Module SCI

Ecole Nationale Supérieure d'Informatique d'Alger (Ex. ini)

Compte rendu TP10

06 Mai 2022

Objectifs

1. **Partie théorique :**
    - Explorer node-red et MQTT
2. **Partie pratique :**
    - Architecture du système
    - Installation sur raspberry-pi
    - Visualiser et accéder aux données via BLYNK

Partie théorique

A. Node-red

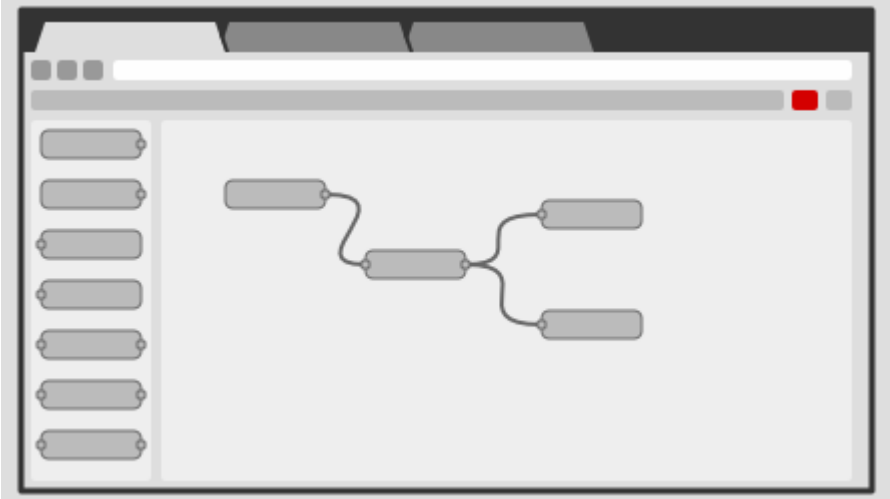
Node-RED est un outil de programmation permettant de relier des dispositifs matériels, des API et des services en ligne de manière nouvelle et intéressante. Il fournit un éditeur basé sur un navigateur qui facilite le câblage des flux en utilisant la large gamme de nœuds dans la palette qui peut être déployée dans son runtime en un seul clic.

Edition de flux par navigateur

Node-RED fournit un éditeur de flux basé sur un navigateur qui facilite l'assemblage des flux en utilisant la large gamme de nœuds de la palette. Les flux peuvent ensuite être déployés vers le runtime en un seul clic. Les fonctions JavaScript peuvent être créées dans l'éditeur à l'aide d'un éditeur de texte riche. Une bibliothèque intégrée vous permet d'enregistrer des fonctions, des modèles ou des flux utiles pour les réutiliser.

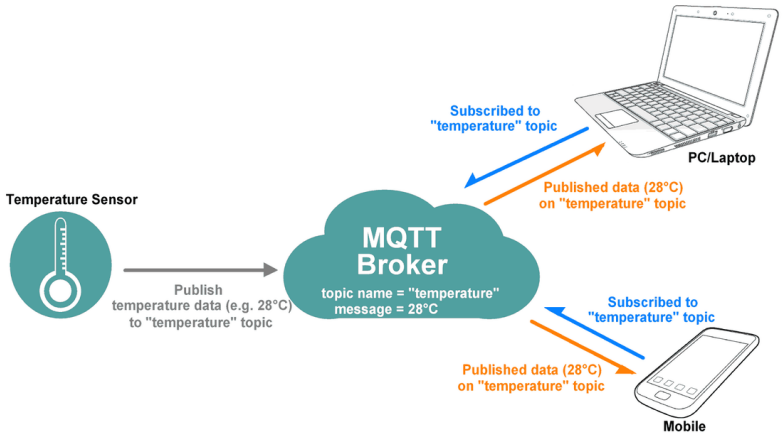
Construit sur Node.js

Le moteur d'exécution léger est construit sur Node.js, tirant pleinement parti de son modèle événementiel et non bloquant. Il est donc idéal pour fonctionner à la périphérie du réseau, sur du matériel à faible coût tel que le Raspberry Pi, ainsi que dans le nuage. Avec plus de 225 000 modules dans le dépôt de paquets de Node, il est facile d'étendre la gamme de nœuds de palette pour ajouter de nouvelles capacités.



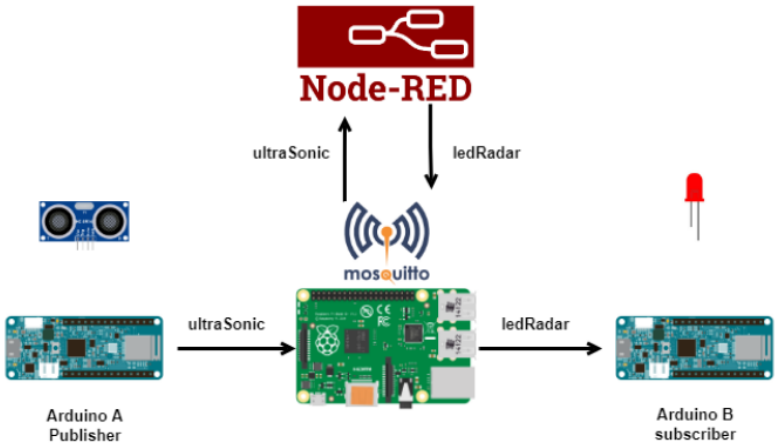
B. MQTT

MQTT est un protocole standardisé reposant sur TCP/IP. "Il est particulièrement utilisé pour transporter des données des objets connectés sur le cloud. Néanmoins, il reste lourd pour les réseaux LPWA contraints type NB-IoT et ne permet d'adresser que la remontée de données sans prendre en compte les services de device management. Ces derniers étant essentiels lors de déploiements massifs d'appareils connectés", souligne Hatem Oueslati, CEO d'IoTerop, start-up française spécialisée dans le device management. Son processus se divise en quatre étapes distinctes : connexion, authentification, communication, terminaison. MQTT permet la gestion des déconnexions et des reconnexions de devices de manière simplifiée. La taille maximale d'un message envoyé avec MQTT est de 256 Mo.



Partie pratique

C. Architecture général du système :



- Le protocole mqtt s'exécutera sur la carte raspberry et communiquera avec le flux sur Node-red selon 2 topics ultraSonic et ledRadar.
- Un arduino mkr 1010 jouera le rôle d'un publisher dans le topic ultraSonic et un autre d'un subscriber dans le topic ledRadar.

D. Installation de MQTT sur la raspberry

- *Taper la ligne suivant et sélectionner'Change User Password' si on veut rajoueter un mot de passe:*

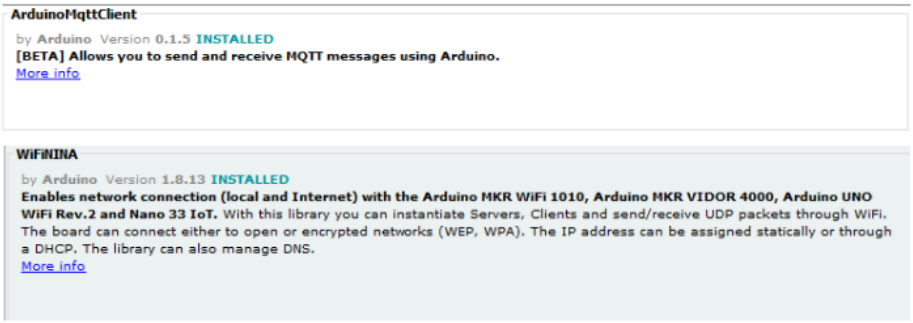
  sudo raspi-config

- *sudo apt-get update*
- *Faîtes les mises à jour nécessaires et passer à l'installation de notre* ***broker MQTT****.*

  sudo apt-get install mosquitto

- *systemctl status mosquitto*
- *sudo systemctl enable mosquitto.service*
- *sudo nano /etc/mosquitto/mosquitto.conf*
- *Modifier les lignes suivantes :*
  - *allow_anonymous true*
  - *listener 1883*
- *Normalement si tout est bien installé on aura le broker mqtt exposé sur l'adresse IP : 1883*

E. Installation requise sur le arduino :



F. Installation de node-red :

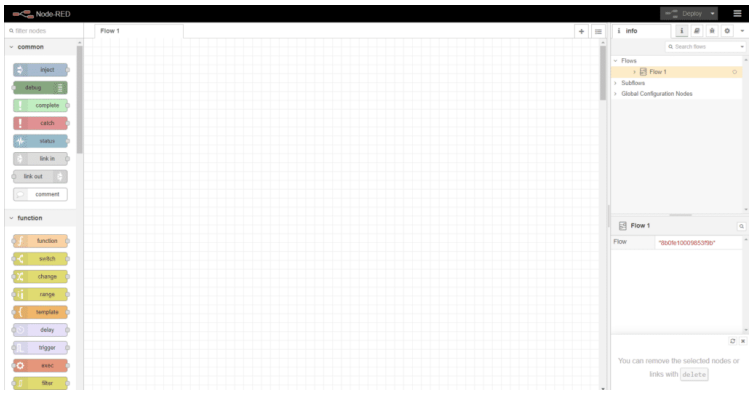
- Pour lancer l'installation de NodeRed avec toutes les dépendances nécessaires, il existe un script. Pour le lancer, il suffit de taper cette commande :

bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered

```
Running Node-RED install for user pi at /home/pi on raspbian


This can take 20-30 minutes on the slower Pi versions - please wait.

  Stop Node-RED                       
  Remove old version of Node-RED      
  Remove old version of Node.js       
  Install Node.js 14 LTS              v14.17.6   Npm 6.14.15
  Clean npm cache                     
  Install Node-RED core               2.0.6
  Move global nodes to local          
  Npm rebuild existing nodes          
  Install extra Pi nodes
  Add shortcut commands
  Update systemd script


Any errors will be logged to   /var/log/nodered-install.log
```

- Pour lancer node-red il suffit de taper une des 2 commandes suivantes :

  node-red ou node-red-start



G. Le code sur le arduino publisher :

Blink §

```
#include <ArduinoMqttClient.h>
#include <WiFiNINA.h>
//Pin capteurs
#define echoPin 2
#define trigPin 3
int distance;
//Spécifications Wifi et Broker
char ssid[] = "Iphone de Linda";
char pass[] = "password2000";
WiFiClient wifiClient;
MqttClient mqttClient(wifiClient);
const char broker[] = "192.168.1.11";
int port = 1883;
const char topic[] = "ultraSonic";
const long interval = 1000;
unsigned long previousMillis = 0;
void setup() {
Serial.begin(9600);
while (!Serial) {
; // ne pas lancer le programme avant d'avoir ouvert le moniteur
}
// Se connecter au Wifi
Serial.print("Attempting to connect to WPA SSID: ");
Serial.println(ssid);
while (WiFi.begin(ssid, pass) != WL_CONNECTED) {
// Erreur, réessayer
Serial.print(".");
delay(5000);
}
Serial.println("You're connected to the network");
Serial.println();
Serial.print("Attempting to connect to the MQTT broker: ");
Serial.println(broker);
//Connexion au broker a échoué
if (!mqttClient.connect(broker, port)) {
Serial.print("MQTT connection failed! Error code = ");
Serial.println(mqttClient.connectError());
while (1);
}
Serial.println("You're connected to the MQTT broker!");
Serial.println();
//initialiser les pin du capteur
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
}
void loop() {
// Envoyer des messages Keepalive au broker pour rester connecté
```

```
mqttClient.beginMessage(topic); //envoie du message au topic
mqttClient.print(distance);
mqttClient.endMessage();
Serial.println();
}
TEST !
}
//fonction de calcul de distance à partir du capteur
int captureDistance()
{
long duration;
int distancetmp;
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distancetmp = duration * 0.034 / 2;
return distancetmp;
}
```

H. Le code sur le arduino subscriber :

Blink §

```
#include <ArduinoMqttClient.h>
#include <WiFiNINA.h>
//Configuration Wifi et broker
char ssid[] = "Iphone de Linda";
char pass[] = "password2000";
WiFiClient wifiClient;
MqttClient mqttClient(wifiClient);
const char broker[] = "192.168.1.11";
int port = 1883;
const char topic[] = "ledRadar";
int ledPin = 7;
int message = 0;
void setup() {
Serial.begin(9600);
while (!Serial) {
; // wait for serial port to connect. Needed for native USB port only
}
// se connecter au réseau
Serial.print("Attempting to connect to WPA SSID: ");
Serial.println(ssid);
while (WiFi.begin(ssid, pass) != WL_CONNECTED) {
// failed, retry
Serial.print(".");
delay(5000);
}
Serial.println("You're connected to the network");
Serial.println();
//se connecter au broker MQTT
Serial.print("Attempting to connect to the MQTT broker: ");
Serial.println(broker);
if (!mqttClient.connect(broker, port)) {
Serial.print("MQTT connection failed! Error code = ");
Serial.println(mqttClient.connectError());
while (1);
}
Serial.println("You're connected to the MQTT broker!");
Serial.println();
Serial.print("Subscribing to topic: ");
Serial.println(topic);
Serial.println();
// Se souscrire au Topic
mqttClient.subscribe(topic);
Serial.print("Waiting for messages on topic: ");
Serial.println(topic);
Serial.println();
//initaliser la pin connectée à la led
pinMode(ledPin, OUTPUT);
```

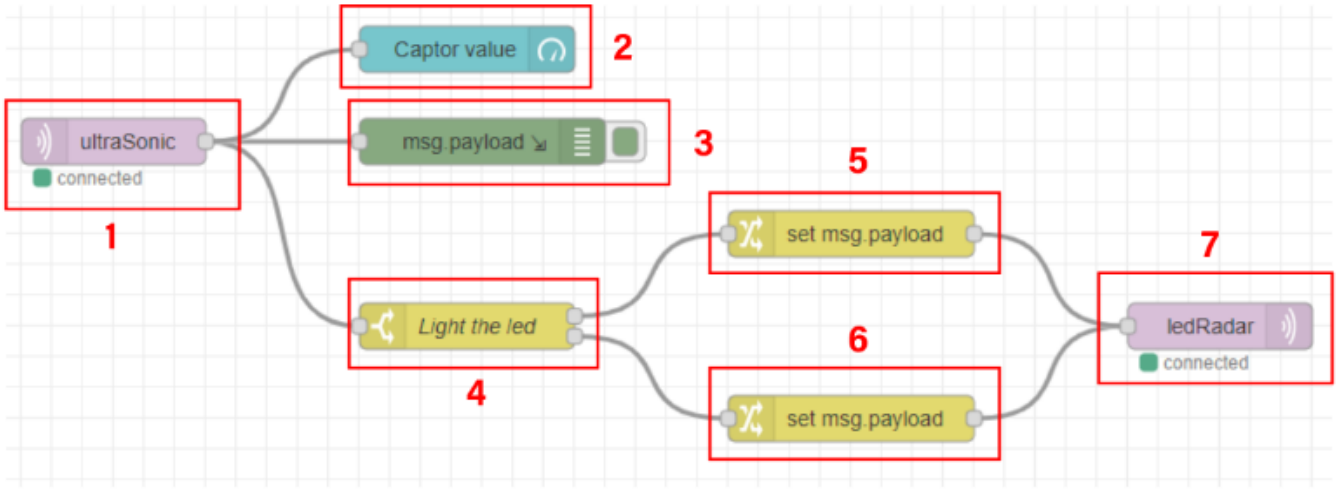
```
void loop() {
int messageSize = mqttClient.parseMessage();
if (messageSize) {
// Un message a été reçu
Serial.print("Received a message with topic '");
Serial.print(mqttClient.messageTopic());
Serial.print("', length ");
Serial.print(messageSize);
Serial.println(" bytes:");
while (mqttClient.available()) {
//récupérer le contenu du message
message =(char)mqttClient.read();
Serial.print(message);
if (message==0)
{
Serial.println("Eteindre la Pin");
digitalWrite(ledPin, LOW);
}
if (message==1)
{
Serial.println("Allumer la Pin");
digitalWrite(ledPin, HIGH);
}
}
Serial.println();
Serial.println();
}
}
```

Sauvegarde annulée.

I. Le flux node red :

Code JSON pour recréer le flux :

[{"id": "7c1633582a8c82b7","type": "ui_gauge","z": "61453f45992eeff4","name": "","group": "e0238fa42e227729","order": 0,"width": 0,"height": 0,"gtype": "gage","title": "Captor value","label": "cm","format": "{{value}}","min": 0,"max": "500","colors": ["#00b500","#00e6cb","#8c00ff"],"seg1": "","seg2": "","className": "","x": 670,"y": 260,"wires": [] }, {"id": "e0238fa42e227729","type": "ui_group","name": "Dashboard","tab": "364090a8dfb1bed5","order": 1,"disp": true,"width": "6","collapse": false,"className": "" }, {"id": "364090a8dfb1bed5","type": "ui_tab","name": "Home","icon": "dashboard","disabled": false,"hidden": false }]



- le node MQTT In : Recevoir les valeurs des capteurs qui sont destinés au topic ultraSoonic
- le node Gauge : Affichage des valeurs reçues
- Switch : Compare les valeurs reçues à un certain seuil prédéfin
- 5. et 6. Change : Allumer la led à travers 5 si valeur < seuil, sinon l'eteindre à travers 6
- le node MQTT Out : Envoie la commande d'allumage de led à la carte subscriber au topic ledRadar