DESIGNING AN ALGORITHM AND MAKING AN INFORMATIC APPLICATION FOR IMPLEMENTING MQTT SPECIFIC ACTIONS

TUREAC Marius[1], SAVU Tom[2]
[1]Faculty of Industrial Engineering and Robotics, Study program: Applied Informatics in Industrial Engineering, Academic year: IV, e-mail: marius.tureac2212@stud.fiir.upb.ro
[2] Faculty of Industrial Engineering and Robotics, Manufacturing Engineering Department, University POLITEHNICA of Bucharest

ABSTRACT: We are going to talk about the MQTT protocol and how it sends data (values measured by sensors in converted to bytes format) from a publisher that is an ESP32 to different topics that are stored on a server then accessed by clients which subscribe to certain topics on the server. The word topic refers to an UTF-8 string that the server uses to filter messages for each connected client. Subscribers can use the topics to control other devices from distance.

KEYWORDS: ESP32, Arduino, Server, Publish, Subscribe

1. Introduction

MQTT is a publish-subscribe network protocol which is designed for remote connections with limited network bandwidth.[1]

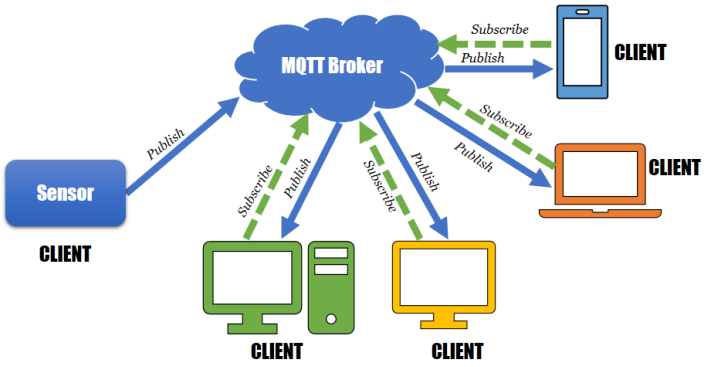

Fig. 1. MQTT diagram Publish-Subscribe [2]

The goal of this project is implementing MQTT protocol specific actions, in order to achieve this, we are going to use several MQTT specific for example:

"Connect" initializes connection with the server.
"Connack" confirms server connection.
"Publish" packet send the data measured from the sensors or data set by the user to the server.
"Puback" packet confirms that the data has been sent to the server.
"Subscribe" packet subscribes to a topic in which first client published data, the second client receiving data.
"Suback" packet confirms that the data sent from the server was received.
"Unsubscribe" packet unsubscribes from the topic that was subscribed and stops receiving data.
"Unsuback" packet confirms that the client unsubscribed.[3]

2. Current state

In this project for data publish to the server we used a ESP32 development board which acts as a client and the software ArduinoIDE to compile the program and uploading it to the board. The ESP32 board having integrated wifi module we connected it to the local wifi for tests then we changed the IP and PORT on which the server runs to publish data.

```
const char* ssid = "******";
const char* password = "********";
const char* mqtt_server = "141.85.192.24";
const char* mqtt_port = "1883"
const char* mqtt_server1 = "192.168.0.94";
void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
```

Fig. 2. Development board configuration used to connect to the internet and server

After configuring the board for internet connection and server connection we configured the temperature, humidity and atmospheric pressure sensors to read the measured data.

```
#define DHTPIN 18
#define DHTTYPE DHT22
#include <Adafruit_BMP280.h>
temperatură=dht.readTemperature();
umid=dht.readHumidity();
atmpres = (bmp280.readPressure()/100);
```

Fig. 3. Define sensor pins connected to the board and read measured data

To publish the measured data to the server we use the client.publish function, this function sends the values in byte format to the server. Every measured value is stored in a specific topic for example, temperature is stored in the topic : „Temperatura”,humidity in the topic „Umiditate”,atmospheric pressure in the topic : „Presiune Atmosferica”.

```
char tempString[8];
dtostrf(temperature, 1, 2, tempString);
client.publish("Temperatura", tempString);
char humString[8];
dtostrf(humid, 1, 2, humString);
client.publish("Umiditate", humString);
char pressString[8];
dtostrf(atmpres, 1, 2, pressString)
client.publish("Presiune Atmosferica", pressString);
```

Fig. 4. Publish measured data to server

For the server and subscriber we use LabView with which we can program through interface.

We open a TCP/IP connection on the „1883” port and wait for a connection between the client (publisher) and server.

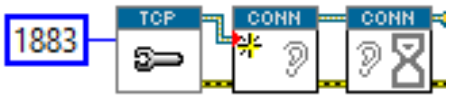

Fig. 5. Opening TCP connection server-client

If there is a connection we start the MQTT server which allows access of data writing in the topics and in a window in the front panel it shows the number of existing connections to the server.

The subscriber connects to the server IP with TCP Connect function and if it returns a valid connection we can subscribe to a topic.

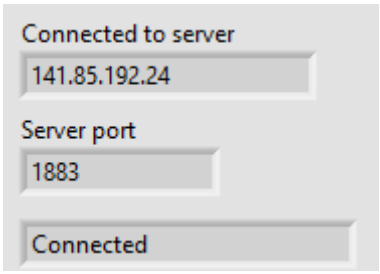

Fig. 6. Subscriber connected to server

To command a device we connected an Arduino UNO board to Labview, selected the communication port to „COM 5” and the bitrate to „9600”.

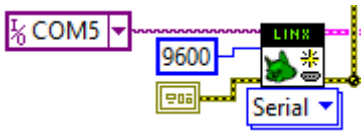

Fig. 7. Labview configuration with the arduino board

After we made the connection between the board and software, we need to login because not all subscribers have access to all the topics , as we see in the login window we have three subscribers „Admin” , „Subscriber 1” and „Subscriber 2”. Admin can subscribe to all topics, Subscriber 1 can subscribe only to Temperature and Humidity and Subscriber 2 can only subscribe to Atmospheric Pressure.

Fig. 8. Login window

Following the authentication process we have access to all topics because we logged in with Admin credentials.

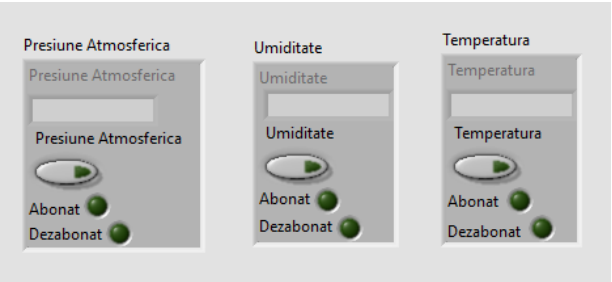

Fig. 9. Subscribe/Unsubscribe interface in LabView

Now, subscribing to a topic , the client (subscriber) receives the data from server (byte value) and converts them to numeric (float) values.

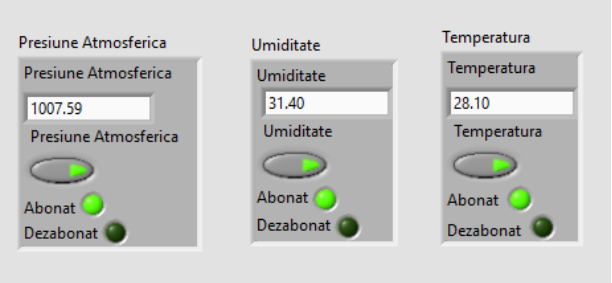

Fig. 10. Subscribed client to topic interface

To control the devices we use for example temperature, this having a limit of 30 degrees Celsius , if its above 30, a fan starts the cooling process.

Fig. 11. Temperature < 30 degrees Celsius LabView

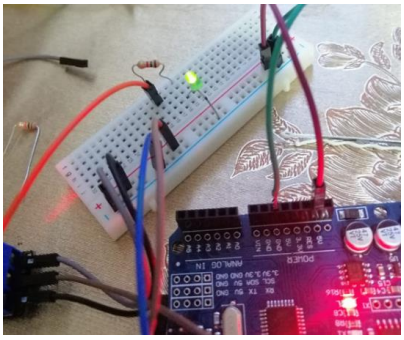

Fig. 12. Temperature < 30 degrees Celsius Arduino LED

Fig. 13. Temperature > 30 degrees Celsius LabView

Fig. 14. Temperature > 30 degrees Celsius Fan

3. Conclusion

Original contributions to this project:

1. Physical connection of sensors on the ESP32 board, fan connection on arduino Uno board.
2. MQTT server connection from LabView of ESP32 board through ArduinoIDE.
3. Making the VI-s.
4. Making the experimental stand.

4. Bibliography

[1].What is MQTT protocol , What is MQTT Protocol and How MQTT works? Applications (microcontrollerslab.com)
[2]. Using MQTT with Mosquitto Getting Started with MQTT using Mosquitto - Embedded Laboratory.
[3].Understanding MQTT protocol packets Understanding the MQTT Protocol Packet Structure (steves-internet-guide.com)