# TP 2 : Internet des objets - collecte et traitements de données EmonPi à l'aide de MQTT et Node-RED

⏱ 3h
Réseaux de capteurs
*Mis à jour le 26/11/2025*

## 🎯 Objectifs

- Savoir développer des applications IoT à l'aide des protocoles et outils logiciels (MQTT, Node-RED)
- Savoir collecter et transmettre des données de capteurs d'EmonPi pour la consommation de l'énergie d'une charge électrique (ici un écran)

## 🔧 Matériel nécessaire

- Ordinateur avec Python 3.x, mosquitto et Node-RED installé
- Accès au réseau

> **Important**
>
> L'évaluation de ce TP se fait pendant la séance. Pensez à faire valider régulièrement votre avancement auprès de l'enseignant.

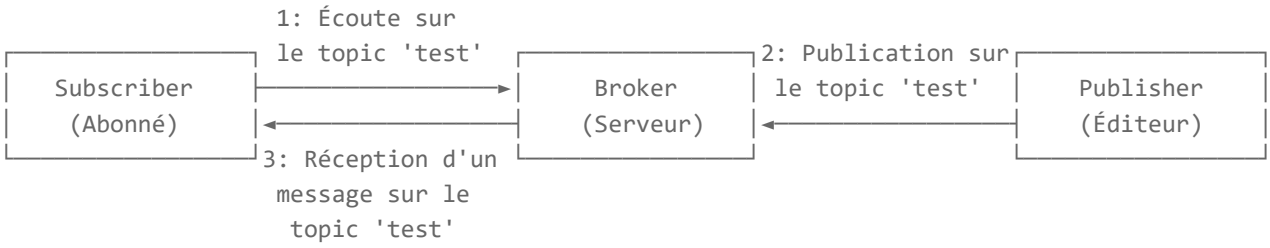
## Présentation du TP

Dans ce TP, nous allons utiliser MQTT pour récupérer les données de consommation d'un écran situé en salle 204 rouge. À partir de ces données, nous générerons un graphique de consommation, puis nous calculerons la consommation cumulée afin d'en estimer le coût total depuis le début du monitoring.

Pour cela, nous interrogerons les capteurs installés en salle 204 rouge via le protocole MQTT, présenté ci-dessous.

## Partie 1 : MQTT

Cette partie se focalise sur l'initiation au protocole MQTT. Le schéma ci-dessous permet de comprendre très rapidement son fonctionnement.




> **Important**
>
> Tout au long de ce TP, **le subscriber doit toujours écouter sur le même topic et utiliser le même broker que le publisher**, afin d'assurer la bonne réception des messages.

**Q1 (1 pt).** Dans une fenêtre PowerShell, lancez un subscriber sur le topic `testtopic` en utilisant le broker local (`localhost`):

```
mosquitto_sub -h localhost -t "testtopic" -v
```

Cette commande place le subscriber en attente d'un message publié sur ce topic.

> **Note**
>
> Utilisez `ctrl+c` dans Powershell pour arrêter un programme en cours d'exécution.

Dans un autre onglet PowerShell, publiez un message sur le même topic et le même broker:

```
mosquitto_pub -h localhost -t "testtopic" -m "Testing"
```

**Q2 (1 pt).** Votre message précédent a été transféré par le broker local de votre machine. Cependant, il existe aussi des brokers publics accessibles sur Internet, sur lesquels tout le monde peut publier et souscrire à des topics.

Dans cette question, vous allez utiliser le broker public proposé à l'adresse: `test.mosquitto.org`.

L'unique modification à effectuer dans vos commandes est de remplacer `localhost` par `test.mosquitto.org`.

1. Dans votre PowerShell, souscrivez au topic `/tprc/{groupe}/{nom}`

où:

- `{groupe}` correspond à votre lettre de groupe
- `{nom}` correspond à votre nom

  Commande:

  ```
  mosquitto_sub -h test.mosquitto.org -t "/tprc/{groupe}/{nom}" -v
  ```

1. Dans un autre onglet PowerShell, publiez un message sur ce même topic:

```
mosquitto_pub -h test.mosquitto.org -t "/tprc/{groupe}/{nom}" -m "Ceci est un message"
```

1. Essayez à présent de vous envoyer des messages entre groupes. Pour cela, modifiez simplement votre commande subscriber ou publisher afin d'utiliser le topic d'un autre binôme.

# Partie 2 : Node-RED

Node-RED est un outil de programmation visuelle qui permet de créer des workflows en connectant des blocs (*nodes*) entre eux, idéal pour l'IoT et l'automatisation sans écrire beaucoup de code.

Dans un onglet PowerShell, lancez Node-RED avec la commande `node-red`. Une fois démarré, accédez à son interface à l'adresse :

👉 **http://localhost:1880/**

Vous pourrez alors construire votre première application IoT.

> 💡 Tip
>
> Il vous sera demandé d'ouvrir l'onglet Debug. Pour cela, cliquez sur l'icône en forme de petit insecte (bug), située dans la barre d'onglets en haut à droite de l'interface Node-RED.

Pour vous familiariser avec Node-RED, réalisez les deux tutoriels officiels suivants :

- **Créer votre premier flow** : https://nodered.org/docs/tutorials/first-flow
- **Créer votre second flow** : https://nodered.org/docs/tutorials/second-flow

Avec Node-RED, il est possible de souscrire et de publier facilement des topics MQTT grâce aux nodes disponibles dans la section **Network** :

- `mqtt-in` pour jouer le rôle de *subscriber*
- `mqtt-out` pour jouer le rôle de *publisher*

**Q1 (1 pt).** Ajoutez un node `mqtt-in`, puis double-cliquez dessus pour ouvrir la fenêtre de configuration. Cliquez sur l'icône `+` afin de créer une nouvelle connexion MQTT, puis renseignez:

- **Serveur** : `localhost` (ou `test.mosquitto.org`)
- **Port** : `1883`

Validez la configuration.

Saisissez ensuite dans le champ **topic** le topic auquel vous souhaitez vous abonner.

Ajoutez un node `debug`, reliez-le au node `mqtt-in`, puis déployez votre flow.

Dans PowerShell, publiez un message sur ce même topic avec un publisher. Le message devrait apparaître dans l'onglet **Debug** de Node-RED.

**Q2 (1 pt).** Ajoutez également un node `mqtt-out` dans le même flow. Saisissez **le même topic et broker** que ceux configurés à la question précédente.

Connectez un node `inject` à ce `mqtt-out` pour envoyer un message. Déployez le flow, puis observez :

- Dans votre **Powershell** à l'aide d'un subscriber MQTT.
- Dans l'onglet **Debug** de Node-RED.

Vous devriez voir, dans les deux cas, le même message contenant le temps au format Linux, être reçu correctement.

**Q3 (2 pts).** Avec Node-RED, il est également possible d'afficher les données des capteurs grâce au tableau de bord *(node-red-dashboard)*.

1. Dans Node-RED, reliez une `gauge` à la sortie MQTT de votre flow précédent.

> ⚠ Warning

> La **première** fois que vous ajoutez un élément graphique (gauge, chart, etc.), il faut créer un groupe. Pour cela, cliquez sur `+`. Une fenêtre de création de groupe s'ouvrira. Si le groupe nécessite la création d'un autre élément, cliquez de nouveau sur `+`. Une fois le groupe créé, vous pouvez terminer la configuration sans problème.

> **Tip**
>
> Pour ouvrir le dashboard, cliquez sur le petit triangle pointant vers le bas en haut à droite et sélectionnez Dashboard. Ensuite, cliquez sur le carré avec une flèche vers le coin supérieur droit pour ouvrir la page du dashboard dans votre navigateur.

1. Simulez la publication de valeurs de température avec votre publisher et observez l'affichage sur la gauge.
2. Ajoutez également un graphique (*chart*) pour visualiser l'évolution d'un topic `temperatures` en fonction du temps.

Cela permet de suivre en temps réel les variations des données sur le dashboard.

**Q4 (4 pts).** Dans cet exercice, nous allons apprendre à publier et souscrire à des topics depuis un programme Python en utilisant la librairie paho-mqtt.

1. Vérifiez d'abord que paho-mqtt est installé sur votre machine en exécutant cette commande dans Powershell:

```
pip install paho-mqtt
```

1. Modifiez ensuite le code Python pour publier la séquence 1,2,3,4,5,4,3,2,1 à un intervalle de 1 seconde sur un topic MQTT:

```python
import paho.mqtt.client as mqtt

# Configuration du broker
broker = "localhost"  # ou "test.mosquitto.org"
port = 1883
topic = "testtopic"

# Callback appelé à la réception d'un message
def on_message(client, userdata, msg):
    print(f"Message reçu sur {msg.topic}: {msg.payload.decode()}")

# Création du client MQTT
client = mqtt.Client()

# Connexion au broker
client.connect(broker, port)

# Souscription au topic
client.subscribe(topic)

# Publication d'un message
message = "Hello"
client.publish(topic, message)

print(f"Message publié sur le topic '{topic}': {message}")

# On attend 5 secondes
time.sleep(5)

# Déconnexion du broker
client.disconnect()
```

Afficher cette série de valeurs dans un chart (courbe) de node-RED.

## Partie 3 : Open Energy Monitor et emonPi

Open Energy Monitor est un projet open source dédié au monitoring de la consommation d'énergie à partir (https://openenergymonitor.org/).

Dans cette partie du TP, nous allons utiliser emonPi pour récupérer et analyser les données de consommation d'un écran (https://guide.openenergymonitor.org/setup/).

Nous avons préalablement connecté plusieurs **EmonPi** dans la salle 204 rouge sur le réseau local Ethernet de l'école. Chaque EmonPi publie les données de ses capteurs sur **son broker local** aux adresses suivantes: 100.64.212.100, 100.64.212.103, 100.64.212.94, 100.64.212.95, 100.64.212.96.

**Q1 (2 pts).** Créez un subscriber sur votre PC qui s'abonne à tous les topics publiés par un EmonPi. Quels sont les différents topics que l'EmonPi publie ?

**Q2 (2 pts).** Éditez un flow en ajoutant un node `mqtt-in` et connectez-le à un node `debug`. Configurez le broker MQTT avec l'adresse IP de votre EmonPi (par exemple : `100.64.212.100`). Déployez ensuite le flow.

Que voyez-vous apparaître dans la fenêtre Debug ?

**Q3 (6 pts).** Nous souhaitons créer un historique de la consommation de notre écran à l'aide d'un tableau de bord Node-RED. Celui-ci devra afficher plusieurs informations.

1. Commencez par afficher la consommation électrique instantanée (en Watts). Ces données sont publiées sur le topic :

```
/emon/emonpi/power1
```

Souscrivez à ce topic avec un node `mqtt-in`, puis affichez la valeur dans un **graphique (chart)**.

1. Nous souhaitons maintenant calculer la consommation cumulée en KWh au fil du temps.

Pour cela, ajoutez un node `function` que vous configurerez avec le code suivant. Dans au démarrage:

```
var kwh = 0;
global.set("kwh", kwh);
```

Dans à chaque message:

```
var watt = parseInt(msg.payload);
var kw = watt/1000;
var kwh_ = global.get("kwh") + kw*(3600/5);
global.set("kwh", kwh_);

msg.payload = kwh_;

return msg;
```

Affichez ensuite le résultat (KWh cumulés) dans un node `chart`.

1. Enfin, nous voulons afficher le coût total de l'énergie consommée. Le prix unitaire de l'électricité est :

```
1 KWh = 0,1841 €
```

Ajoutez un node `function` supplémentaire pour calculer le coût. Le code est volontairement laissé libre pour que vous le complétiez vous-même.

Affichez cette valeur dans un node `text`.

---

*Rédigé par Matthieu Amet*

---

Thomas Mongaillard
thomas.mongaillard@univ-lorraine.fr

Supports de TP/TD à l'ENSEM