Contents

1 Introduction **2**
- 1.1 What is IoT? . . . 2

2 IoT architecture **2**
- 2.1 Three-tier architecture . . . 2
- 2.2 Five-tier architecture . . . 3

3 Hardware Resources **4**
- 3.1 DHT11 . . . 4
- 3.2 HC-SR501 . . . 5

4 MQTT Protocol **5**
- 4.1 Connection . . . 6
- 4.2 Publish/Subscribe Model . . . 6
- 4.3 Publish . . . 7
- 4.4 Subscribe . . . 7
- 4.5 Why MQTT? . . . 7

5 Adafruit Broker **7**

6 Design **8**
- 6.1 Pseudo-Code . . . 8
- 6.2 Flowchart . . . 8
- 6.3 Implementation . . . 9

7 Temperature Application **10**

8 Discussion **11**
- 8.1 Design Summary . . . 11
- 8.2 Pitfalls . . . 11

9 Conclusion **13**

References **14**

1 Introduction

Internet of things (IoT) has been developed from the convergence of a series of technologies, which have emerged independently since its beginning. In the internet of things (objects) have an internet connection at any time and place. In a technical sense, it consists of integrating sensors and devices into everyday, objects that remain connected to the internet through fixed and wireless networks [Patel et al., 2016]. The fact that the internet is present at the same time everywhere makes the massive adoption of this technology more feasible. Given their size and cost (IoT), the sensors are easily integrable in homes, work environments, and public places. In this way, an object can connect and manifest on the network. Furthermore, IoT implies that every object can be a data source. That is the beginning to modify the way of doing business, the organization of the public sector and the daily life of millions of people. This report shows the current view of the technologies used in IoT, application models and their impact on our daily life.

1.1 What is IoT?

Internet of things (IoT) encompasses a series of devices and technologies that, together with the Internet, ubiquitous in all corners of the world, are affecting our daily lives in unimaginable ways [Patel et al., 2016]. Today humanity lives in an era in which the great variety of devices are connected through the Internet and therefore with each other

2 IoT architecture

Although different architectures have been proposed for IoT, there is no agreement, different investigations have proposed architecture models that collect in a kind of the different aspects of IoT. Some of the most relevant architectures are shown below.

2.1 Three-tier architecture

One of the most basic architectures is the three-tier architecture: the sensing layer (EPC sensor network), the network layer, and the application layer (information service system) [Zhang and Zhu, 2011], architecture shown in figure 1.

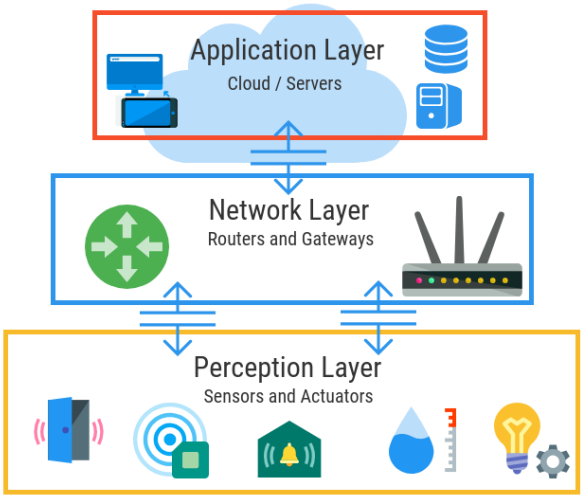

Figure 1: Three-tier Architecture

· The level of perception is the physical level, where the sensors collect information from the environment.

· The network level is responsible for connecting the sensors and servers together to transmit and process the data collected by the sensors.

· The application level is where IoT can be deployed in different application areas.

2.2 Five-tier architecture

It is formed by levels of perception, transportation, process, application and level of business. In this architecture model, the levels of perception and application are the same as in the three-level architecture [Wu et al., 2010]. Levels on the figure 2 and explanation:

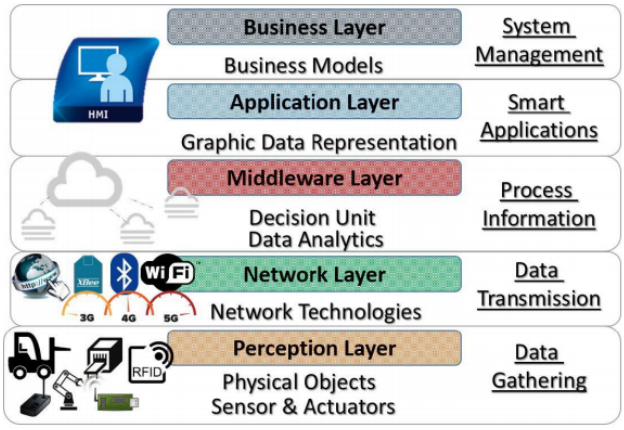

Figure 2: Five-tier Architecture

· The transport level transfers the data from the sensors through 4G, LAN, Bluetooth, RFID and NFC networks, from the perception level to the process level and vice versa.

· The process level stores, analyzes and processes large amounts of data from the perception level. It can provide and manage services at the lowest levels using a database, cloud computing and big data technologies.

· The business-level manages the applications, the business model and privacy.

3 Hardware Resources

Sensors and actuators are fundamental pieces for IoT. They enable objects to interact with each other and humans through the internet, gathering information from the environment or interacting with it. These devices are increasingly being smaller, helping their integration in any area

3.1 DHT11

Through this DHT11 sensor, It gets a very accurate temperature and humidity measurement wherever It is, with fast response and no interference. It is a resistant, simple and very practical sensor with a very economical price. The DHT11 sensor has three pins, one pin for the voltage input, another pin for the ground, and another pin for data transfer, shown in figure 3 .

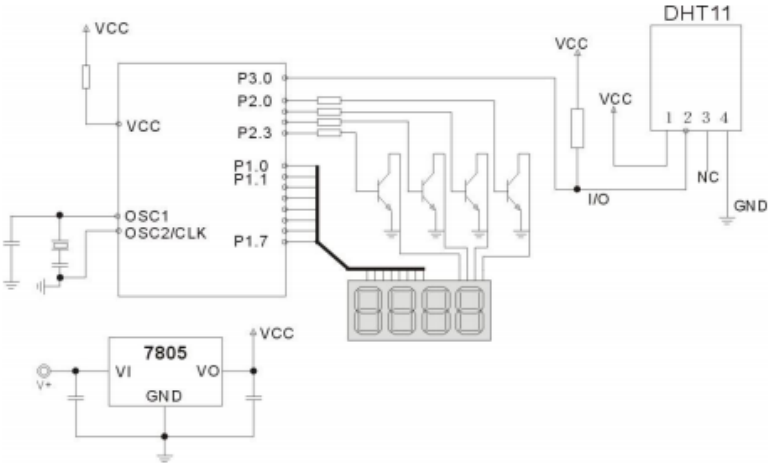

Figure 3: DHT11 Circuit

3.2 HC-SR501

A passive infrared sensor is a sensor electronic that measures the infrared (IR) light radiated by objects (alive or not) located in your field of vision. These sensors are mainly used as motion detectors.

These devices work thanks to infrared radiation that is imperceptible for the human eye, therefore, objects with a temperature above absolute zero emit heat that can be detected by these devices that have been designed for this purpose, HC-SR501 shown in figure 4.

Figure 4: HC-SR501

4 MQTT Protocol

MQTT is a publish/subscribe, shown if figure 5, message transport protocol between client and server. Is light, open, simple and designed to be easy to implement. These characteristics make it an ideal choice in many situations, including restricted environments such as communication in Machine to Machine (M2M) and Internet of Things (IoT) contexts, where a small amount of code footprint and/or network bandwidth is very important [Hunkeler et al., 2008].

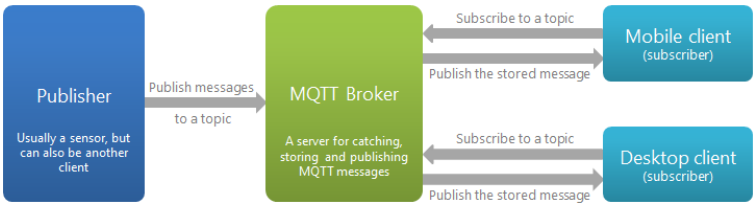

Figure 5: Publication/Subscription

An MQTT client is any device that runs an MQTT library and is capable of connecting to an MQTT broker over the internet. Therefore, any device that communicates through MQTT and through the TCP / IP protocol stack, it can be called an MQTT client. Both publishers and subscribers are MQTT clients. In turn, the same client is capable of publishing and receiving messages [Hunkeler et al., 2008].

The previously called broker is the complementary element to the MQTT client, verified as the main element of any publication/subscription protocol. It is responsible for receiving

all messages, filter them, determine who is subscribed to each message and send the messages to those clients subscribed

In order to handle messages correctly, the broker has several filtering options: topic based, type-based and content-based. The message filter used by MQTT is based on on a topic. This type of filtering is based on a topic that is part of each message. The client subscribe to the broker to be aware of the topics of interest. From that moment, the broker ensures that the client gets all the messages that have been published in those topics [Hunkeler et al., 2008].

4.1 Connection

The broker always handles client connections. To initiate a connection, the client sends a CONNECT message, which answers with CONNACK and a status code, returning 0 in case the connection has been accepted [Yokotani and Sasaki, 2016], it is shown in figure 6.

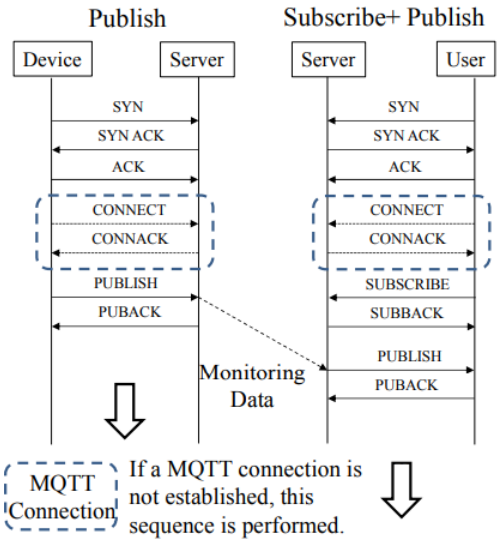

Figure 6: MQTT Connection

4.2 Publish/Subscribe Model

The publish/subscribe model provides an alternative to the traditional client/server architecture. In the client/server model, a client communicates directly with the endpoint. The pub/sub model separates the client sending the message (publisher) from the client or clients receiving the message (subscribers). Publishers and subscribers never communicate with each other directly. In fact, they may not be aware of each other's existence. A third component, the broker handle the connection between them. The broker's job is to filter all received messages and correctly distribute them to subscribers [Yokotani and Sasaki, 2016].

4.3 Publish

Each message presents a payload that contains the data to be transmitted in the format that the client indicate. The content of this message carries with: topic, QoS, payload. When a client sends a message to the broker to be published, it reads the message, agrees, and processes, so that it determines which clients are subscribed to the topic and sends them the message [Hunkeler et al., 2008].

4.4 Subscribe

Clients tell the broker what topic they are interested. When a client subscribes to a topic, any message posted to the broker is distributed to the subscribers of that topic. Clients can also unsubscribe to stop receiving broker messages on that topic [Hunkeler et al., 2008].

4.5 Why MQTT?

Having seen the protocols taught in lectures (CoAP, HTTP), I have realized that this protocol seems to have a series of advantages to be easily implemented with basic knowledge and straightforward, Additionally, with this protocol, it is not necessary to build an HTTP server which will need to have a domain on the internet to have access to it and also open the MQTT port (1883 or 8883) in the router and build an HTML page. Another of the advantages that I have valued are these:

· Efficient data transmission and quick to implement due to its light protocol.

· Low use of the network, due to the minimization of data packets.

· Efficient distribution of data;

· Successful implementation of remote sensing and monitoring.

· fast and efficient message delivery.

· Use of small amounts of energy, which is good for connected devices and reduction of network bandwidth.

5 Adafruit Broker

For the recreation of the applications, the platform www.io.adafruit.com will be used. Adafruit is a globally distributed broker, made up of Mosquitto servers in the cloud. Adafruit implements the MQTT protocol which, as it has been seen previously, provides light methods for sending messages using the publication/subscription queue model.

Message queues provide an asynchronous communication protocol, through it the sender and the recipient of the message does not need to interact with the message queue at the same time. The messages that are entered in the queue are kept until the receiver claims it or until the message expires.

The Adafruit broker uses a port for connection and also offers a secure connection. For unsecured connections, the port number begins with 1, for TLS / SSL at 8.

The MQTT protocol has a username and password field in the CONNECT message for the authentication. In this way, to be able to obtain communication with the broker, it is simply necessary to know the server name or IP address, a user, the password and the Adafruit key.

First, to start using Adafruit, it is necessary to register, then take the Adafruit key with the user name, moreover to creating the different feeds, dashboards.

6 Design

6.1 Pseudo-Code

The temperature application runs with this pseudocode, the code itself of the application is much more, but this in itself shows the main duty.

Algorithm 1 Show Temperature

```
if data from toggle is different from nil then
  if data from toggle is equal to ON then
    creation of the timer of 3 seconds
    read temperature from the sensor
    publish the temperature
  end if
  if data is equal to OFF then
    turn off the timer
  end if
end if
```

To turn off the timer tobj:unregister() is used [NodeMCU-Documentation, nd].

6.2 Flowchart

To understand all the application code is shown in the diagram figure 7. The application connects to the broker using the MQTT protocol on port 1883 through username and password.

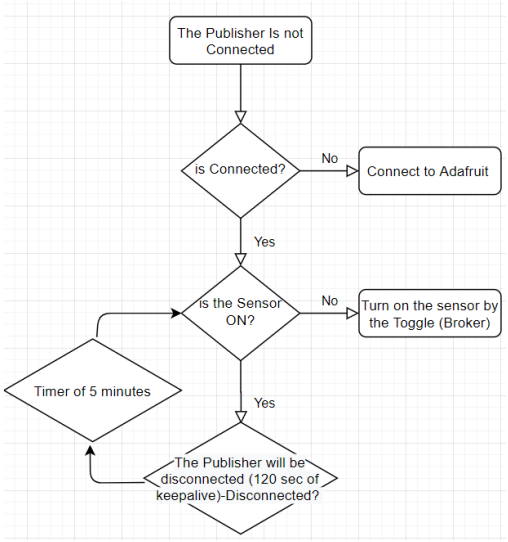

Figure 7: Flowchart

Once connected, the session will last about 120 seconds [NodeMCU-Documentation, nd], during this time, the node will send keepalive packets to keep the session alive.
When the connection is established, the client can turn the sensor on or off through the broker, when the session has ended it will create an alarm of 5 minutes (300000 milliseconds on the code) which will make the user enter to the main code again to turn on or turn off the sensor.
Without this timer, it would make the session last only 120 seconds, and the node of the broker will be disconnected

6.3 Implementation

For the implementation there is a pin that is connected to the GPIO2, in this case the blue wire and the other wires are one for electricity 8.

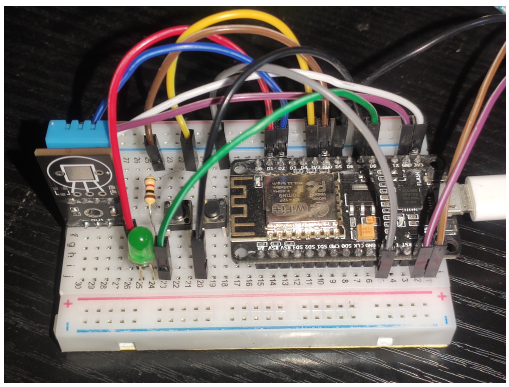

Figure 8: DHT11 in Breadboard

7 Temperature Application

This application aims to show the current temperature with an update margin every three seconds. The user can turn on or off the sensor remotely through an MQTT broker.
When any user subscribes to this topic, they will be able to know the temperature where the sensor is located, through the broker who will publish the data as shown in the figure 9.

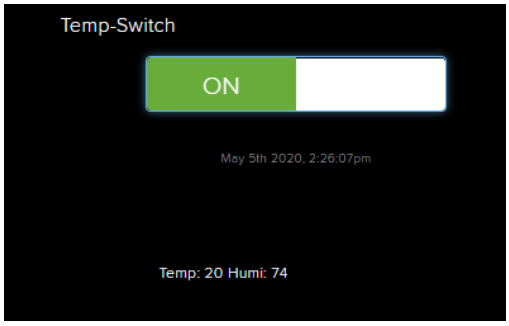

Figure 9: Output Through The Broker

To make it more flexible and comfortable for the end user, it can also be connected through the smartphone using an MQTT client, as shown in the following figure 10.

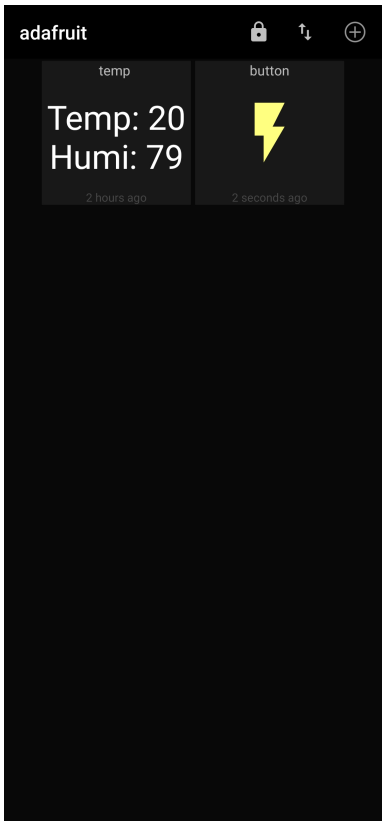

Figure 10: MQTT Client From a Smartphone

8 Discussion

8.1 Design Summary

This application is basically to control a temperature sensor which is remotely connected. It can be accessed by PC, smartphone.

8.2 Pitfalls

This application could be developed much more. However, the most significant disadvantage found when building it has been that the broker (Adafruit) sends the same type of data on its buttons, that is, if someone puts a button like on/off this will always send the same data on or off, which can be modified, although it does not save changes, as shown in the following figure 11.

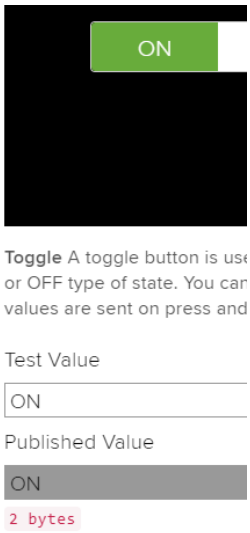

Figure 11: Toggle Values Modification

On the other hand, the clients used through smartphone do allow to change this type of value, as shown in the following figure 12.

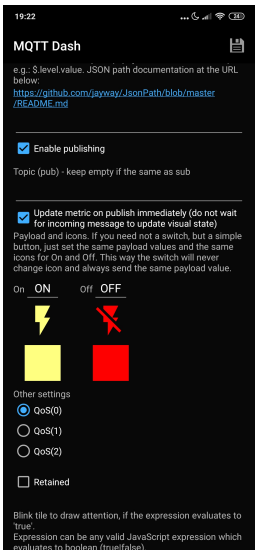

Figure 12: MQTT APP Values Modification

This value is fundamental since in the code, there is a condition that allows advancing to the next step like this in this figure 13.

```
if data ~= nil  then
   if data =='ON' then
    print(data)
```

Figure 13: Code of Data

A solution would be either to use a premium broker which helps to develop the code much more or to implement an HTTP server and pay for an internet domain to be able to access the application.

9 Conclusion

IoT is a very current field of work with great potential. Before this project I was unlucky to do any kind of work related to the Internet of Things, which has led me to the enquiry process and knowledge of this architecture. Its capabilities are practically infinite as well such as the number of technologies in which it can be implemented. In this case, it has been decided to leave aside with the traditional HTTP protocol technology and proceed with unknown technology.

The MQTT protocol was totally strange for me, and this has made me a pleasant surprise thanks to its simplicity and management both from the client and the server. The adafruit MQTT platform has been an extraordinary discovery due to its great comfort. It presents when creating a cloud server. In less than five minutes, it is possible to have an active broker waiting for connections.

References

[Hunkeler et al., 2008] Hunkeler, U., Truong, H. L., and Stanford-Clark, A. (2008). Mqtt-s—a publish/subscribe protocol for wireless sensor networks. In *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08)*, pages 791–798. IEEE. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4554519.

[NodeMCU-Documentation, nd] NodeMCU-Documentation (n.d.). https://nodemcu.readthedocs.io.

[Patel et al., 2016] Patel, K. K., Patel, S. M., et al. (2016). Internet of things-iot: definition, characteristics, architecture, enabling technologies, application & future challenges. *International journal of engineering science and computing*, 6(5). http://ostadr.ir/trans/iot/i4.pdf.

[Wu et al., 2010] Wu, M., Lu, T.-J., Ling, F.-Y., Sun, J., and Du, H.-Y. (2010). Research on the architecture of internet of things. In *2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, volume 5, pages V5–484. IEEE. https://ieeexplore.ieee.org/abstract/document/5579493.

[Yokotani and Sasaki, 2016] Yokotani, T. and Sasaki, Y. (2016). Comparison with http and mqtt on required network resources for iot. In *2016 international conference on control, electronics, renewable energy and communications (ICCEREC)*, pages 1–6. IEEE. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7814989.

[Zhang and Zhu, 2011] Zhang, H. and Zhu, L. (2011). Internet of things: Key technology, architecture and challenging problems. In *2011 IEEE International Conference on Computer Science and Automation Engineering*, volume 4, pages 507–512. IEEE. https://ieeexplore.ieee.org/abstract/document/5952899.