeen palvelin työntää ko. aiheen viestit tilaajalle. Eli tässä tapauksessa korkeakoulun MQTT –palvelin tarjoaa tilattavaksi sellaisia aiheita joita sen IoT -laitteet muodostavat MQTT –palvelimelle. Kaikki tieto mitä tarjotaan MQTT –palvelimelta on JSON –muodossa.

Korkeakoulun MQTT –palvelimen ja siihen liitettyjen laitteiden väliset sanomat (numero 3 kuvassa) on kuvattu Metadatassa. Tieto (kuvassa 1) voi tulla varsinaiselta anturilta ensin IoT Gatewayhin, josta se lähtee edelleen MQTT –palvelimelle JSON -muodossa. Metadata kuvaa vain MQTT –palvelimelle tulevan sanoman, ei IoT –Gatewaylle laitteelta tulevia sanomia (numero 1 kuvassa).

Kuvassa numerolla 3 osoitettu yhteys tarkoittaa MQTT –palvelimelle meneviä tai sieltä tulevia sanomia. Sanomien rakenne on kuvattu palvelimella olevassa Metadata -tietokannassa.

3 Metadata –tietokanta

Metadata –tietokannan voidaan ymmärtää koostuvan varsinaisesta tarjottavasta IoT –tiedosta (anturitieto), tiedon esitysmallista ja aiherakenteesta sekä Metadatan sijainnista. Metadata on sijaittava paikassa, jonne on pääsy Internetin välityksellä. Se voi olla erillinen tietokanta joko yksittäisenä keskitettynä ratkaisuna tai hajautettuna erillisinä tietokantoina tietoa tarjoaviin organisaatioihin.

IoTTi -hankkeessa Metadata –tietokanta on hajautettuna IoT –liityntöjä tarjoavien ammattikorkeakoulujen palvelimille eikä sitä ole keskitetty millekään palvelimelle tai organisaatiolle. Tietokannat voidaan replikoida myös muille osapuolille. Metadata -tietokannan ylläpidosta vastaa jokainen osapuoli omalta osaltaan.

Metadatan tietovarasto, joka kuvaa varsinaisen IoT –tiedon esitystä ja tulkintaa, voidaan teknisesti toteuttaa palvelimille usealla eri tavalla. Se voi olla varsinainen tietokantapalvelinsovellus, esimerkiksi CouchDB –tietokanta, jossa tieto on tallennettu JSON –muodossa. Toinen kevyempi mahdollisuus on käyttää MQTT –palvelinta ja sen pysyvien viestien -ominaisuutta (retained messages). Nämä viestit ovat pysyviä MQTT –palvelimella tietyssä aiheessa ja kyseisen aiheen tilaus aiheuttaa aina siihen liitetyn viestin lähettämisen. IoTTi -verkostossa on valittu jälkimmäinen tapa, jossa pysyvät viestit sisältävät JSON Schema –tyyppisen kuvauksen, jonka perusteella viestit tulkitaan.

3.1 Metadatan toteutus MQTT –palvelimelle

MQTT –palvelimelle perustetaan vakioaihe Metadata/metadata. Tarvittaessa rakennetta voidaan kasvattaa. Tällä hetkellä metadatassa on vain määritelty se mitä tietoja on saatavilla. Lisäksi voitaisiin määritellä seuraavia asioita:

Vipuvoimaa EU:lta 2014–2020

Euroopan unioni
Euroopan sosiaalirahasto

- erilaiset oikeudet lukemiseen
- tarkempia kuvauksia laitteista
- laitteiden tilatiedot
- ym.

3.2 Metadata/metadata

JSON –muotoinen tiedosto, jossa on kuvattu kyseiseltä palvelimelta löytyvät aiheet (topic). Jokaisesta aiheesta löytyy nimi ja siihen liittyvä tiedon (sanoman) JSON Schema –muotoinen tiedon kuvaus. Alla on esimerkki tiedostosta ja sen osien selvennys.

```
[
    {
        "topic" : "oamk/teuvo/B321/temp/json/celcius",
        "schema" : {
            "$schema": "http://json-schema.org/draft-04/schema#",
            "type" : "object",
            "title" : "EnvSensor1 tEvent Schema",
            "description" : "defines the structure of a temperature event in degrees Celsius",
            "properties" : {
                "t" : {
                    "description" : "temperature in degrees Celsius",
                    "type" : "number",
                    "minimum" : -273.15,
                    "default" : 0.0
                }
            },
            "required" : ["t"]
        }
    },
    {
        "topic" : "oamk/teuvo/B322/temp/json/celcius",
        "schema" : {
            "$schema": "http://json-schema.org/draft-04/schema#",
            "type" : "object",
            "title" : "EnvSensor1 tEvent Schema",
            "description" : "defines the structure of a temperature event in degrees Celsius",
            "properties" : {
                "t" : {
                    "description" : "temperature in degrees Celsius",
                    "type" : "number",
                    "minimum" : -273.15,
                    "default" : 0.0
                }
            },
            "required" : ["t"]
        }
                                                                                {
]
```

Tiedostossa aiheet ovat taulukossa objekteina pilkulla erotettuina ja aaltosuluilla rajattuna.

Vipuvoimaa EU:lta 2014–2020

```
[
   {
               … aihe 1 …
   },
   {
               … aihe 2 …
   }
]
```

Jokainen aihe sisältää vähintään kaksi nimeä: topic ja schema. Nimen topic arvona on kyseinen aihe mqtt –palvelimella. Nimi schema puolestaan sisältää aiheen sisältävän sanoma kuvauksen, jonka avulla sanoman sisältö tulkitaan.

Ylläolevassa esimerkissä ”topic” -nimen arvona on tässä "oamk/teuvo/B321/temp/json/celcius". Tähän aiheeseen liittyvä tiedon esitys on kuvattu seuraavalla ”schema” –nimellä, jonka arvona on JSON Schema –muotoinen kuvaus aiheen tiedosta. Mainittu kuvaus noudattaa JSON Schema –määrittelyä.

3.3 JSON Schema kenttien käyttö

Avain	Kuvaus	Huomiot IoTTissa
$schema	Ilmoittaa schema version	http://json-schema.org/draft-04/schema#
type	Scheeman/Key tyyppi	object tai tiedon (Key) tyyppi
title	Scheeman otsikko	
description	Scheeman/Key kuvaus	Koko scheemalle kuvaus mitä tietoa sanomassa välitetään Yksittäisen tiedon (Key) kohdalla kuvaus voi sisältää tarkentavaa selitystä
properties	Scheemassa olevien tietojen (key) ja niiden arvojen kuvaus	esimerkiksi lämpötilalle ”t”
required	ominaisuus on pakollinen	
minimun	mahdollinen minimiarvo	
default	mahdollinen oletusarvo	

3.4 MQTT –palvelimen aihehierarkia

MQTT-palvelimen aihe on merkkijono joka voi muodostua useasta kauttaviivalla erotetusta hierarkiatasosta. Esimerkiksi *talo/olohuone/lämpötila*, jossa kuvataan talon olohuoneen lämpötila.

Esimerkkinä voisi olla Oamk:n MQTT –palvelimelta tilattava *teuvo/B321/lämpötila*. Tässä *teuvo* on rakennus ja B321 rakennuksen huone, jonka lämpötilaa lähetetään.

JSON –sanomien rakenne kuvataan Metadatassa JSON Schema -kuvausten avulla. Jokaisesta MQTT –palvelimelle tulevasta ja sieltä lähtevästä JSON –muodossa olevasta sanomasta täytyy löytyä Metadatasta JSON Schema –muotoinen kuvaus. JSON Schema kuvaa sanoman rakenteen JSON –kuvauksen avulla.

Euroopan unioni
Euroopan sosiaalirahasto

Aihe (topic):

oamk/teuvo/B322/lämpötila/kelvin

oamk/teuvo/B322/lämpötila/celsius

Aihehierarkkiaan voi sisällyttää sen formaattiin, jossa payload eli jos sen haluaa muussa kuin JSON –muodossa. Tällöin hierarkkia voisi olla:

oamk/teuvo/B322/lämpötila/json/kelvin

oamk/teuvo/B322/lämpötila/raw/kelvin

Tällöin pitää erikseen ohjelmoida muiden kuin JSON muotoisten aiheiden käsittely.

Sanoman sisältö(payload):

JSON mjono

JSON mjono n tulkinta riippuu metadatassa olevasta tulkintaohjeesta.

4 Toimintamalli

Kun halutaan luoda uusi sanoma omalle MQTT –palvelimelle, niin toimitaan seuraavasti:

1. Uudelle sanomalle luodaan json –schema kuvaus.
2. Jos MQTT –palvelimella on jo olemassa oleva metadata.json –tiedosto, niin uuden sanoman json –schema kuvaus liitetään siihen.
3. metadata.json –tiedosto lähetetään (julkaistaan) MQTT –palvelimelle. (Ohjelma md-writemetadata.py).

Jokaisella MQTT-palvelimella on pysyvä (retained) sanoma aiheessa Metadata/metadata. Sanoman sisältö on tiedosto metadata.json. Sanomien formaattien kuvaukset ovat JSON Schema kuvauksina mainitussa metadata.json –tiedostossa. Mainitussa tiedostossa on sanoman formaatin kuvauksen lisäksi sanoman aihe eli topic.

4.1 Sanomien formaattien ylläpito metadata.json –tiedostossa

Kaksi tiedostossa metadata.json olevaa lämpötila –tiedon sisältävän sanomaa JSON Schema –muotoisella kuvauksella ja niihin liitetyt aiheet (ks. 3.3):

Vipuvoimaa EU:lta 2014–2020

Euroopan unioni
Euroopan sosiaalirahasto

```
[
    {
        "topic" : "oamk/teuvo/B321/temp/json/celcius",
        "schema" : {
            "$schema": "http://json-schema.org/draft-04/schema#",
            "type" : "object",
            "title" : "EnvSensor1 tEvent Schema",
            "description" : "defines the structure of a temperature event in degrees Celsius",
            "properties" : {
                "t" : {
                    "description" : "temperature in degrees Celsius",
                    "type" : "number",
                    "minimum" : -273.15,
                    "default" : 0.0
                }
            },
            "required" : ["t"]
        }
    },
    {
        "topic" : "oamk/teuvo/B322/temp/json/celcius",
        "schema" : {
            "$schema": "http://json-schema.org/draft-04/schema#",
            "type" : "object",
            "title" : "EnvSensor1 tEvent Schema",
            "description" : "defines the structure of a temperature event in degrees Celsius",
            "properties" : {
                "t" : {
                    "description" : "temperature in degrees Celsius",
                    "type" : "number",
                    "minimum" : -273.15,
                    "default" : 0.0
                }
            },
            "required" : ["t"]
        }
   }
]
```

Ylläolevan kuvauksen mukainen varsinainen sanoma:

```
{
  "t" : 34.5
}
```

Tiedoston metadata.json ylläpidon voi toteuttaa manuaalisesti tai automaattisesti. Manuaalisesti tehtäessä muokataan kyseistä tekstimuotoista tiedostoa. Automaattisesti tiedoston sisältö voidaan tuottaa erilaisilla verkosta löytyvillä työkaluilla.

Kun tiedostoa metadata.json on muokattu, niin sen sisältö pitää päivittää MQTT –palvelimelle Metadata/metadata aiheeseen uudeksi pysyväksi sanomaksi. Tämä tapahtuu Python –kielisellä metadatatotopic.py –ohjelmalla. Kyseinen ohjelma lukee metadata.json –tiedoston sisällön ja lähettää sen sisällön sanomaksi aiheeseen Metadata/metadata.

4.2 Uuden anturitiedon lisäys

1. Päivitetään metadata.json tiedostoa vastaamaan uutta tai muokattua anturin sanomaa

Vipuvoimaa EU:lta 2014–2020

Euroopan unioni
Euroopan sosiaalirahasto

2. MQTT –palvelimelle päivitetään uuden anturitiedon sisältävä metadata.json –tiedosto metadatatotopic.py –ohjelmalla.
3. Kirjoitetaan ohjelma, joka lähettää anturitietoa edellä mainitussa JSON Schemassa kuvatussa formaatissa.

4.3 Tietojen lukeminen MQTT –palvelimelta

1. Luetaan ohjelmalla md_topicstofile.py –ohjelmalla MQTT –palvelimelta sanoma Metadata/metadata –aiheesta.
2. Tuloksena syntyy tiedosto metadata_out.json, jossa on asanomien formaatit JSON Schema kuvauksena ja niihen liitetyt aiheet.

4.4 Sanomien tulkinta

Sanomat luetaan esimerkiksi ohjelmalla MQTT –palvelimelta ja ohjelmassa hyödynnetään metadata_out.json tiedostoa sanomien tulkinnassa. Tämä voidaan toteuttaa eri ohjelmointikielillä.

5 Laitteella tapahtuva käsittely

Käsitellään ennen datan lähettämistä pilveen. Käsittely sisältää mm.

- tiedon muuntaminen eri formaattiin
- paketointi turvalliseksi ja käytännöllisiin paketteihin
- tiedon validointi sääntöjen avulla
- tiedon mahdollinen järjestäminen haluttuun järjestykseen
- laajennetaan tietoa mahdollisella lisätiedolla
- tiedosta laaditaan yhteenveto määrän vähentämiseksi ja tarpeettomien yksityiskohtien poistamiseksi
- yhdistetään tietoa isommiksi kokonaisuuksiksi

6 Esimerkki lämpötilan ja kosteuden lähettämisestä

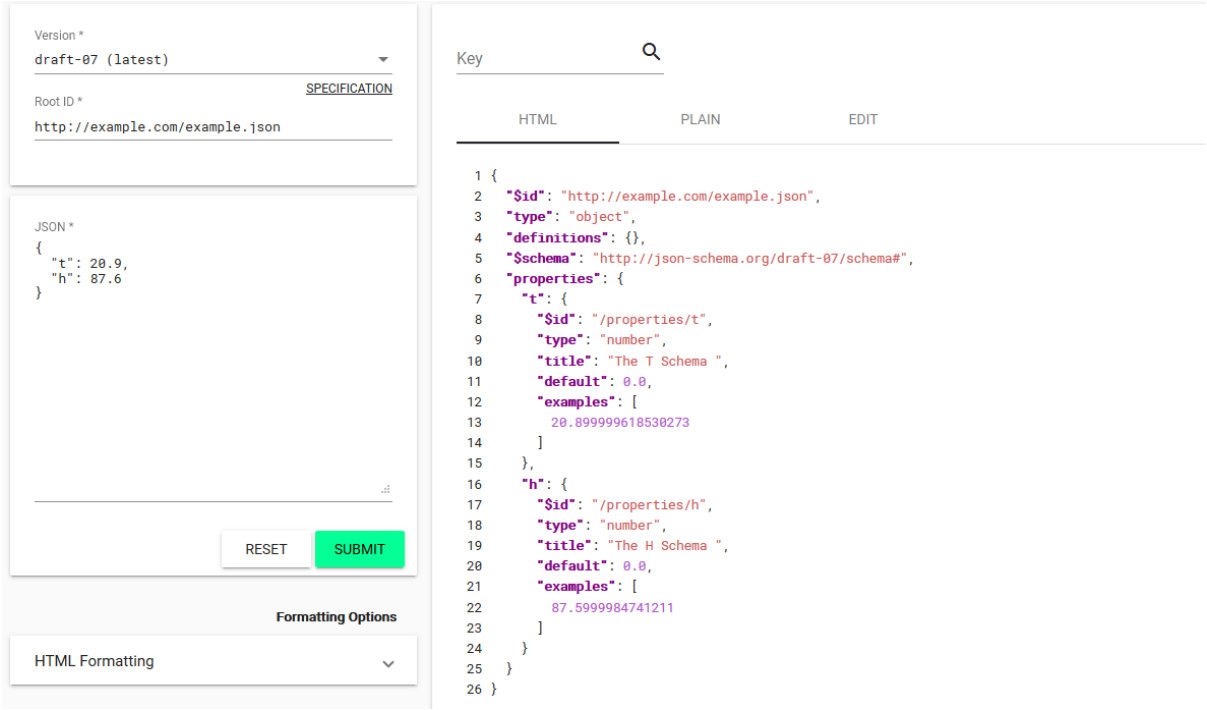
Esimerkkinä on lämpötila- ja kosteustiedon lähettäminen huoneesta B322. Muodostetaan lämpötilalle ja kosteudelle JSON Schema –kuvaukset. Lämpötila lähetetään Celsius asteina ja suhteellinen kosteus prosentteina.

Vipuvoimaa EU:lta 2014–2020

Euroopan unioni
Euroopan sosiaalirahasto

6.1 Metadata MQTT -palvelimelle

Kirjoitetaan tekstieditorilla JSON Schema kuvaus tiedostoon huoneb322.json tai käytetään netistä löytyviä onlinen –työkaluja. Alla on kuvakaappaus https://jsonschema.net/ -sivulta löytyvän työkalun käytöstä. Vasemmalla on JSON –muotoinen sanoma ja sen oikealle puolelle tulee sitä vastava JSON Scheema kuvaus. Kuvausta voidaan muokata vielä ja laittaa sitten tiedostoon.



Kopioidaan teksti (plain –muoto) tiedostoksi huoneb322.json. Alla alkuperäinen muoto

Vipuvoimaa EU:lta 2014–2020

```
{
  "$id": "http://example.com/example.json",
  "type": "object",
  "definitions": {},
  "$schema": "http://json-schema.org/draft-07/schema#",
  "properties": {
    "t": {
      "$id": "/properties/t",
      "type": "number",
      "title": "The T Schema ",
      "description": "An explanation about the purpose of this instance.",
      "default": 0
    },
    "h": {
      "$id": "/properties/h",
      "type": "number",
      "title": "The H Schema ",
      "description": "An explanation about the purpose of this instance.",
      "default": 0
    }
  }
}
```

Koska yo. tiedostossa lämpötila ja suhteellinen kosteus ovat yhdessä, niin tehdään niistä erilliset kuvaukset ja muokataan tiedosto seuraavaan muotoon:

```
{
  "type": "object",
  "description": "Huoneen B322 ilman lämpötila ja kosteustiedot.",
  "$schema": "http://json-schema.org/draft-07/schema#",
  "properties": {
    "t": {
      "type": "number",
      "title": "The T Schema ",
      "description": "Huoneen lämpötila Celsius asteina",
      "default": 0
    },
    "h": {
      "type": "number",
      "title": "The H Schema ",
      "description": " Huoneen suhteellinen kosteusprosentti.",
      "default": 0
    }
  }
}
```

Jos palvelimella on jo Metadata/metadata sanoma, niin yo. tiedoston sisältämä teksti voidaan lisätä siihen ja päivittää muut tiedot. Yo. JSON –teksti on yhden sanoman kuvaus. Tässä esimerkissä eri tiedot lähetetään erillisinä sanomina ja sen vuoksi lopulliseen metadata.json tiedostoon ne täytyy eriyttää ja niihin voi laittaa myös selventäviä avainarvoja, kuten esimerkiksi ”required”.

Kun on saatu valmiiksi metadata.json tiedosto, joka sisältää MQTT –palvelimen tarjoamien sensori sanomien kuvaukset eli metadatan, niin siirretään ko. tiedosto MQTT –palveliemlle aiheeseen Metadata/metadata. Alla on metadata.json tiedosto ja python koodi, jolla tiedosto siirtyy palvelimelle pysyväksi sanomaksi (retained).

metadata.json –tiedosto:

```
[
  {
    "topic":"oamk/teuvo/B322/temp/json/celcius",
    "schema":
      {
      "type": "object",
      "description": "Huoneen B322 ilman lämpötila.",
      "$schema": "http://json-schema.org/draft-07/schema#",
      "properties": {
        "t": {
          "type": "number",
          "title": "The T Schema ",
          "description": "Huoneen lämpötila Celsius asteina",
          "default": 0
        }
       }
    }
  },
  {
    "topic":"oamk/teuvo/B322/humidity/json/percent",
    "schema":
    {
      "type": "object",
      "description": "Huoneen B322 ilman kosteustiedot.",
      "$schema": "http://json-schema.org/draft-07/schema#",
      "properties": {
        "h": {
          "type": "number",
          "title": "The H Schema ",
          "description": " Huoneen suhteellinen kosteusprosentti.",
          "default": 0
        }
       }
    }
  }
]
```

Python –koodi tiedoston läettämiseksi MQTT –palvelimelle Metadata/metadata –aiheeksi (topic):

```
#!/usr/bin/env python3
"""Convert metadata.json to mqtt topic Metadata/metadata with retained flag set
"""

import json
import paho.mqtt.client as mqtt

# Open file and load data as json format
with open('metadata.json') as json_data:
    data = json.load(json_data)
#    print(data)

# Open connection to mqtt -server and write data to specific topic using retain flag
client = mqtt.Client()
client.connect("localhost",1883,60)
client.publish("Metadata/metadata", json.dumps(data), qos=0, retain=True);
client.disconnect();
```

Yllä olevassa koodissa metadata.json –tiedoston sisältö muunnetaan sopivasti sanomaksi ja lähetetetään MQTT –palvelimelle, joka tässä tapauksessa on paikallinen. Sanoma lähetetään retained –lippu asetettuna, jolloin se pysyy siellä kunnes uusi on lähetetty tilalle.

Jos halutaan sekavasta JSON –sanomasta seklkeämmin luettava versio, niin voi käyttää netti –osoitteessa http://jsonprettyprint.com./ olevaa apuria tekstin muotoilemiseksi selkeämmäksi.

6.2 Sensoritiedon lähetys MQTT –palvelimelle

Alla on Python koodi sensoritietojen lähettämiseksi MQTT –palvelimelle. Koodissa generoidaan satunnaisluvuista sensoriarvoja. Koodissa ei myöskään hyödynnetä validointi eli ei käytetä scheema tiedon oikeellisuuden varmistamiseksi.

```
#!/usr/bin/env python3
"""Test program to write certain topic to mqtt server

Lack of validation
No actual sensors
Using uniform to generate random values as sensor data

"""

import json
from random import uniform
import time
import paho.mqtt.client as mqtt

# Open file (metadata.json) and load data as json format
with open('metadata.json') as json_data:
    data = json.load(json_data)
    print("Käydään läpi kaikki json -tiedostossa luetteloidut aiheet ja niihin liittyvät schemat")
    for x in data:
        for y in x:
            print(y, " :", x[y])


# Open connection to local mqtt -server and write data to specific topic using retain flag
client = mqtt.Client()
client.connect("localhost",1883,60)

# # generate ten (10) sensor values using random generator every second
for p in range(10):
    temp = round(uniform(19,  25 ), 1) # Temperature values are between 19 and 25 and with one decimal
    humidity = round(uniform(50, 85), 0) # Humidity is beteween 50% and 85% with zero decimal
    time.sleep(1)

    client.publish("oamk/teuvo/B322/temp/json/celcius", json.dumps({'t': temp}));
    client.publish("oamk/teuvo/B322/humidity/json/percent", json.dumps({'h': humidity}));

client.disconnect();
```