Final Degree Project

Degree in Industrial Engineering

Comparison of different Internet of Things platforms

Report

Author: Daniel Torán Mercadé
Director: Manuel Moreno Eguílaz
Call: January 2018

ETSEIB

Resum

Aquest projecte s'emmarca en el concepte de la IoT (la Internet de les coses) i té com a objectiu dur a terme el desplegament d'una arquitectura IoT utilitzant un dispositiu amb connectivitat Sigfox i diferents plataformes IoT com AWS IBM Watson IoT, Microsoft Azure i thethings.iO.

El dispositiu Sigfox està transmetent mesures de temperatura i aquestes s'envien a les plataformes IoT, que han estat configurades per rebre i emmagatzemar les dades. Un cop les dades s'han emmagatzemat a les plataformes, l'objectiu principal és poder analitzar-les i treure'n informació valuosa. Per tant, s'han dut a terme visualitzacions gràfiques.

Després de desplegar les diferents arquitectures IoT, les plataformes han estat testejades per verificar les diferències entre elles pel que fa al processament de dades i a la visualització. S'han comparat considerant característiques com facilitat de connectar els dispositius i analitzar les dades, preu, flexibilitat o serveis.

Aquesta memòria explicarà com configurar aquestes plataformes per poder rebre i analitzar les dades, utilitzant l'exemple del sensor de temperatura, i quina d'elles és més adequada per resoldre una necessitat concreta d'algun camp del coneixement.

Resumen

Este proyecto se enmarca en el concepto del IoT (el Internet de las cosas) y tiene como objetivo llevar a cabo el despliegue de una arquitectura IoT utilizando un dispositivo con conectividad Sigfox y diferentes plataformas IoT como AWS, IBM Watson IoT, Microsoft Azure y thethings.io.

El dispositivo Sigfox está transmitiendo medidas de temperatura y éstas se envían a las plataformas IoT, que han sido configuradas para recibir y almacenar los datos. Una vez los datos se han almacenado en las plataformas, el objetivo principal es poder analizarlos y sacar información valiosa. Por lo tanto, se han llevado a cabo visualizaciones gráficas.

Después de desplegar las diferentes arquitecturas IoT, las plataformas han sido testeadas para verificar las diferencias entre ellas en cuanto al procesamiento de datos y a la visualización. Se han comparado considerando características como facilidad de conectar los dispositivos y analizar los datos, precio, flexibilidad o servicios.

Esta memoria explicará cómo configurar estas plataformas para poder recibir y analizar los datos, utilizando el ejemplo del sensor de temperatura, y cuál de ellas es más adecuada para resolver una necesidad concreta de algún campo del conocimiento.

Abstract

This project falls into the concept of IoT and aims to carry out the deployment an IoT architecture using a Sigfox device and different IoT platforms such as AWS, IBM Watson IoT, Microsoft Azure and thethings.iO.

The Sigfox device is sending periodically temperature readings and these are sent to the different IoT platforms, which have been configured to receive and store the data. Once the data is stored in the platforms, the main objective is to get useful insight from them. Therefore, graphical visualizations have been carried out.

After all the IoT architectures have been deployed, the different IoT platforms have been tested to verify the differences between each other when it comes to data processing and visualization. A comparison of the platforms considering features, such as easiness to connect and analyse, pricing, flexibility or services, has been accomplished

This report will explain how to configure these platforms in order to receive and analyse data, using the example of the temperature sensor, and which one is more suitable for a specific purpose or needs of a particular area of knowledge.

Table of contents

RESUM ______ **1**

RESUMEN ______ **2**

ABSTRACT ______ **3**

TABLE OF CONTENTS ______ **5**

1. **GLOSSARY** ______ **7**

2. **PREFACE** ______ **9**

2.1. Project Genesis 10
2.2. Motivation 11
2.3. Prerequisites 11

3. INTRODUCTION ______ **12**

3.1. Project Objectives 12
3.2. Long Term Goal 12

4. STATE OF THE ART ______ **14**

4.1. Development of the IoT concept 14
4.2. LPWAN 18
4.3. Sigfox 20
4.4. TD1208 EVB 23

5. CONFIGURATION OF THE TD 1208 EVB ______ **25**

5.1. How the TD1208 module works 28
5.2. Sending a message Command 29
5.3. Temperature Command 30
5.4. Python Programme that gets the temperature and sends it to the Sigfox Cloud. 30

6. IOT ARCHITECTURE ______ **33**

6.1. The 4 stage IoT solutions architecture 34
6.1.1. Stage 1: IoT peripheral nodes 34
6.1.2. Stage 2: Internet Gateways and Data Acquisition Systems 35
6.1.3. Stage 3: Edge IT 35
6.1.4. Stage 4: Data Centre/ Cloud 36

6.2. IoT architecture solutions 37
6.2.1. AWS Architecture Solution 38
6.2.2. Azure Microsoft Architecture Solution...39
6.2.3. IBM Watson IoT Architecture Solution...40
6.2.4. thethings.iO Architecture Solution 42

7. IOT CLOUD PLATFORMS ____________________________________ 43

7.1. Configuration of the IoT Platforms.. 44
7.1.1. AWS Configuration 44
7.1.2. Azure Microsoft Configuration 49
7.1.3. IBM Watson IoT Platform Configuration..51
7.1.4. thethings.iO Configuration 52
7.1.5. Next configurations 54
7.2. Comparison of the IoT platforms... 56
7.2.1. AWS 57
7.2.2. Azure Microsoft...57
7.2.3. IBM Watson IoT platform..57
7.2.4. thethings.io 57
7.2.5. Comparison table of different IoT platforms 58

8. BUDGET ____ 59

9. ENVIRONMENTAL IMPACT _______________________________ 60

10. CONCLUSIONS _______________________________________ 61

ACKNOWLEDGMENTS ______________________________________ 63

BIBLIOGRAPHY ____ 64

1. Glossary

Abbreviations

DDB	Dynamo Data Base
ESD	Electrostatic Discharge
FTDI	Future Technology Devices International
GND	Ground
GUI	Graphical User Interface
HTTPS	Hypertext Transfer Protocol Secure
I/O	Input/Output
IT	Information Technology
MCU	Microcontroller Unit
MQTT	Message Queuing Telemetry Transport
NIST	National Institute of Standards and Technology
NoSQL	No Structured Query Language
SMA	SubMiniature version A
VPC	Virtual Port Channel
WIoTP	Watson Internet of Things Platform

Definition of basic concepts

API: Application Programming Interface is a set of definitions, protocols and tools for enabling the communication between various software components.

DAS: Direct-attached storage (DAS) is digital storage directly attached to the computer accessing it, as opposed to storage accessed over a computer network.

IP address: Internet Protocol address is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication.

ISM: The Industrial, Scientific and Medical radio bands are radio bands reserved for the use of radio frequency for the purposes that its name indicates.

JSON: JavaScript Object Notation is an open-standard file format based on a design that foster human-readable text to transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value).

LPWAN: Low-Power Wide-Area Network is a kind of telecommunication network devised for having a wide range and a low bit rate. It is suitable for things like sensors powered by batteries, which with this technology can last for years.

M2M: Machine to Machine refers to the communication between two machines using any communication channel. Recently the term has evolved into a system of networks that convey data to personal appliances.

Parser: A parser is a software component that takes input data (frequently text) and builds a data structure, so as to give a structural representation to the input.

SDK: A software development kit (SDK or devkit) is typically a set of software development tools that allows the creation of applications for a certain software package, software framework, hardware platform, computer system, video game console, operating system, or similar development platform.

Sigfox**:** Sigfox is a company founded in 2009 whose aim is to connect low-energy objects, the ones used in IoT architecture. It uses an ISM radio band of 868 MHz in Europe and UNB, which, used jointly, is called the LPWAN technology.

SSD: Solid-state drive is a solid-state storage device that uses integrated circuit assemblies as memory to store data persistently.

UNB**:** Ultra Narrow Band is a technology that transmits over very narrow spectrum channel, less than 1 kHz. This narrow spectrum allows to reach ultra-long distance (5 km in urban and 25 km in rural areas)

2. Preface

Another master and degree students at ETSEIB have worked on the Sigfox technology and have used the evaluation TD1204 board from Telecom Designs developed for Sigfox.

The essence of each project will be explained and how this project links to what previously has been worked on in Department of Electronic Engineering at ETSEIB-UPC.

The use and experimentation of Sigfox Technology using a TD1204 module in academic projects at ETSEIB-UPC began in 2014. At first, as it was a little-known technology the bulk of the information was found in very specialized forums and blogs. Each project has made a step in using all the potentialities of the TD1204 board and creating different usabilities and platforms for its use. However, the motive of using this technology has been its importance in the development of the IoT revolution. Therefore, every project has the aim of assessing how the TD1204 board can be implemented in an IoT project and give the keys of how each project could be developed.

Following, there is a brief summary of the content and the aims of each previous project:

- Familiarize with all the functions that TD1204 board has and then apply this expertise to create an IoT application related to mobility. Using the options included in the TD1204 evaluation board such as the geolocator and the accelerometer to create an app that was installed in an electric motorbike developed by the university. This application is able to assess the level of charge on the batteries and whether the vehicle is on or off. This allows to have real-time information and therefore the motorbike team will be able to make decisions more well-grounded. [1]

- The aim of this project was, using the options included in the TD1204 evaluation board and a PIC18F4520 microcontroller, to create an application able to inform about the condition of the bicycles of a public company hiring. Using an accelerometer, the application was able of considering if the biker has suffered an accident. If a sudden movement change occurs and the biker does not press a button confirming he is okay, a message will be delivered to emergency services. Moreover, the bike has a burglar alarm and a location service. Another purpose is to assess the benefits of using IoT approach to solve problems of mobility regarding the use of public services and eventually being able of reducing its price. [2]

- Building on the knowledge of previous projects, the aim of this project was to receive real-time information about a motorbike taking part in the Smart Moto Challenge and then show this information in a very clear way using a website created with this purpose. The information shown in a Control Panel is about battery level and location. As well as exploring all the options of the TD1204 it has been crucial to learn how to use the API technology in order to convey the information from the Sigfox console to the website. [3]

- This project aims to create a GUI (Graphic User Interface) in order to interact with the TD1204 evaluation board and get to its functions from a graphics environment, with didactic and academic purposes. The GUI, that is programmed in Python using the Tkinter library, allow people who do not have previous knowledge of how the TD1204 board works to use it in a clear way since it also permits to see, by means of a terminal, the interactions between the GUI and the board. [4]

- The objective of this project was to create a library of functions and procedures for getting IoT data using Microsoft Excel. This library is used to treat data generated by IoT devices. To accomplish this goal, it is necessary to bridge the gap existing between the data generated and the spreadsheet program of choice that could easily handle this data, in this case, Microsoft Excel. This missing link comes in the form of a library written in "Visual Basic of Applications" (VBA), the language that Microsoft Excel uses for user-defined operations. [5]

2.1. Project Genesis

The origin of this project stems from the realization that there was a lack of assessment of which IoT analysis platforms are more suitable regarding the user necessities. That is to say, a clear guide of how to choose an IoT platform when having to start a project.

Therefore, it is necessary to remark the differences between each other according to criteria such as price, effectiveness, simplicity, etc.

The platforms to study have been chosen following many criteria but the most important was that they were able to connect to a Sigfox Device.

2.2. Motivation

At the beginning of this project, when I started to grasp the infinite world of the IoT field, one of the main interests was to understand how we can benefit from receiving data from every-objects and the way in which this is managed.

Not only I wanted to understand why this flow is important but also how it is done, in order to be able to create myself an IoT application and make a fair comparison of what IoT Platforms are more useful for each type of IoT project, with its pros and cons.

Last but not least, gaining experience of how to obtain the data that then is used to create applications, in other words, to get in touch with the emerging field of the Big Data.

2.3. Prerequisites

In this project, state-of-the-art technology and concepts have been tackled, which means that they are continually in evolution. In order to have a major understanding of the procedures and concepts described in this report, it is compulsory that the reader has some knowledge in the field of IoT development, devices and platforms.

Furthermore, in case the reader would want to follow and build the IoT infrastructure described in this report, he will need to fulfil the following requirements:

- Having access to the TD1208 EVB, or an equivalent Sigfox device, and the documentation that thoroughly explains how to configure the device.
- Making a subscription to some IoT platforms such as AWS, IBM Watson, Azure Microsoft, Redash or thethings.iO. In all cases, it can be free for a few months.

Nonetheless, due to the novelty of this field, had the reader possessed a lack of knowledge on how to use an IoT platform or a Sigfox device but lot of interest and drive to understand the main lines of this project, he would find feasible to comprehend how the device and the platforms have been configured so as to achieve insight using an IoT platform from the data sent by the Sigfox device.

A lot of the information of how to configure and create IoT architectures is explained in the pages and blogs of the IoT software and hardware providers aforementioned in case the reader wanted to broaden the knowledge provided in this project.

3. Introduction

3.1. Project Objectives

The aim of this report is to explain how different IoT Platforms work and the way in which they can be connected to IoT devices. The IoT platforms which are studied during the creation of an IoT architecture are: AWS, IBM Watson, Azure Microsoft and thethings.iO.

For testing the different IoT Platforms services and connectivity, a Sigfox device— TD 1208 EVB — is used to convey messages to the IoT platforms and test its connectivity to this kind of IoT technology.

While comparing the different IoT Platforms, the main objectives are assessing the difficulties to connect the Sigfox device, finding a way to store, in the long term, all the data conveyed by the Sigfox device and use this data to get useful insight by doing graphic visualizations.

Another purpose of the project is to configure the TD 1208 EVB in a way that it is able to send real-time temperature readings automatically from a sensor located in the Sigfox chip, thus, providing real data for the testing of the IoT Platforms and creating a temperature service that can be used in the future for improving building insulation or the heating/cooling systems.

The underlying objective throughout this project is to give the reader a broad vision of the importance of the IoT revolution and which the future trends of its development are. Besides, this objective is fulfilled by using a real life example such as monitoring the temperature.

3.2. Long Term Goal

The final goal of this project is to provide a tool for users of IoT devices that want to develop an IoT architecture solution using Sigfox technology, besides, providing a real-case service that can be developed and used in many engineering problems.

The value added provided in this project is the comparison of the IoT Platforms so as the user would be able to choose one platform based on what his needs and intentions of developing an IoT project are.

Moreover, the user would be able to start his IoT project building on an existing architecture, thus, making the deployment project much faster than if he would have to start without a previous architecture. Furthermore, it gives to any user time to think only on what the service he wants to provide might be and then he can make it happen using this architecture.

4. State of the art

4.1. Development of the IoT concept

The Internet of Things is a concept that refers to the connection of ordinary objects— such as cars, house appliances, monitoring sensors, surveillance cameras, and so on— to the network, which allows them to exchange data. When a lot of data is collected and thoroughly analysed, it can be able to solve problems of many different fields.

Hence, by means of the development of better telecommunication technologies, the IoT will foster the development of innumerable applications that transform the infinite raw data provided by IoT devices into objective information and facts that allow citizens, companies and governments to make more well-grounded decisions.

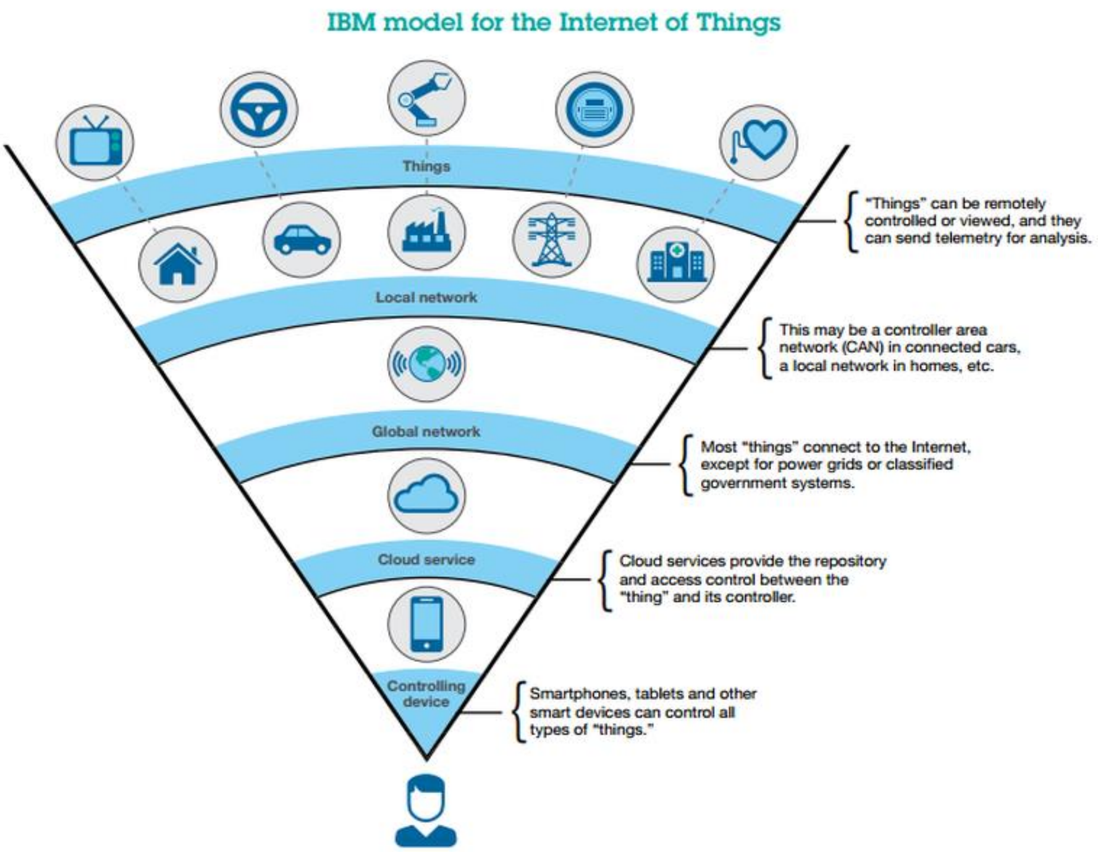

Figure 1: IBM model for the Internet of Things. Source: [6]

The heart of this technology is that it allows making instant decisions and since the data is conveyed regularly, the taking of decisions is easily automatized using machine learning. This kind of data flow between devices is known as machine to machine communications (M2M). With machine learning, algorithms are capable of learning from the past data and then they can make automatically decisions by detecting patterns on the new data. Then, the algorithms can change the software parameters to adapt to these patterns. [7]

One of the fields that are leading the implementation of IoT devices and its use to make decisions is the so-called Smart City concept. It consists on the improvement of ICT technologies to provide better public services to the citizens. [8]

It has been used in fields so unalike as transport and parking, lighting, surveillance and maintenance of public areas, preservation of cultural heritage, garbage collection, salubrity of hospitals and school. [8]

The Smart City concept will lead to a revolution in the way people have lived in cities in the past centuries. By collecting huge amounts of data of how the city works, the governments will be able to make wiser decisions involving the adoption of greener policies. Another profit will be an improvement of the quality of life since every-day problems, as the ones mentioned above, will be softened.

Another positive aspect of the increase in the use of IoT devices in the city has been the democratization of data. The Open-Data concept, which has the same spirit than open source and open hardware movements, stimulates the general public to play an active role in the progress of public services.

Moreover, it fosters the creation— by start-ups and entrepreneurs, which promote a very dynamic economic environment in the city— of applications that provide completely new services just using the data collected by the IoT devices.

The early, quick development of IoT technology on cities and public services will be useful for the introduction of IoT devices to a wider range of fields. [8]

Furthermore, obtaining IoT data not requires a huge investment as the more devices you have connected, the better data you get. The price of these devices depends on the quality and number of sensors and the kind of message you want to send but usually is nearby 10 euros. [8]

It has been broadcasted that by 2020 there will be around 30 billion of IoT connected devices. This refers to any object that has a sensor and is conveying data to the network.

As it can be seen in Fig. 2, in 2014, the number of IoT devices was around 5 billion, a similar number to the Earth population. However, in only five years this number will increase to 30 billion IoT devices, which means 4 devices per person when 5 years before there was only one device per person. This increase will be possible because of the reduction of the sensor cost, which is declining year after year. [9]

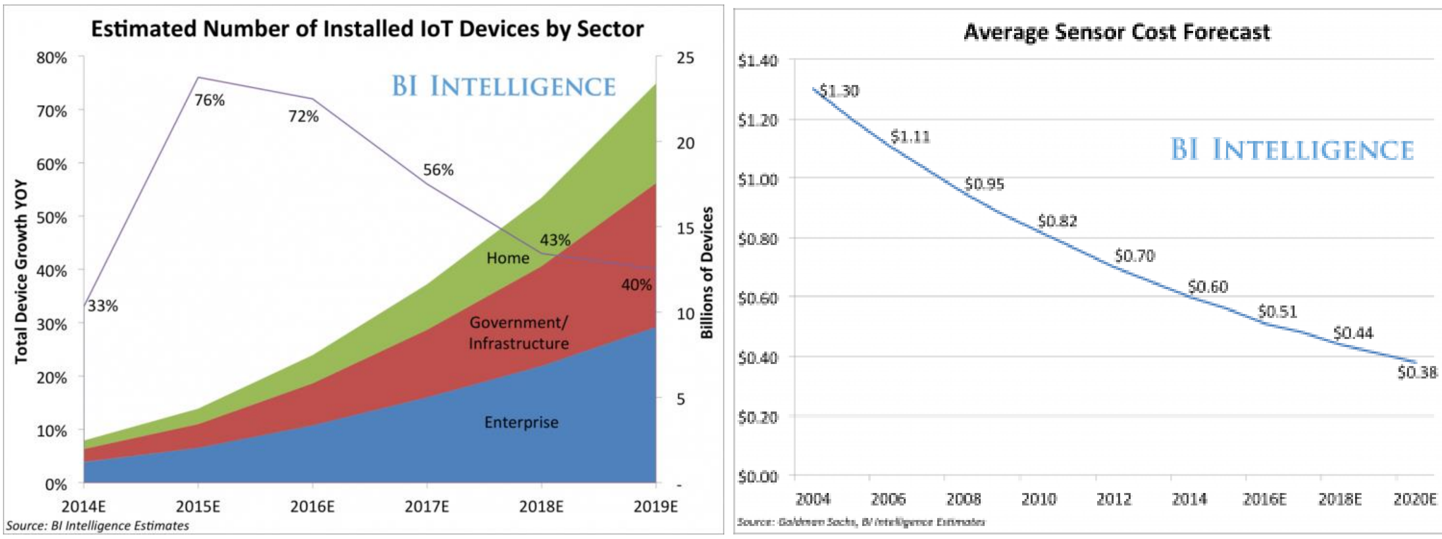

Figure 2: Today's trends in the IoT field. Source: [9]

The fields in which the IoT technology can be used are almost infinite. Applying smart connected devices, sensors, and gateways to control each part of the production process is something a lot of companies are paying heed to its importance to increase operational efficiency.

As it can be seen in Fig. 3, the IoT revolution is having an impact on sectors like Connected Industry, Smart City, Smart Energy and the Connected Car. America has a quantitative lead regarding the bulk of IoT segments except for the Smart City one in which Europe is the main actor. [10]

The connected industry sector will play a major role in the Industry 4.0, which will reshape manufacturing processes. IoT will empower industrial automation projects allowing manufacturing production facilities and products themselves to be mixed with computing hardware. This makes much easier to monitor the manufacturing process since different parts of the production line have communication between each other.

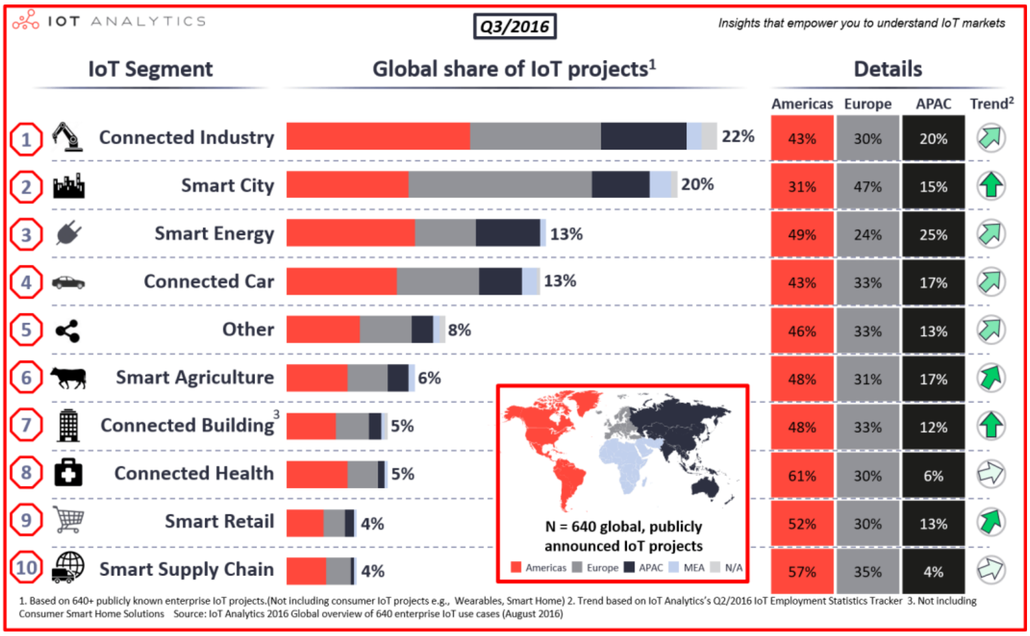

Figure 3: IoT industry segments. Source:[10]

The Industry 4.0 development is grounded on the massive amount of data that IoT devices provide. The idea is that this data will be processed in a Cloud Computing Platform to get useful insight of the production process, and thus creating what is called a smart factory. Using IoT, cyber-physical systems can communicate and cooperate with each other and with humans in real time. Since the production process will reach a high level of automation, the companies will be able to provide mass customization of manufacturing products. [11]

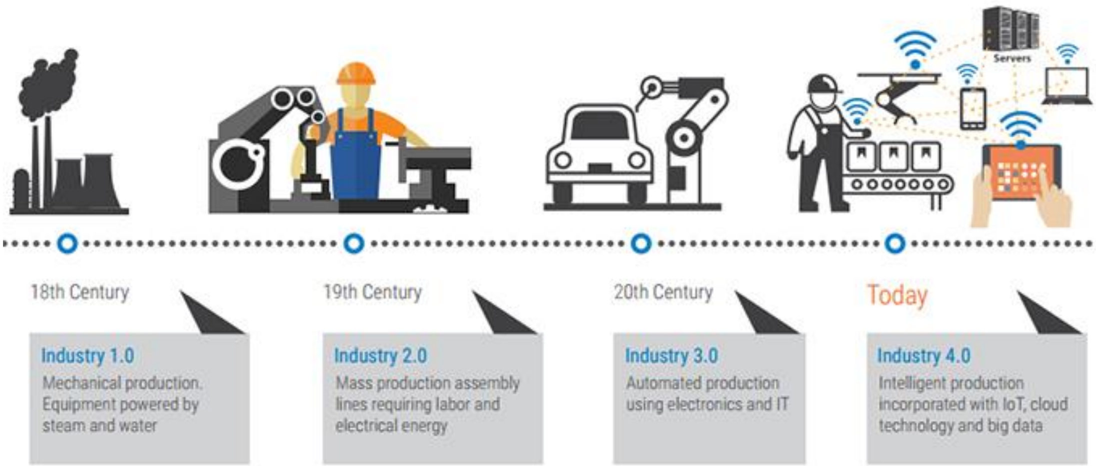

Figure 4: Industrial Revolutions and its implications. Source: [12]

To sum up, the Internet of Things is a broad term that involves M2M communications between devices and the applications developed in many fields with the purpose of using data to optimize processes and increase efficiency in any way possible.

4.2. LPWAN

LPWAN is a type of wireless communication wide area network designed to allow long-range communications at a low bit rate among connected objects, such as sensors operated on a battery.

It establishes communication among things or objects over long range at low bit rate specifically, where traditional wireless network such as Bluetooth, BLE, WiFi, or ZigBee does not fit and M2M connection requires high power and involves high costs.

As it is clearly shown Fig. 5, LPWAN technology has a high energy efficiency, a long range of transmission and a low data rate compared to other current technologies. Hence, it is the technology that suits bests with the requirements of IoT development. [13]

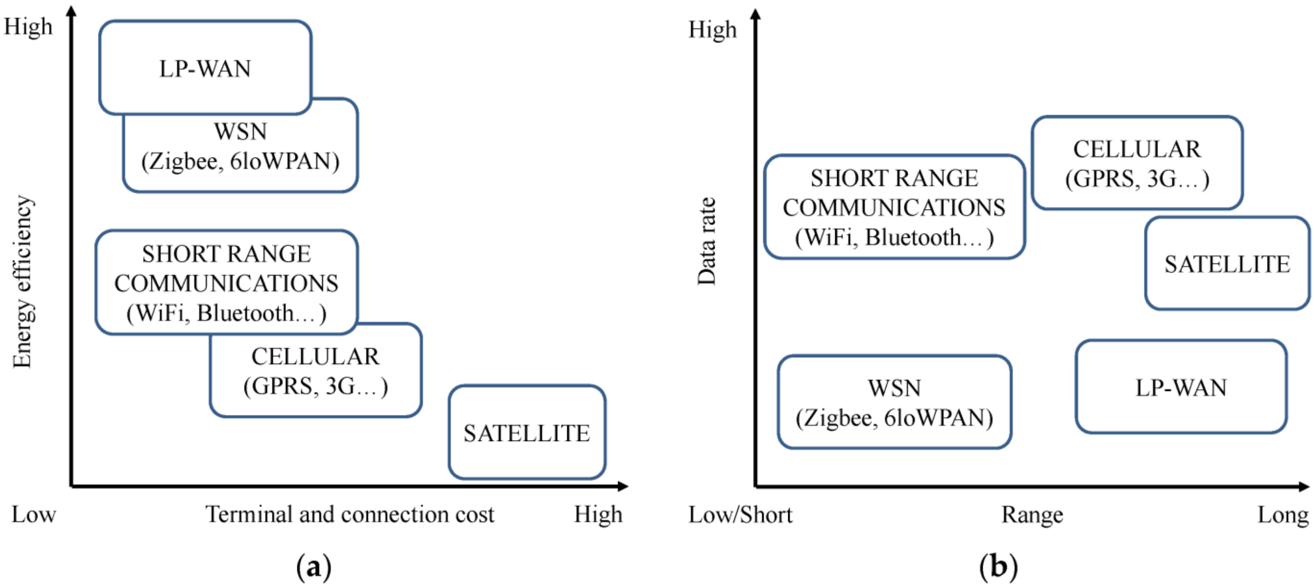

Figure 5: Principal characteristics of IIoT-enabling technologies. Source: [13]

But, why is LPWAN so important for the advancement of IoT technology?

This type of network is especially useful for M2M and IoT communication since it operates with greater power efficiency than traditional mobile networks. Hence, it supports a greater number of connected devices over a larger area and allow the batteries of the sensors to have a longer life expectancy—at least of 10 years— without recharging them.

Another reason to use this technology in IoT projects is the capability of achieving long-range transmissions because some requirements will involve sending data from afar devices. LPWAN is very useful sending a message across rural areas as it has and an excellent geographical coverage—it covers up to 40 km in the open field.

Moreover, the amount of data that can be sent is very limited— 20-256 bytes per message are sent several times a day— which completely suits with the type of data sent by IoT sensors.

The architecture of this network is described Fig. 6. The network architecture has similarities with that of a cellular network, which means that a series of base stations provide direct connectivity from edge-devices to the network and, then, to the cloud, where data is accessible and stored.

The difference of this type of network is that end-nodes have a direct connection to the base station, and, thus, simplifying the network management complexity and decreasing energy consumption because of the lack of routing tasks. As shown in the figure, all edge-nodes have a direct connection to the base station, which is called star topology.[13]

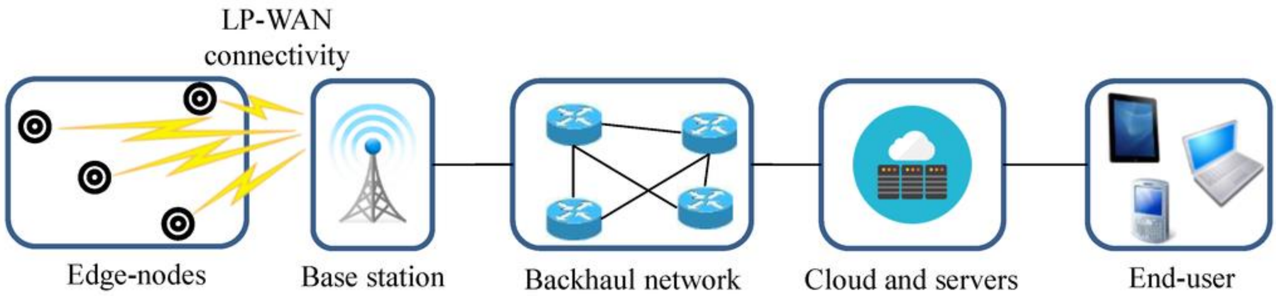

Figure 6: LPWAN network architecture. Source: [13]

However, the LPWAN technology comprises a group of low-power technologies with its own distinctive features. Usually, each technology is provided by different companies, whose aim is to increase the competitiveness of its own product.

Having the best technological solution does not always lead to success. For example, we have seen that ultra-narrow-band technology presents a series of drawbacks in comparison with other modulation schemes that offer better connectivity. However, Sigfox seems to be very attractive to potential customers due to its simplicity and its higher degree of deployment. [13]

Another type of LPWAN technology is LORA, which offers a higher bandwidth than Sigfox but a lower power and it is not an open standard technology.

To sum up, the importance of this technology in IoT communication lies in the long range, low data rate and low power consumption features that it is able to provide.

In this project, the Sigfox technology will be used because the evaluation board used to emulate the IoT devices sends data to Sigfox Backend. In the following section, this technology will be further explained.

4.3. Sigfox

Sigfox [https://www.sigfox.com/en] is a French company that has developed a LPWAN technology. This technology works with the UNB transmission technology, which employs narrow channels in order to achieve higher distances using the minimum energy.

With the UNB modulation, Sigfox exchanges radio messages over the air with a frequency of 200kHz. Each message is 100 Hz wide and transferred at 100 or 600 bits per second a data rate, depending on the region. Hence, reducing the problems of the noise over long distances and increasing its robustness.

The security of the network has a well-thought planning since every device has an identification code and it uses VPN and https coding protocols.

The functioning of this network is very similar to the ones of cellular style network since Sigfox works with the installation of multiple receivers and transmitter stations. The difference between Sigfox and other cell phone technologies is that the devices connected to Sigfox can convey messages to any Sigfox station, by average 3 stations, and from there it can reach the Sigfox Cloud.

In Fig. 7, it can be seen the stages the data has to go through.

Figure 7: Sigfox network architecture. Source: [14]

Each message is sent from the device unilaterally using a pseudo-random frequency, and the receiver station has no previous communication with the device about each message. All the stations that have received the message send it to the Sigfox Cloud. There, the message can be seen using Sigfox Backend [https://backend.sigfox.com/welcome/news] or it can be sent to any other cloud platform such as AWS, IBM Watson or Azure. [14]

The Sigfox network performance is characterised by the following [15]:

- Up to 140 messages per object per day
- Maximum payload size for each message is 12 bytes
- Wireless throughput up to 100 bits per second

The Sigfox technology functions thanks to the use of ISM radio bands. These radio bands are used all over the world for industrial and scientific purposes. In this case, the 868MHz band is widely used in Europe whereas in America the used band is 915MHz.

A regular message can travel an average distance of about 30-50km in rural areas and between 3-10km in urban areas. The less conveying capacity in cities is due to the number of obstructions and hence the noise is greater, so the message cannot travel longer. [15]

The distance at which data can be transmitted will influence the design of the Sigfox network. That is to say, the density of the cells has to be higher enough to support the correct communication of all messages.

At this point of reading how Sigfox Technology works, you might be asking why it is so important to the development of IoT devices. The answer is that the features that Sigfox provide fit utterly with the needs of IoT devices.

DESIGN CHOICES	BENEFITS				
	Long device battery life-cycle	Low devices cost (SW/HW)	Low connectivity fee	Spectral efficiency High network capacity	Long Range Wide Area Network (LPWAN)
Small messages	✓			✓	
Few messages per day	✓			✓	
No signalling for MAC	✓	✓			
Pseudo-random access to PHY				✓	
Ultra Narrow Band (modulation + bitrate)		✓		✓	✓
Transmission power	✓	✓			✓
Operation in unlicensed bands			✓		

Figure 8: Design choices and benefits. Source: [14]

In Fig. 8, it can be seen what the positive aspects of choosing each design option are.

First of all, the payload size, that is to the say the size of the message, is very small. They cannot be larger than 12 bytes. Nonetheless, this size is not a problem for the kind of information that IoT sensors are delivering.

For instance, in Fig. 9, it is shown the size each device needs to convey information regularly. A geolocator sensor can convey one GPS coordinate with 6 bytes. A temperature sensor can send its temperature reading with 2 bytes. Also using only 2 bytes it can send one reading of the speed of a vehicle.

Moreover, with just 1/8 bytes the device is saying that it is on or off, or sending any other binary information. It can also send a message of 0 bytes, meaning the message itself is the communication and saying that the device is working correctly. [14]

Figure 9: Kind of data a Sigfox message can carry. Source: [14]

Secondly, and as a consequence of the former, Sigfox devices have a long battery life-cycle, between, 8 and 10 years, which means that you can have this device attached to your sensor without needing constant supervision or changing of the batteries.

4.4. TD1208 EVB

The TD 1208 EVB [http://rfmodules.td-next.com/modules/td1208/] is an evaluation board that emulates the behaviour of an IoT device. It contains the gateway TD 1208, which is designed by Telecom Design, and supplied by “AVNET Silica”.

Telecom Design TD1208 devices are high performance, low current SIGFOX™ gateways. The TD1208 device versatility provides the gateway function from a local Narrow Band ISM network to the long-distance Ultra Narrow Band SIGFOX™ network at no additional cost.

The broad range of analog and digital interfaces available in the TD1208 module allows any application to interconnect easily to the SIGFOX™ network.

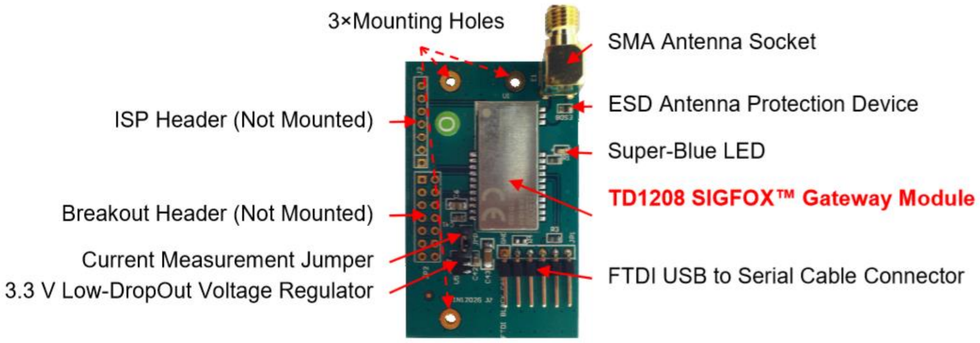

Figure 10: TD EVB 1208 Top View. Source: [16]

The board main features are:

- TD1208 SIGFOXTM Gateway module in LGA25 package
  - SIGFOX ReadyTM
  - Frequency range = ISM 868 MHz
  - Receive sensitivity = -126 dBm
  - Modulation
    - (G)FSK, 4(G)FSK, GMSK
    - OOK
  - Max output power
    - +14 dBm
  - Low active radio power consumption
    - 13/16 mA RX
    - 37 mA TX @ +10 dBm
  - Power supply = 2.3 to 3.3 V

These features are found in [16].

While reading the TD1208 EVB User's Guide it is important to follow the set up requirements so as to install the EVB TD1208 as soon as possible.

The requirements are:

- A PC running Windows (this is only required for being able to flash the device using the provided utility program, for a simple connection to the module, any operating system for which FTDI devices are supported should work)
- A Web browser running on the PC with access to the Internet
- A serial terminal emulation program running on the PC (In this project it has been used PuTTy [https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html])
- The FTDI Virtual COM Port Driver (VCD) which is appropriate for your machine [http://www.ftdichip.com/Drivers/VCP.htm]
- The Telecom-Design "TD1208Loader.exe" utility program in order to reflash the TD1208 module firmware [https://developers.insgroup.fr/cloud-on-chip/td1208/download.html].

These requirements are found in [16].

5. Configuration of the TD 1208 EVB

As said before one EVB TD 1208 has been used to simulate the behaviour of an IoT device.

First of all, it is necessary to connect the EVB TD 1208 to one terminal, which will send the orders. In this project, it has been used a computer as the terminal. The power needed to make work the TD1208 EVB comes from the USB port connection.

Then, the FTDI black wire is aligned with the label on the TD1208 EVB board corresponding to the GND pin, as shown in Fig. 11. The SMA swivel antenna must be connected to the TD1208 on-board SMA socket and positioned perpendicularly to the TD1208 EVB board top surface.

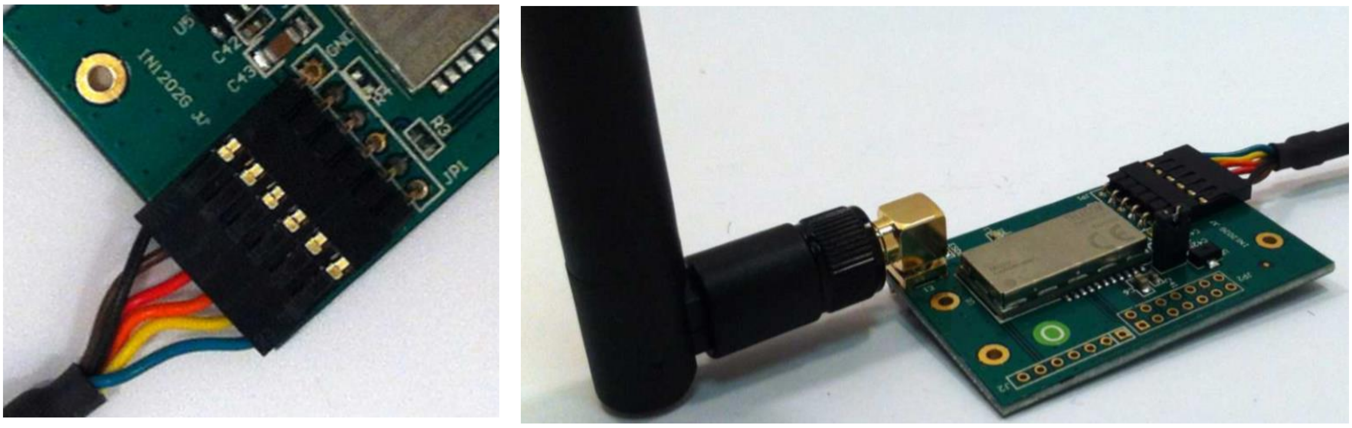

Figure 11: FTDI cable connection and General View of the EVB TD 1208 working. Source: [16]

Then, it is compulsory to install an FDTI VPC driver. The main function of the driver is to convert the USB port into a VPC.

In the computer, it is necessary to install one terminal programme, such as “PuTTY”, which will act as an emulation software. This programme allows to send data from the computer to the Sigfox Backend.

As the serial terminal emulation software will require the virtual port corresponding to the newly attached device, the best way to get it is to use Window’s Device Manager from the Control Panel, by clicking on the System icon and selecting the Hardware tab and pressing the Device Manager button.

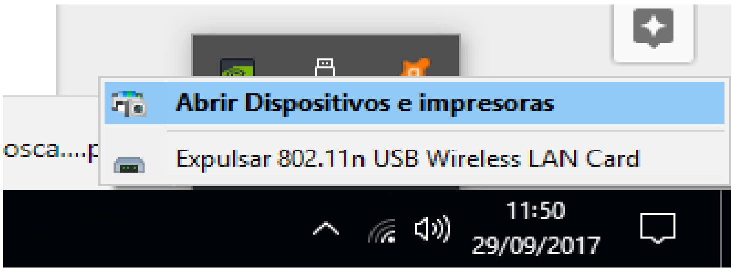

Figure 12: Desktop screen shot. Source: own

Now, the Ports (COM & LPT) entry has been unfolded into the device tree list. An USB Serial Port (COMx) entry corresponding to the newly attached TD1208 EVB device should be seen. If unsure, the USB cable can be unplugged and plugged again so as to observe the changes in the “Device Manager” window.

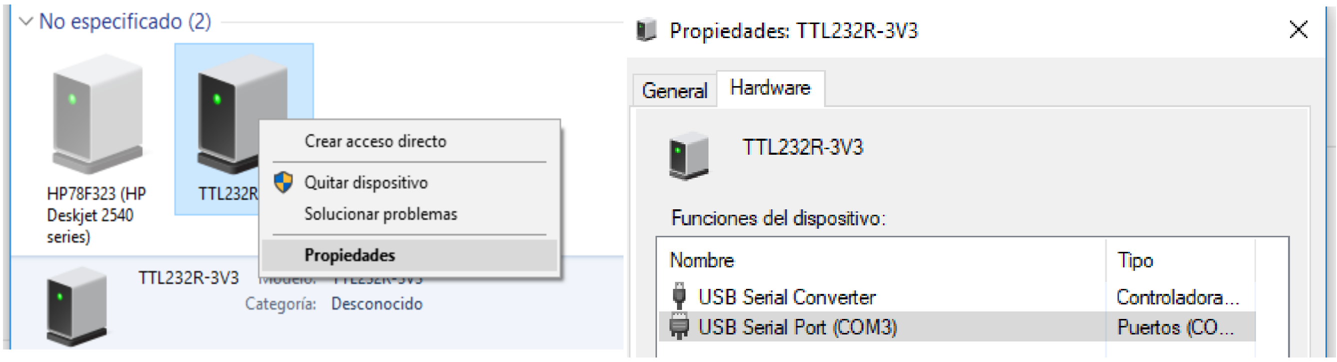

Figure 13: Desktop screen shot. Source: own

It is necessary to write down this COMx information, so it can be provided later to the serial terminal emulation software. Right after, the selected serial terminal emulation software has been launched, with the following serial parameters.

- Port as obtained from Window’s “Device manager”
- LVTTL electrical level
- 9600 bps
- 8 data bits
- No parity
- 1 stop bit
- No hardware/software flow control

All this procedure is found in [16].

Inside PuTTY the configuration has to be changed. As connection type, Serial has been chosen and as serial line COMx information that it has been written down previously from Window’s Device Manager has been written. Now, the terminal has to be opened.

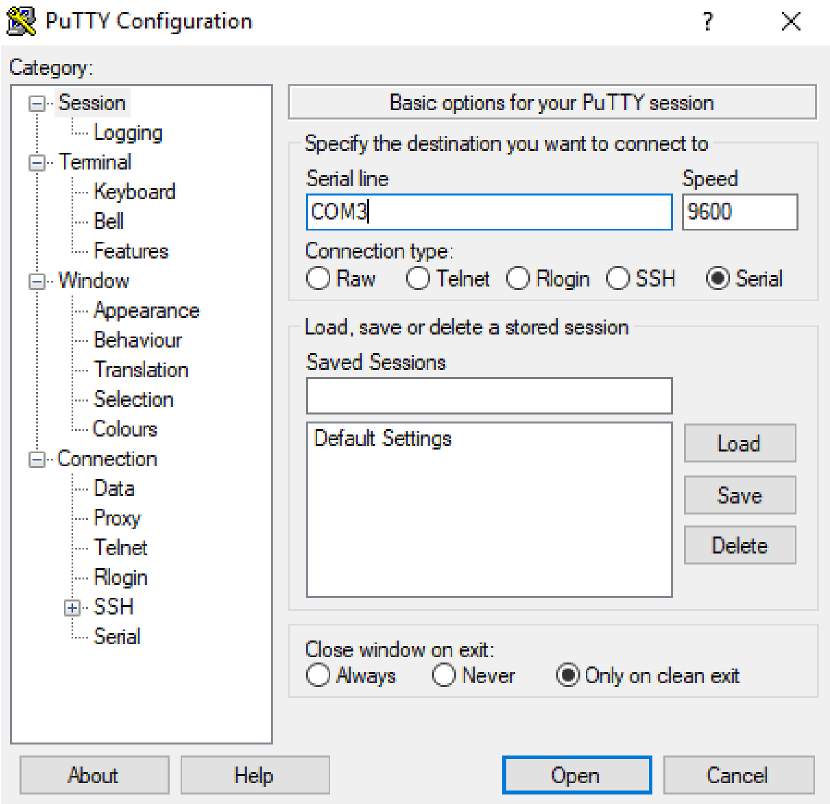

Figure 14: PuTTY interface. Source: own

From here, the messages can be sent by following the sending requirements that are explained later on.

In order to see the messages, it is necessary to enter to the Sigfox Cloud [https://build.sigfox.com/backend-callbacks-and-api]. To do so, a subscription to the Sigfox Backend is provided with the purchase of the TD1208 EVB. This Sigfox backend is the interface with which the user can manage to access the Sigfox Cloud, where all the messages from the station are sent.

5.1. How the TD1208 module works

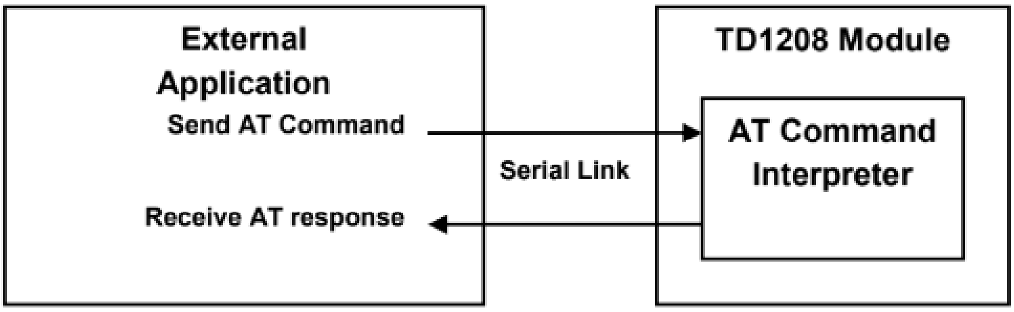

Figure 15: AT Command Interpreter. Source: [17]

The TD1208 module provides an integrated dual AT command interpreter for interfacing with an external application over a serial link. The external application, in this case, PuTTY, is managed by a computer.

The serial link is the process of sending data one bit at a time, sequentially, over a communication channel. In contrast, in parallel link, several bits are sent at the same time since the link has several parallel channels, so the information can be sent as a whole and not fragmented like in the serial link.

Once the TD1208 module sends to the network the message, the external application PuTTY receives a response pointing out that the message has been sent correctly.

To ensure that the Sigfox Device responds to the PuTTY, a prove message has been sent and it has been checked that it reaches the SigFox Backend. As it can be seen Fig. 16. if you type the following command AT&V<CR>, the following should be got:

Where “<CR>” represents a press on the Carriage Return key and the black letter the command you type on the Console. [17]

```
AT&V<CR>
```

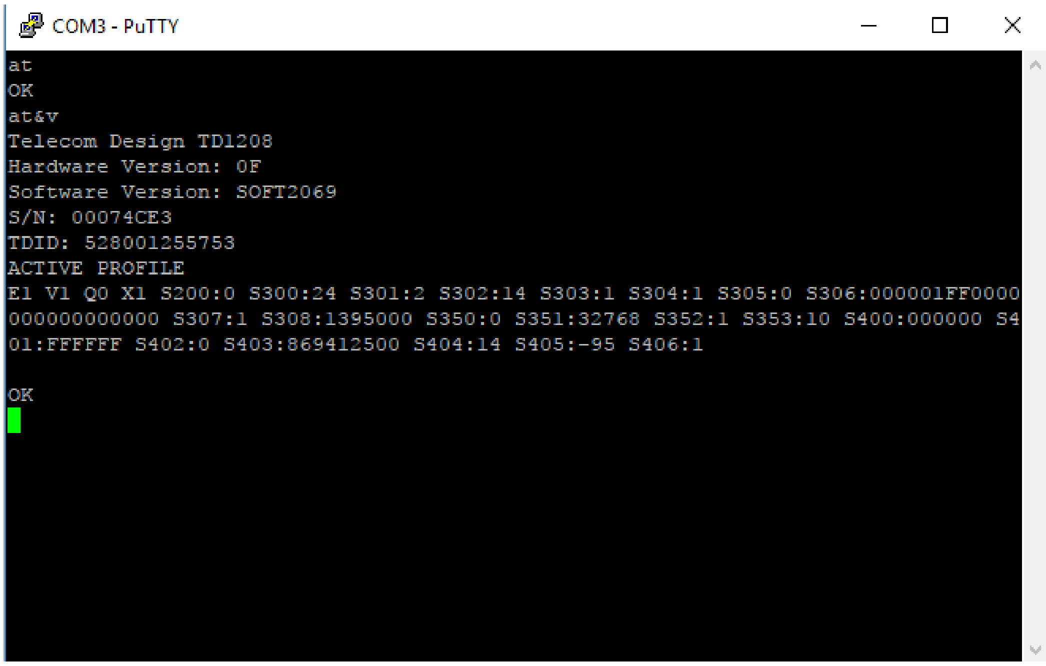

Figure 16: PuTTY Console. Source: own

5.2. Sending a message Command

AT$RAW=<hex_byte1> [[ ]..<hex_byte2>..]<CR> OK

This command sends a Raw Message. If the Module Type is set to Device, then a Local Register and a Sensor Register command must have previously been issued for the message to be received. It the Module type is set to either Gateway or Transmitter. A Sensor Register command must have previously been issued for the message to be received.

Which are the characters you can write in it?

The data can be conveyed with 2-digit hexadecimal byte value (‘0’ to ‘9’, ‘a’ to ‘f’ and ‘A’ to ‘F’ characters are valid)
There can be from 0 to 10 <hex_bytex> parameter values, optionally separated by single or multiple space (IA5 2/0) or tabulation (IA5 0/9) characters.

An example would be:

```
AT$RAW=01 02 03 04 05 06 07 08 09 0A
OK
```

More information about this command can be found in [17].

5.3. Temperature Command

This command causes the device to transmit one or more lines of specific information text. In this case, the temperature of the module.

```
ATI26
Module temperature in ºC
OK
```

Display module temperature in °C as a decimal number.

5.4. Python Programme that gets the temperature and sends it to the Sigfox Cloud.

As mentioned before, one of the aims of this project is to construct an IoT architecture of a temperature sensor. The temperature sensor is included in the TD 1208 chip and it will give the temperature surrounding the chip.

With the ATI26 and AT$RAW commands, it is possible to get the temperature and send a message with raw information. Then, the message reaches the Sigfox cloud through all the antennas of the Sigfox network.

Once in the Sigfox Cloud, the message has to be resent to one of the four IoT platforms that will be compared: IBM Watson, AWS, Azure Microsoft and thethings.io. Each platform has to be configured in order to be able to receive and store the messages.

The most important part of the process is the analysis of the data sent from the TD1028. To do so and to have a clearer and more direct insight of the data we have, the best tool is to use visualization techniques. It will be shown how to configure the platforms to make visualizations.

First of all, it is necessary to get and send the temperature data. The TD1208 module has the commands that carry out these actions but to make things easier, a Python programme to automatize this process has been developed. (see Annex 1)

It is necessary to install the Python module serial since it will be needed to replace the terminal emulation programme, in this case, PuTTY. It is necessary to configure the serial attributes as it has been done in the PuTTy interface. So, where there is the port information, it is necessary to put the port where you have connected the TD 1208 EVB.

Extract of the programme code:

```
ser1 = serial.Serial(
    port='COM#',
    baudrate=9600,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_ONE,
    bytesize=serial.EIGHTBITS
)
```

The two main functions are GetTemp() and SendRaw(). The former uses ATI26 command to force the chip to send temperature data whereas the latter handles the message sending by using the AT$RAW command.

The message position in which the essential information will be has to be taken into account. This will be crucial later, when configuring the IoT platforms, since it will be required to parse the message and say what each pair of bits consists of.

This programme sends the temperature information in the third and fourth pair of bytes of the message. As stated, bear in mind this fact since it will be compulsory to be used in further steps.

Another necessary insight of the code to take into account is the interval in which the messages are sent. In this project, it has been convened, after studying the changes in temperatures and its rapidity, that an interval of 30 minutes is enough to notice all the temperature changes with accuracy enough.

However, had this interval required changes, it is only necessary to modify the number of seconds at the command time.sleep().

```
time.sleep(1800)
```

In Fig.17, it can be seen how the script automatically gets the temperature of the TD1208 chip and delivers the data to the Sigfox Cloud (see code in Annex A).

In order to start the process, the programme has to be run and the most straightforward path to do so is shown in Fig. 17.

```
C:\Python27\serial\pyserial-3.2.1>python TestScript2.py
True
```

Figure 17: Code to start the Python programme. Source: Own

Then, once the programme has been started, it runs steadily until it is told otherwise, and it keeps conveying the data to the Sigfox Cloud.

```
C:\Python27\serial\pyserial-3.2.1>python TestScript2.py
True
OK
['9', 0]
END
09
[65, 84, 36, 82, 65, 87, 61, 32]
OK
['8', 0]
END
08
[65, 84, 36, 82, 65, 87, 61, 32]
OK
['8', 0]
END
08
[65, 84, 36, 82, 65, 87, 61, 32]
OK
['8', 0]
END
08
[65, 84, 36, 82, 65, 87, 61, 32]
OK
['7', 0]
END
07
[65, 84, 36, 82, 65, 87, 61, 32]
```

Figure 18: Python programme running. Source: Own

6. IoT Architecture

In order to visualize the data conveyed by the sensor and to analyse them to get useful insights, it is necessary to follow a staged architecture. This IoT architecture is personalized for each IoT service or functionality and it explains how the data flows between the devices and the services that process them.

The most challenging aspect when designing an IoT architecture is its complexity. Hence, it is necessary to compartmentalize the different functions of the process into stages. Each stage has to accomplish both an individual and a common purpose.

As a whole, an IoT service architecture can have four parts as stages in a process. The stages are integrated and they act in coordination with each other. Sometimes the edge between one and other is blurry.

The main aspects of each stage of the process are shown in Fig. 19:

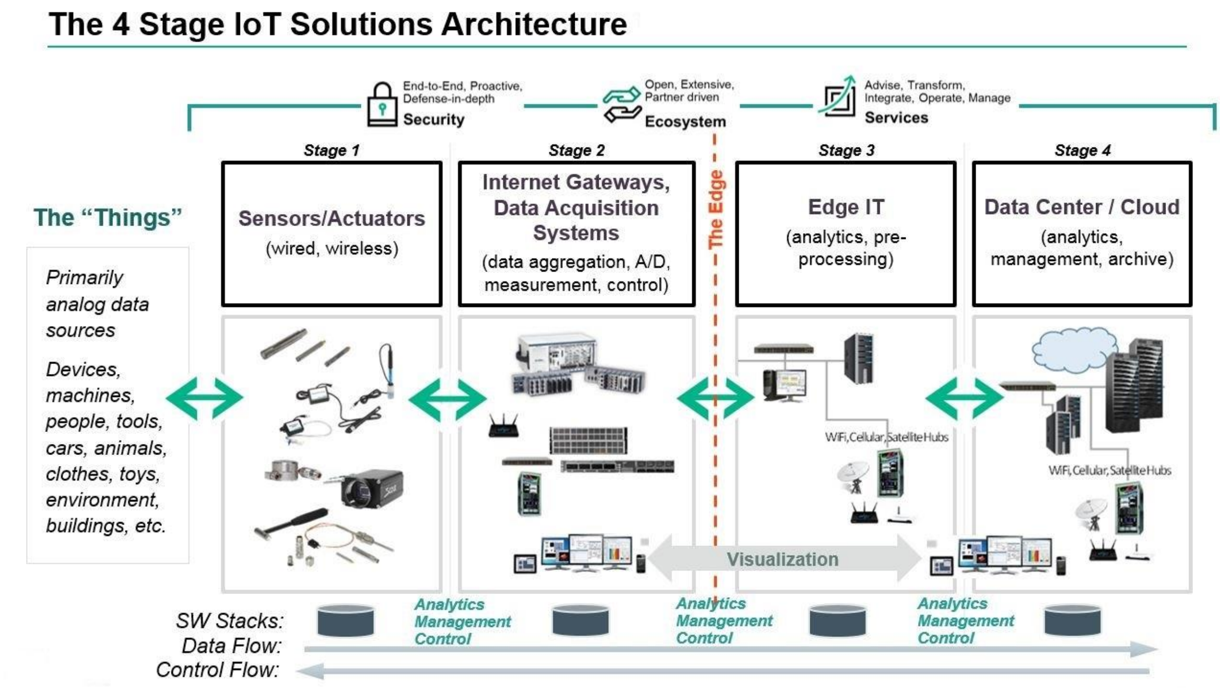

Figure 19: Scheme of The 4 Stage IoT Solutions Architecture. Source:[18]

For example, in all four stages, data processing can be carried out. Nonetheless, the data processing level that a sensor is capable of doing is very limited. It depends on the software power of each IoT device.

The more processing the data is going to undergo, the deeper and more effective insight you can get. The decision to process the data on one or another stage will depend on whether or not you have an immediate need for information.

Considering the type and rapidity of the decisions you have to make, you might need to process the data right after it is conveyed. This will allow you to get the fastest response possible but having lost some information since the processing is not done so thoroughly than had the data reached the cloud or the data centre system. [18]

6.1. The 4 stage IoT solutions architecture

6.1.1. Stage 1: IoT peripheral nodes

These are the sensors responsible for collecting the raw data, which in other stages will be made useful by means of data analysis. These data will be delivered to the control centre and there will be processed in the bulk of the cases.

Actuators are the other type of IoT nodes. These are in charge of changing the physical conditions that generate the data. After an analysis of the data conveyed by the sensors, the actuators receive an order from the control centre in case some change has to be done to the system that is being controlled.

Mobile devices, such as smartphones, tablet PCs, or laptops, may also be an important part of an urban IoT, providing other ways to interact with it.

6.1.2. Stage 2: Internet Gateways and Data Acquisition Systems

The key element in this stage is the DAS. The data acquisition systems are in charge of converting the data given by the sensors from analog to digital form, in order to be easily processed down the process.

The Internet gateways systems achieve the aim of receiving the digital data from DAS and send it to Stage 3 systems. Therefore, gateways interconnect the IoT nodes to the main analysis systems of the IoT architecture.

The data is conveyed by using Wi-Fi, wired LANs or the Internet. Recently, a lot of enterprises have developed different technologies so as to send the data produced by IoT sensors.

Stage 2 systems often are very close to the sensors and actuators. Sometimes they can be mixed in the same MCU whereas in other cases one DAS is used by many sensors at the same time for digitising the data. Then, a nearby gateway device transmits the data to the next systems for further analysis.

The main reason why it is necessary to convert the raw data into digital form is to reduce the amount of data having to be sent to Stage 3 and 4 systems. Furthermore, analog data presents a set of problems, such as specific timing and specific software to meet the requirements of its structure.

6.1.3. Stage 3: Edge IT

This stage is used to make the last processing part before the data enters the data centre. They serve to lessen the quantity of data sent to the data centre. To fulfil this purpose, a preliminary analysis of the data is made and only the most important one is chosen to be sent to the next stage.

6.1.4. Stage 4: Data Centre/ Cloud

The main element in this stage are the backend servers, whose goal is to collect, store and make in-depth analysis of the data. As a consequence, they are capable of producing the most reliable information and added-value services of all four stages.

The downside of making analysis in this stage is that it is not so fast as the decision-making process in Stages 1 and 2 since we have to wait until the data get there. However, as mentioned, it is useful for deeper analysis considering that data from other sources can be used.

The places where the data is stored are called the database management systems. They store all the necessary information that the sensors are conveying. The dimensioning of the database must be accurate so as not to waste resources unnecessarily.

Another type of service in Stages 3 and 4 are Web Sites where the data is upload so as to be accessible to would-be consumers such as public authorities, service operators, utility providers and common citizens. [8]

Once the data reaches Stages 3 and 4 it is time not only to make a qualitative analysis of them but also make some kind of visualization in order to get a clearer picture of the data we have. Dashboard, maps or graphs are usually used to present the information in a more striking way.

6.2. IoT architecture solutions

In this chapter, the four IoT architecture solutions by using a different IoT Platform to analyse the data conveyed are explained. The Stage 1 and 2 are common to all the solutions since it comprises the configuration of the Sigfox device in order to send the temperature lectures automatically.

In the architecture case that has been studied in this project, Stage 1 consists of a temperature sensor that it is integrated inside the gateway chip, the TD 1208 module. Usually, some sensors are shared by one gateway.

Stage 2 is formed by the TD1208 from Sigfox Technology that is in charge of sending the temperature reading to the Sigfox Network and then it reaches the Sigfox Cloud. In the Sigfox Cloud the message can be viewed but it is difficult to make any kind of analysis.

From the Sigfox Cloud, it can be sent to any IoT Platform by means of a callback API. Right after, the message is conveyed to the chosen IoT Platform where the connected device has been created.

The difference is found in how the connection between the Sigfox device and the IoT Platform is configured and what the power of analysis of each Platform are. For instance, depending on the Platform the data can be visualized or some kind of machine learning can be implemented.

In the next sections, the IoT solutions experimented with each IoT Platform are explained, focusing on the services that are to be configured and how they are connected in order to move the data from the Sigfox Cloud to the visualization part.

6.2.1. AWS Architecture Solution

In Fig. 20, an infrastructure scheme of the project, which has been carried out with AWS, can be seen.

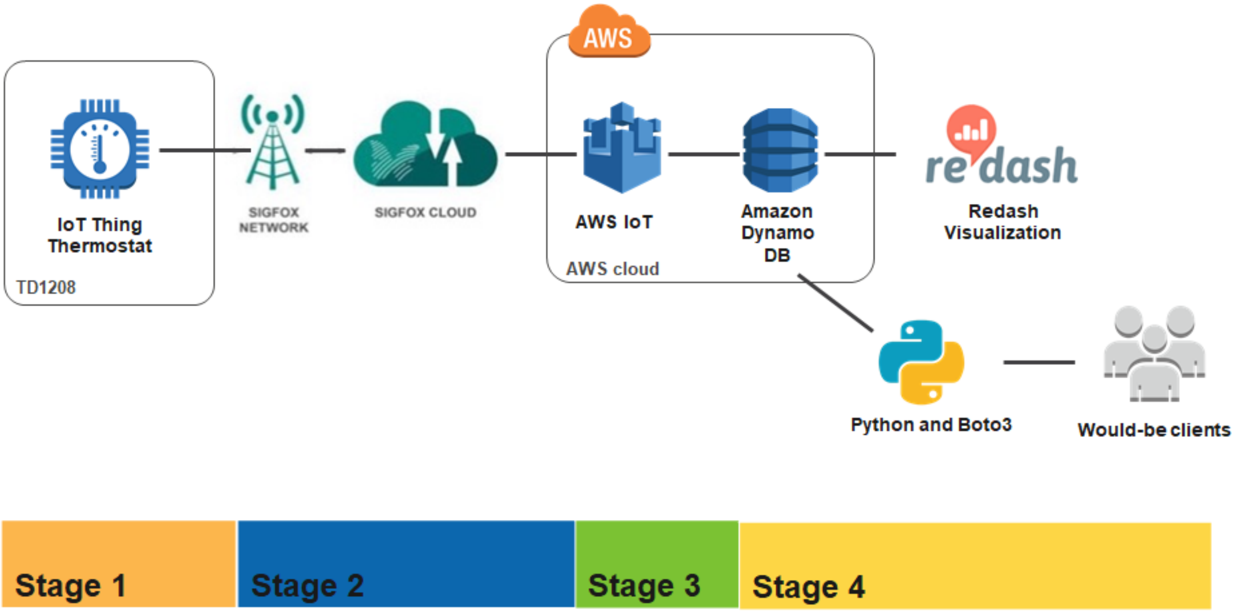

Figure 20: IoT architecture solution using AWS. Source: own

The AWS IoT Device Gateway enables devices to communicate securely and efficiently with AWS IoT. With this gateway, even Things with different protocols can communicate with each other. In this moment of the process, we can state that Stage 3 is on.

Then a Rule Engine is created. The rules engine evaluates inbound messages published in AWS IoT, transforms and delivers them to another thing or a cloud based on business rules the user defines.

The rules engine can also route messages to cloud endpoints such as AWS Lambda functions, or a DynamoDB table.

In Stage 3 the message is pre-processed and stored momentarily in AWS. Just after that, it is sent to Amazon DDB, which is a NoSQL store base. There, it is stored with safety, in case someone will need this data in the future.

When the data enters the DDB service, it can be assured that the data is in Stage 4 of the architecture IoT of this device. Then, the data is sent to a service of another company specialized in visualizations, which is called Redash.

The information arrives at Redash from the DDB service and once there it can be selected by means of a parser. Any kind of visualization that helps to have a broader scope of the data can be done.

Nonetheless, the data stored in DDB service can be looked up at any moment, only by using Python and the Boto3 module, for example. This allows any user to consult the data stored at any time without the need of having the account of AWS in which the data is stored

The Boto3 acts as a manager but without being connected to AWS. The user only needs to delimit the data he wants to download. Then, the data appears in the Command Prompt and it can be handled by the user for further analysis in its infrastructure.

6.2.2. Azure Microsoft Architecture Solution

In Fig. 21, an infrastructure scheme of the project, which has been carried out with Microsoft Azure, can be seen.

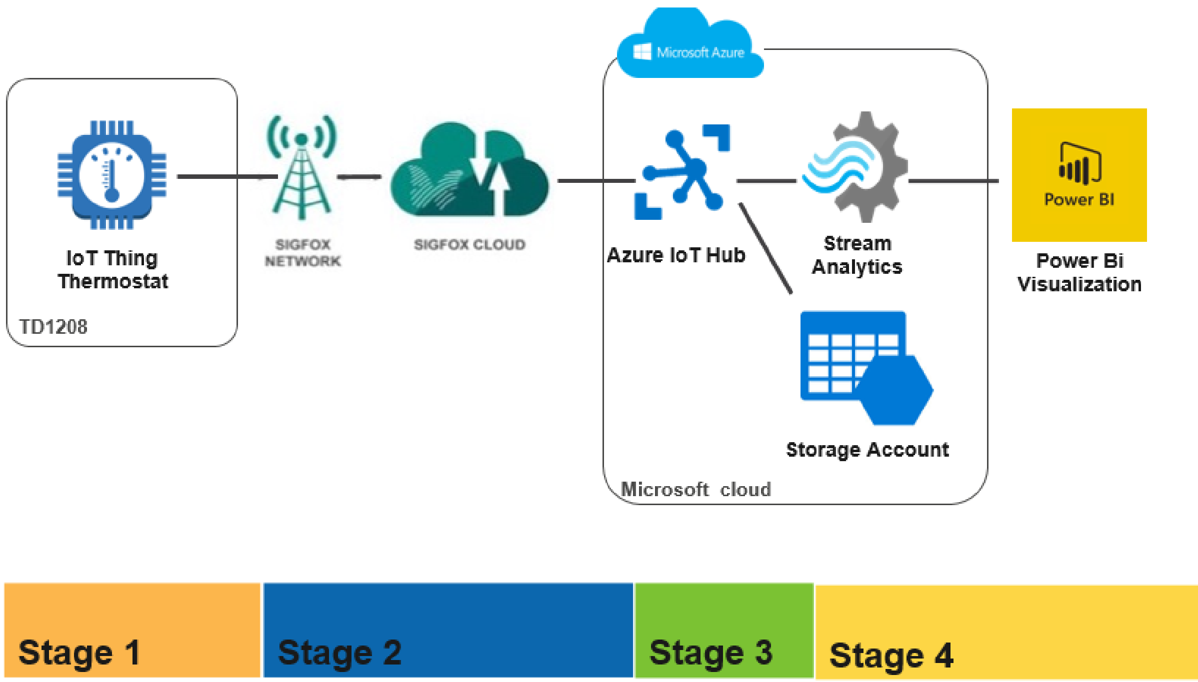

Figure 21: IoT architecture solution using Microsoft Azure. Source: own

The Azure IoT Hub service allows bi-directional communication between the devices and the Microsoft IoT Platform. The configuration of the Sigfox Callback allows the data to go from the Sigfox cloud to the Microsoft cloud. Once the gateway is crossed by the data, it can be assured that the data is in Stage 3 of the IoT architecture.

Once there, the data follows two paths. One path is in charge of keeping the data in an Azure Storage Account using a Blob file kind of storage, where the data can be consulted any time and stored using a low level of storage resources basis. The stored data can be used for extensive analytics, machine learning or other big data initiatives.

The other path consists of sending the data to a service called Stream Analytics, whose main purpose is to enable powerful real-time analytics on data from the IoT device. Azure Stream Analytics is seamlessly integrated with Azure IoT Hub. That means that the data that reaches the IoT Hub arrives instantly to Stream Analytics.

From there, by using an I/O configuration, where the Input is Azure IoT Hub and the Output is Microsoft Power Bi, the data is able to arrive to Stage 4 of the IoT architecture and, hence, a deep analysis and visualization can be carried through.

Then, all the incoming data can be clearly visualized in a Power Bi Dashboard in order to make immediate decisions when the telemetry start to change its pattern or the data can be compared to other application datasets founded there to perform more extensive analysis.

6.2.3. IBM Watson IoT Architecture Solution

The IoT architecture shown in Fig. 22 describes how to create a connection between Sigfox device that stores the data in the SigFox Backend and an IBM Bluemix application where an IoT Gateway is built in order to register Sigfox Device in the IBM Watson IoT Platform (WIoTP) as an individual device. From there, the data can be received in an IoT Application Node-Red development where analysis and visualization can be carried through. In this project, the IoT Application created is very simple but it can be very complex and includes visualizing received data on a dashboard or machine learning. [19]

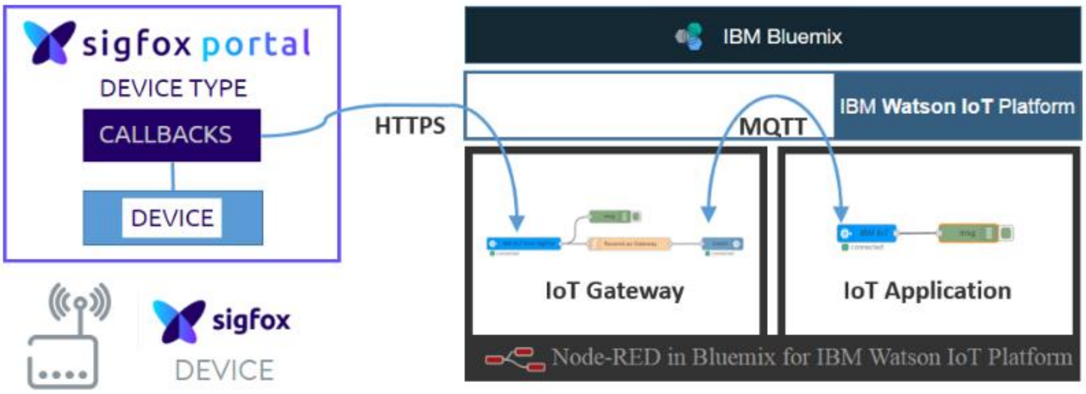

Figure 22: IoT architecture solution using IBM Bluemix. Source: [20]

The data is sent from the Sigfox Cloud to the IoT gateway using HTTPS protocol. In the IoT gateway two device type —one corresponds to the Sigfox backend and the other to the WIoTP application— and one device have been created, which are necessary so as to authenticate the connection between the Sigfox Cloud and the WIoTP application.

Then, the callback definition in the Sigfox backend is created and connected to the Bluemix application by means of a real-time API, which allow to connect the applications, that are in charge of analysing the data, with the data coming from the device.

As it can be seen in Fig. 22, the data is received in the WIoTP at the IoT Gateway and, from there, through the API, the data is parsed and submitted, in MQTT format, to the IoT application created using Bluemix.

This data flow is created using Node-Red service in Bluemix. By using this service, the data flow can be changed and analysed in the way the user needs without having to change the configuration of the transmitting device, in this case, Sigfox.

As recently stated in this section, the Node-Red flow that is in charge of the IoT application can be very complex but in this project it is only capable of receiving the data submitted from the Sigfox Device.

In Fig. 23, an example of an IoT application, is shown, constructed using Node-Red, capable of making the visualization in a dashboard of the readings of temperature and humidity obtained from a Sens'it 2.0 device, which is a device that integrates multiple sensors and conveys data using the Sigfox network.[19]

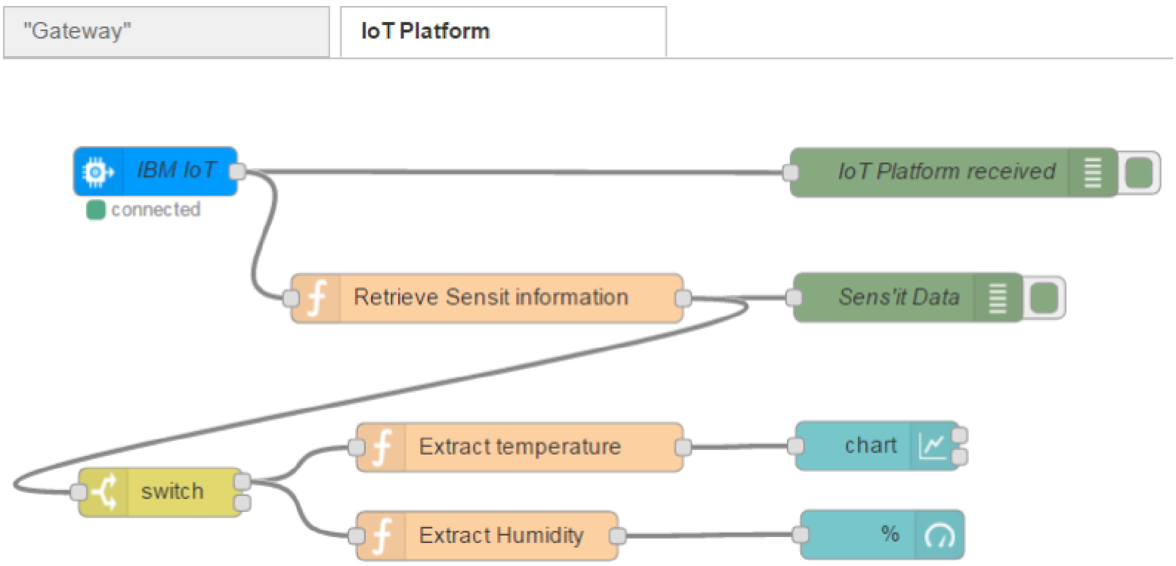

Figure 23: IoT Application flow constructed using Node-Red. Source: [19]

6.2.4. thethings.iO Architecture Solution

The architecture solution implemented using thethings.iO is much simpler than the implemented using the other platforms since receiving , parsing and visualizing the data can be done using the same platform in a very intuitive way.

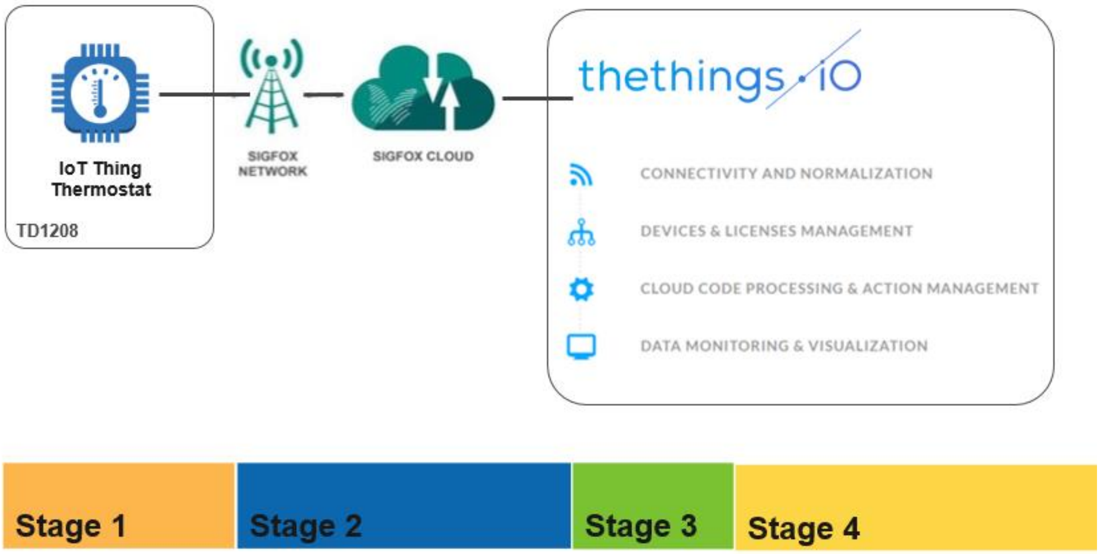

Figure 24: IoT architecture solution using thethings.iO. Source: Own

The connection between the Sigfox device and the IoT Platform is achieved by means of a Callback URL, which is the gate that allows the data stored in the Sigfox Cloud to be transferred to thethings.iO.

Then, in order to let the thethings.iO understand how the message is structured, the data is parsed. Hence, each piece of information is given an attribute, since then, this piece of data is associated with this attribute.

Once the data is stored in the databases of the platform, the analysis process can be carried through. In this platform, the messages can be shown in a table as soon as they arrive to the cloud and then, instantly, the graphic is updated.

Hence, the data that the TD1208 EVB is sending can be monitored in real-time and all the information can be accessed instantly by means of a dashboard.

7. IoT Cloud Platforms

Until now, how the Sigfox device works and sends messages, and the way in which the IoT architecture is constructed have been explained. However, one of the main objectives of this project is to make a comparison of different IoT platforms by using as an example a real IoT case.

As it has been seen in Figs. 20, 21, 22, 24, which depict the IoT architecture solution used for each IoT platform studied, the paramount part of the process when it comes to get useful insight is found in Stages 3 and 4. These stages correspond to the services provided by IoT Platforms.

Before starting to compare the different IoT platforms, it is necessary to define what exactly an IoT Platform is and what kind of services provides.

According to the NIST, “Cloud Computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction”. [21]

IoT platforms suppliers offer a Cloud Computing service that completely suits with the development of IoT technology. In other words, an IoT Platform is a service designed to facilitate the information society, enabling advanced services by interconnecting —physical and virtual — things based on a Cloud Computing model.[22]

The main benefit of choosing an IoT platform instead of creating a new one for every IoT project that an individual or a company develop is the facility in deploying software applications without the cost and complexity of acquiring and managing the underlying hardware and software layers. [22]

All the computing resources offered by an IoT Platform can be rapidly provisioned and released with minimal management effort or service provider interaction

The most important services offered by an IoT cloud platform are application development, device management, system management, heterogeneity management, data management, tools for analysis, deployment, monitoring, visualization, and research. [22]

7.1. Configuration of the IoT Platforms

So as to test each and every platform with the temperature data sent by the Sigfox device it has been necessary to configure the platforms in order to receive the data in the platform cloud, store them in a data base and ultimately make an analysis by visualizing the information in a graphic.

For every platform, the configuration process has been explained and the different degree of achievement made with each one has been pointed out. Moreover, the references to make a further development using these platforms are specified.

7.1.1. AWS Configuration

Following the steps found in Annex D, the communication between AWS and the Sigfox Backend has been achieved. This connection allows the data to be transferred from the Sigfox Cloud to the AWS Cloud, where a more in-depth analysis can be carried through.

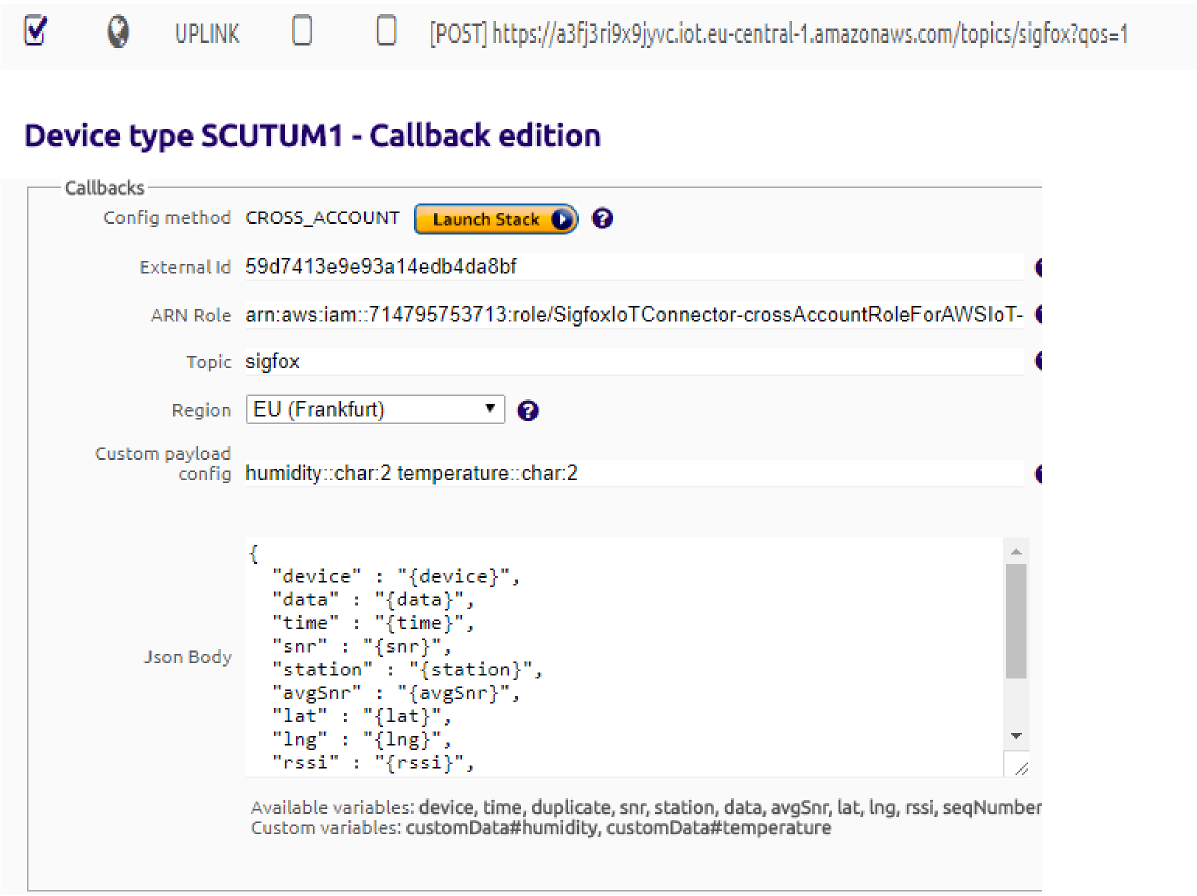

Figure 25: Configuration of the callback between Sigfox and AWS. Source: Own

In Fig.25, the ARN Role can be seen, that is the key between the AWS individual account and the Sigfox Cloud. The Json Body is the way in which the data that the sensor transmits is made understandable for the IoT platform. This link is made by using AWS the IoT service.

Once this connection is working correctly, the data has been stored in a database so as to be able to consult them any moment in the future. To fulfil this aim, the Amazon DynamoDB service has been used.

Amazon DynamoDB [https://aws.amazon.com/dynamodb/] is a managed NoSQL service that provides a strong consistency and a predictable performance, which enable the users not to struggle during the manual setup. The interaction between the user and DDB service is fulfilled by using the AWS Management Console or a DynamoDB API.

The DDB structure is divided into three basic data model units. Attributes, Items and Tables. Attributes are basic units of information. Items are a collection of attributes. Tables are a collection of items.

In Fig.26, different tables can be seen, each one with its distinctive partition and sort keys, which correspond to an attribute. In order to make data visualization in future steps, it has been necessary to choose as partition key timestamp and temperature as a sort key.

Name	Status	Partition key	Sort key	Indexes	Total read capacity
sigfox	Active	deviceid (String)	timestamp (String)	0	5
sigfox2	Active	deviceid (String)	data (String)	0	5
sigfox3	Active	devideid (String)	timestamp (String)	0	5
sigfox4	Active	timestamp (String)	data (String)	0	5
sigfox5	Active	timestamp (String)	temperature (String)	0	5
sigfox6	Active	timestamp (String)	humidity (String)	0	5

Figure 26: Set of DynamoDB tables. Source: Own

In Fig.27, the items and all the messages sent from Sigfox can be seen. The main attributes are singled out, timestamp and temperature, whereas all the other attributes just like the raw data are stored in the payload.

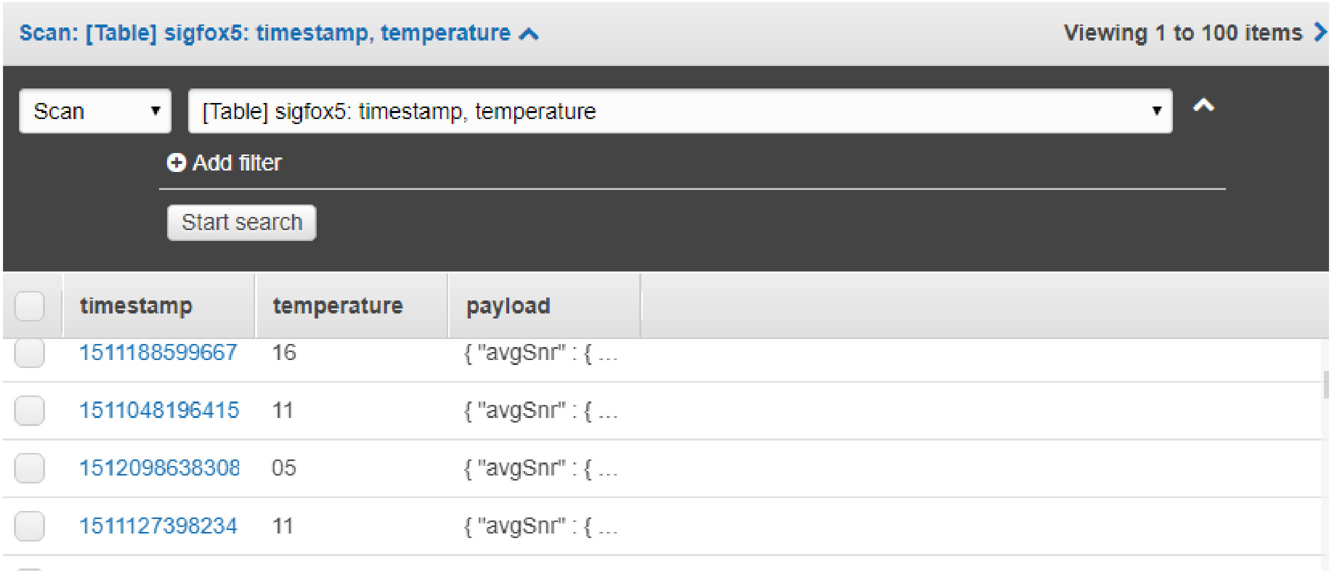

Figure 27: Set of DynamoDB items inside a table. Source: Own

In the payload, where all the attributes with its values are found, it can be seen clearly that the temperature reading sent by the Sigfox device corresponds to the 4 last bytes of the raw data message. Then, the JSON code used while configuring the gateway between the Sigfox Cloud and AWS is what establishes what part of the message is the temperature.

Figure 28: Item with the attributes and the data associated with them. Source: Own

Then, as shown in the architecture solution, the data stored in DDB is transferred to the Redash application to fulfil the visualization of the temperature during the period of time the data has been sent.

In Fig. 27, the data is scanned from the Sigfox5 DynamoDB and immediately it appears in the table that give the information of the Timestamp, the Payload and the DeviceID.

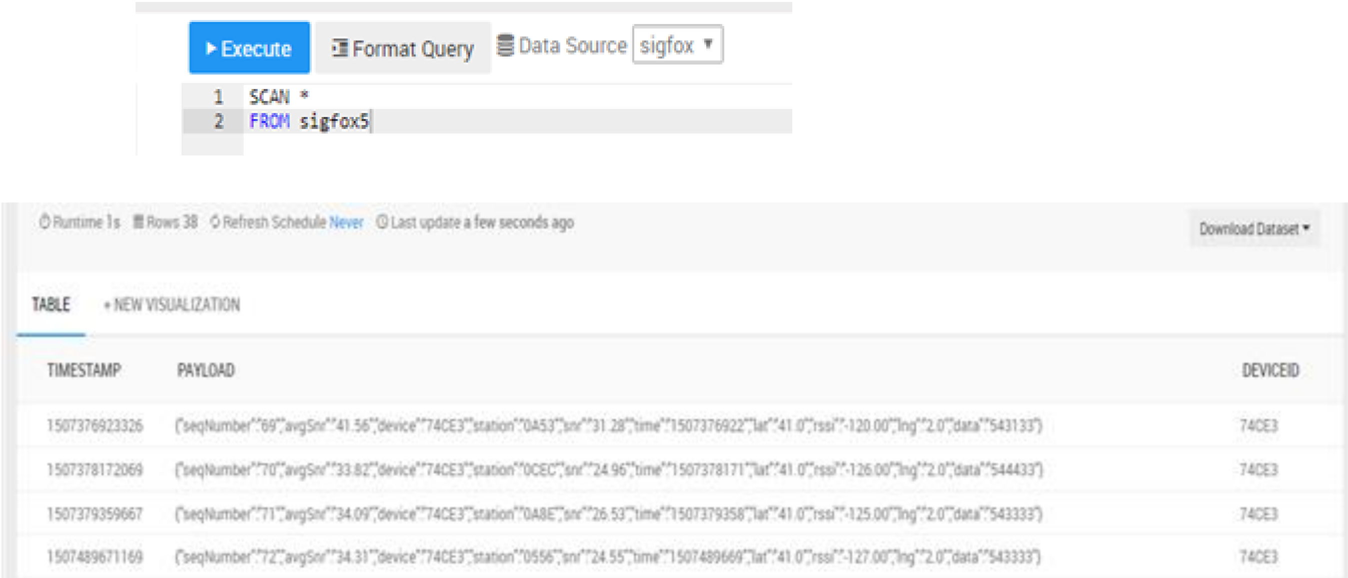

Figure 29: Query and storing of the data in a Redash Table. Source: Own

Then, using this data, many visualizations can be created and posted on a dashboard, where the visualization can be seen at the same time to see how trends interact. In Fig. 28 the created visualization is shown, corresponding to the temperature over the time. It can be monitored how the temperature change over a day.

Figure 30: Visualization Chart in Redash. Source: Own

Finally, the data stored in DynamoDB service can be downloaded by any external user without the need of having access to the AWS account where the DDB service has been developed. The python script that allows this functionality can be found in Annex B.

This has been achieved by creating a python script using the Boto3 module. [23]

In order to accomplish the access of the data to people that do not have the AWS account where the data is stored, it is necessary to provide to python the CLI. This is the AWS Command Line Interface that is used to interact with AWS. The following command has been written in the Command Prompt in order to provide the CLI to the Python.

```
C:\Python27>aws --version
aws-cli/1.14.4 Python/2.7.9 Windows/8 botocore/1.8.8

C:\Python27>aws configure
AWS Access Key ID [None]: AKIAIXHVMCPHEQ743RZQ
AWS Secret Access Key [None]: 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
Default region name [None]: eu-central-1
Default output format [None]: json
```

Figure 31: Introducing the CLI in the Command Prompt. Source: Own

Then, the data appears very mixed in the Command Prompt. After an iterative process of splitting the data between time and temperature and erasing the data that is not necessary the final data is downloaded in the way shown in Fig. 32.

```
C:\Python27\boto3-1.4.8>python downloaddata.py
time
1511997836275
1511945634751
1512098638308
1512165239587
1512190439787
1512114838176
1512249841608
1512143638940
1511832232963
```

```
1511735030730
1512150839079
temperature
08
09
05
05
03
```

Figure 32: Data retrieved using Boto3 that was stored in DynamoDB. Source:Own

First, all the time data appears and, after them, the corresponding temperature readings. Then, these data can be extracted and an stadistical analysis can be carried through with software such as Microsoft Excel , Minitab.

7.1.2. Azure Microsoft Configuration

As explained in Annex E, the configuration to allow communication between the Sigfox Cloud and the Azure Microsoft services has been carried through in this project.

First, an Azure IoT hub service has been created. This service is the gate, the link that makes possible the communication between the two clouds. The connection string associated with the IoT hub is copied in the Sigfox Callback configuration, as it can be seen in Fig. 33.

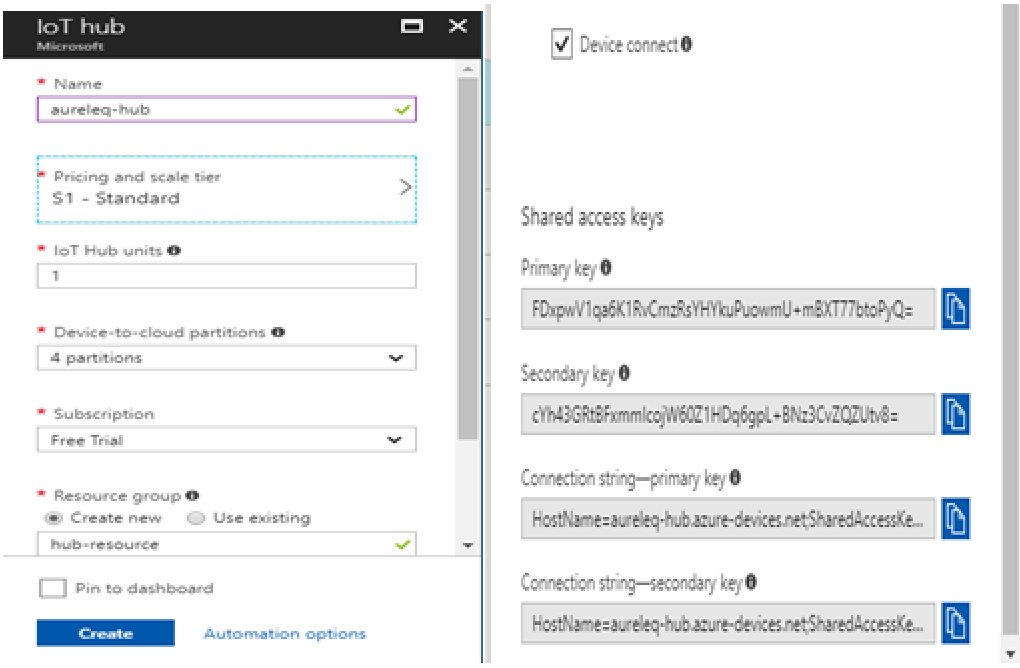

Figure 33: Azure IoT hub connection strings. Source: Own

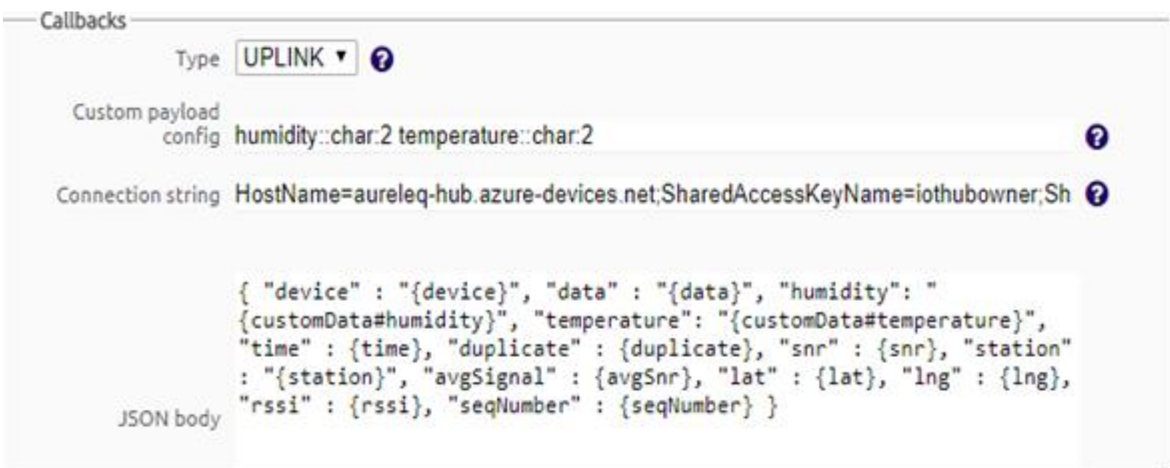

Figure 34: Sigfox Callback configuration for Microsoft Azure. Source: Own

Once the data arrives to the Azure Cloud, it is stored in a Storage account service and the messages appear on the blob section, stored in files that correspond to the day the messages are received.

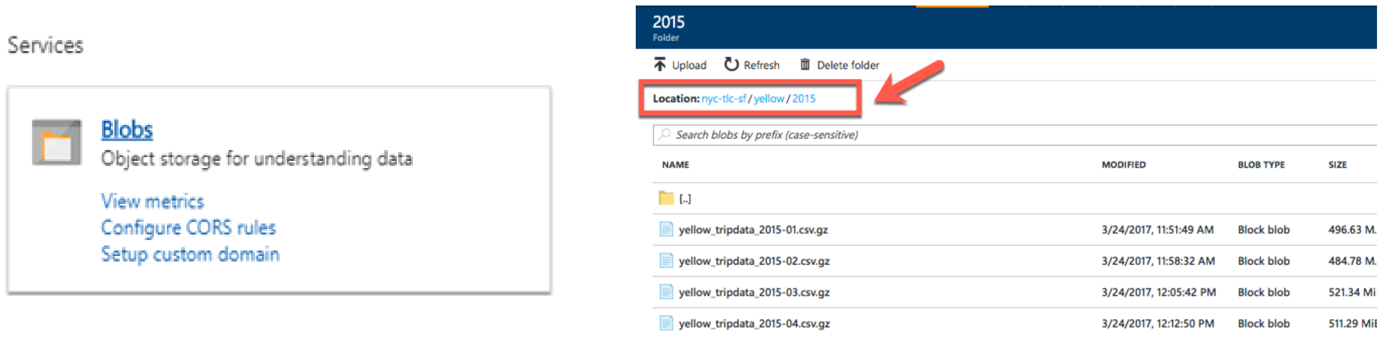

Figure 35: Blob storage account in Microsoft Azure. Source: [24]

To get the data to reach a visualization service like Power Bi, it is necessary to configure a Stream Analytics service, which is in charge of transmitting the data from the IoT hub to the Power Bi database.

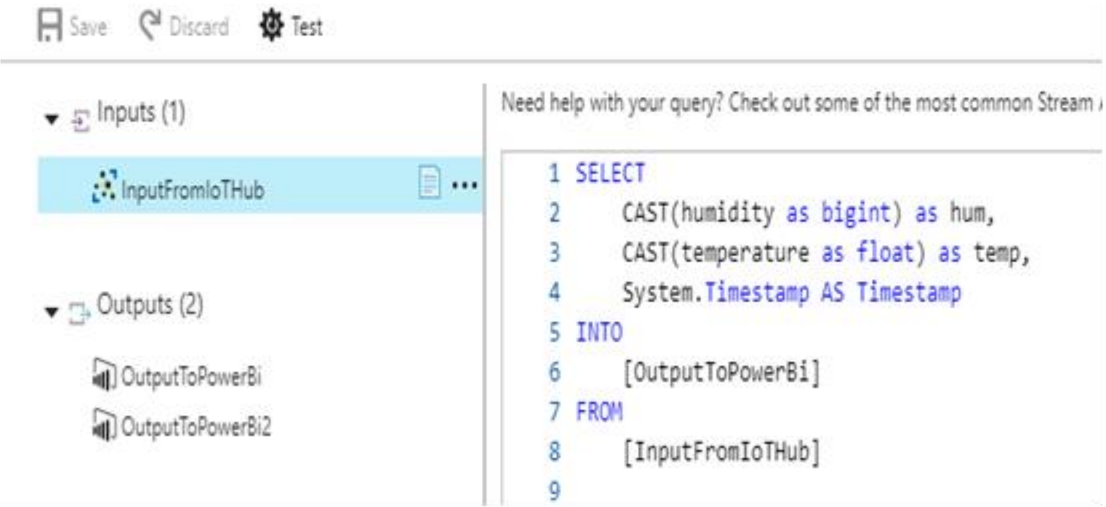

Figure 36: Stream Analytics query. Source: Own

7.1.3. IBM Watson IoT Platform Configuration

In annex C, the complete configuration of how to connect the Sigfox device, which stores the data in the SigFox Backend, and the IBM Bluemix application is explained.

As it has been explained in the two previous paragraphs, the connection is fulfilled by using a string provided in the IoT platform and, then, it is copied in the Sigfox Callback configuration. In the IBM Watson IoT Platform, the process is very similar.

Figure 37 Sigfox Callback configuration for IBM Watson IoT platform. Source: Own

The flow that represents the gateway is shown in Fig. 38. All incoming Sigfox device messages are received and published via MQTT as separate devices. The message structure that is being received in Node-Red application is shown in Fig. 39.

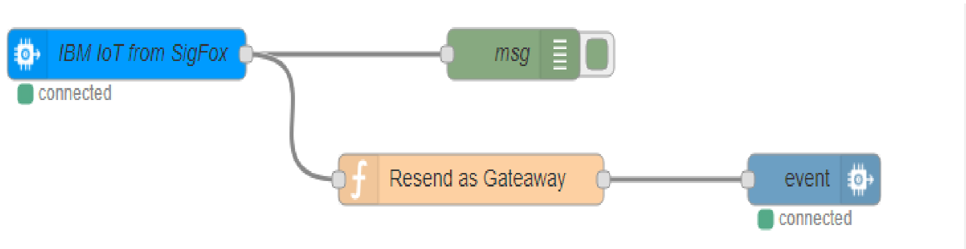

Figure 38: Node-Red Gateway flow application. Source: Own


iot-2/type/SigFoxAuthDevice/id/AuthDevice/evt/status/fmt/json : msg : Object

{ "topic": "iot-2/type/SigFoxAuthDevice/id/AuthDevice/evt/status/fmt/json", "payload": { "time": "1475841149", "deviceType": "Sensit_2.0", "device": "1CB1A5", "duplicate": "false", "snr": "16.77", "rssi": "-26.00", "avgSnr": "113.09", "station": "347E", "lat": "56", "lng": "12", "seqNumber": "233", "data": "816a1a4d" }, "deviceId": "AuthDevice", "deviceType": "SigFoxAuthDevice", "eventType": "status", "format": "json", "_msgid": "b6a90feb.4956f" }

Figure 39: Message structure received in the Gateway application. Source: [20]

7.1.4. thethings.iO Configuration

The process that has been followed to configure the thethings.iO Platform in order to receive the messages from the Sigfox Device and show the data in a visualization graph is described in Annex F.

The steps to follow are much simpler than the steps done in the other platforms. The physical thing, which corresponds to the Sigfox device, which is conveying data, is registered in the platform cloud and a distinctive Callback URL is given.

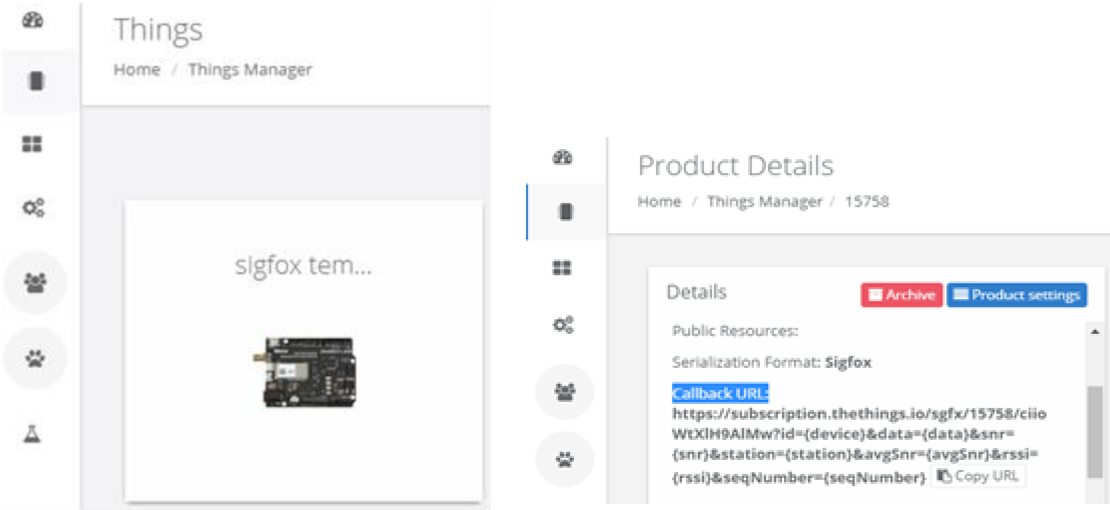

Figure 40: Callback URL associated with the created thing. Source: Own

Then, it is necessary to copy the Callback URL in the Sigfox Backend Callback configuration since it is what allows the platforms to be connected.

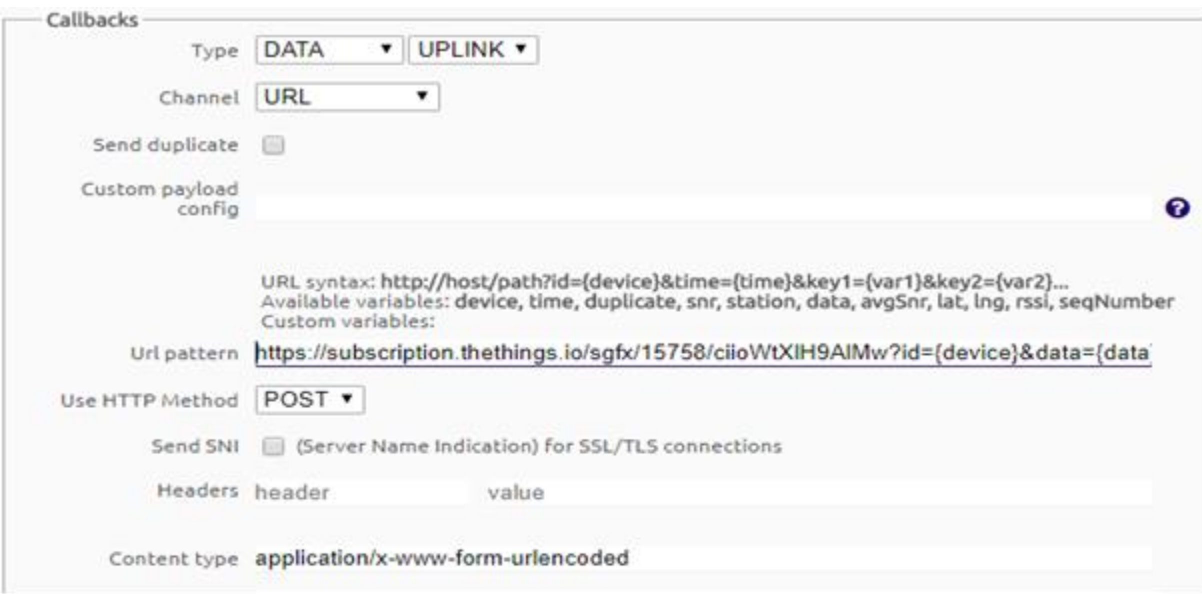

Figure 41: Sigfox Callback configuration for thethings.iO. Source: Own

Then, a parser function has been introduced. This allows decoding every message that is being sent through the Sigfox device, so it will be understood by thethings.io Platform.

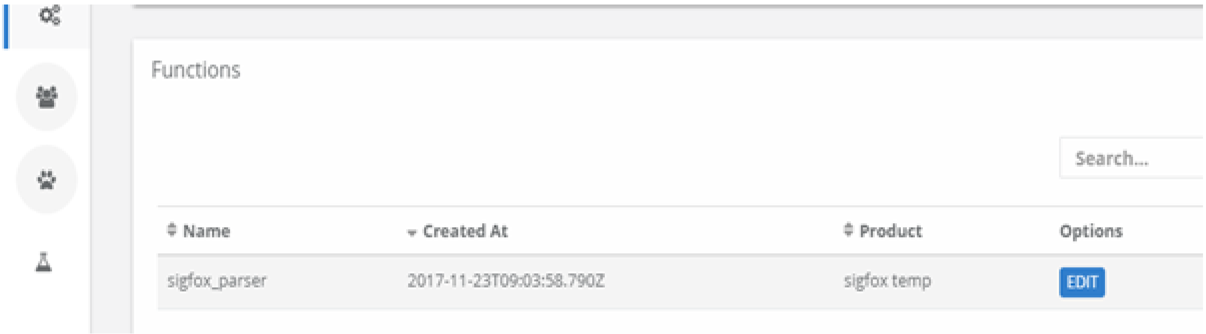

Figure 42: Function where the parser is introduced. Source: Own

Right after, it has been verified that the device is conveying data and that the function parsers correctly the message, thus allowing thethings.io Platform to understand the information.

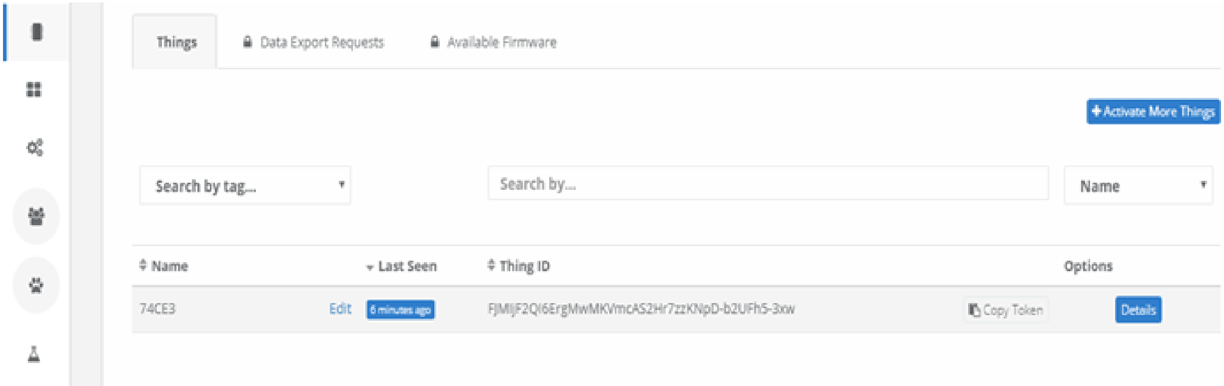

Figure 43: Things Menu. Source: Own

In the analysis part, some graphic visualization of the data has been carried through in a dashboard structure. The dashboard is where all the graphics that have been created can be watched jointly. First, a table, which lets the user to see the messages that have been sent and its attributes and payload, has been created.

Figure 44: thethings.iO table. Source: Own

Finally, a graphic that shows all the temperatures readings during a time interval has been created.

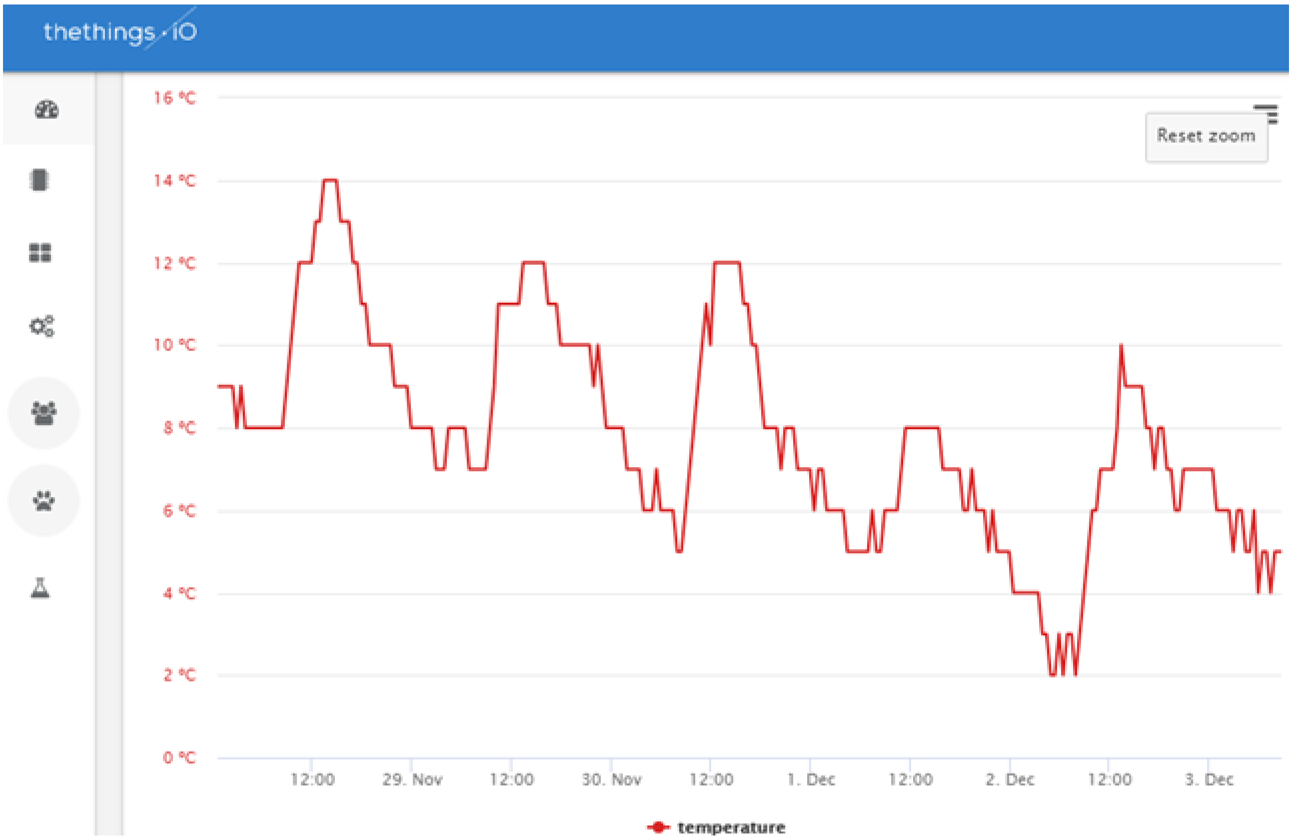

Figure 45: thethings.iO visualization graphic. Source: Own

7.1.5. Next configurations

The configuration process explained above for the four platforms has been carried through during the development of this project. However, due to a lack of time and the complexity of some type of configurations, the visualization using the Microsoft Azure and IBM Watson IoT platforms has not been done.

Nonetheless, only developing a little bit more the configuration done in this project and following some recipes, visualization can be achieved in these platforms.

At the Azure Microsoft platform configuration, the process of configuration ended in the Stream Analytics service. From there, as described in Annex E, the data can be connected with the database of Power BI and then a visual analysis can be done.

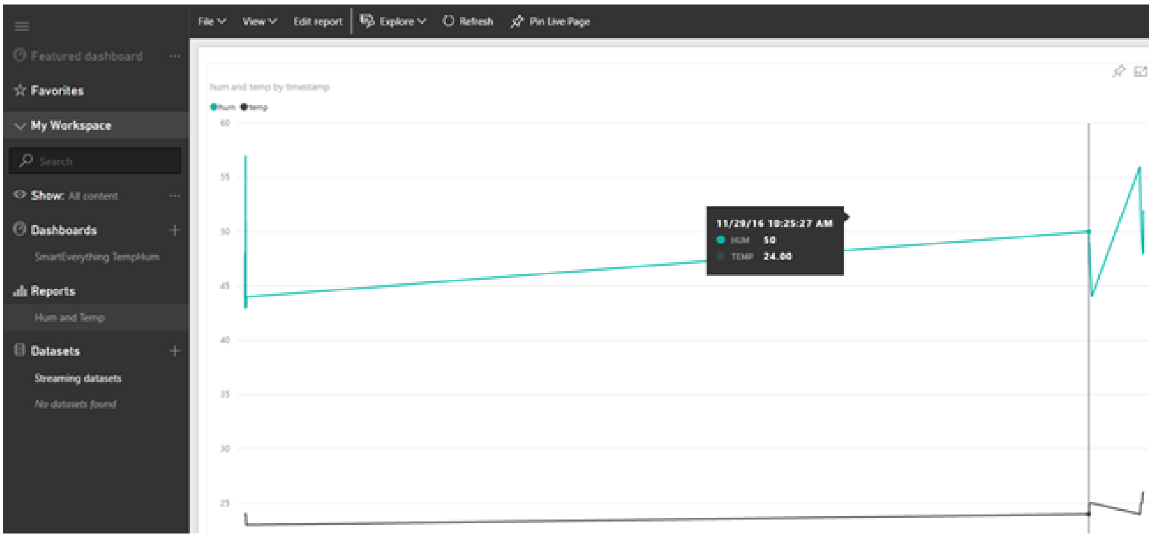

Figure 46: Microsoft Azure visualization graphic. Source: [25]

With IBM Watson IoT platform, by creating a more complex IoT application, like the shown in Fig. 23, the visualization of the data can be achieved using the pre-existing configuration of the data gateway. This process is described in [20] .

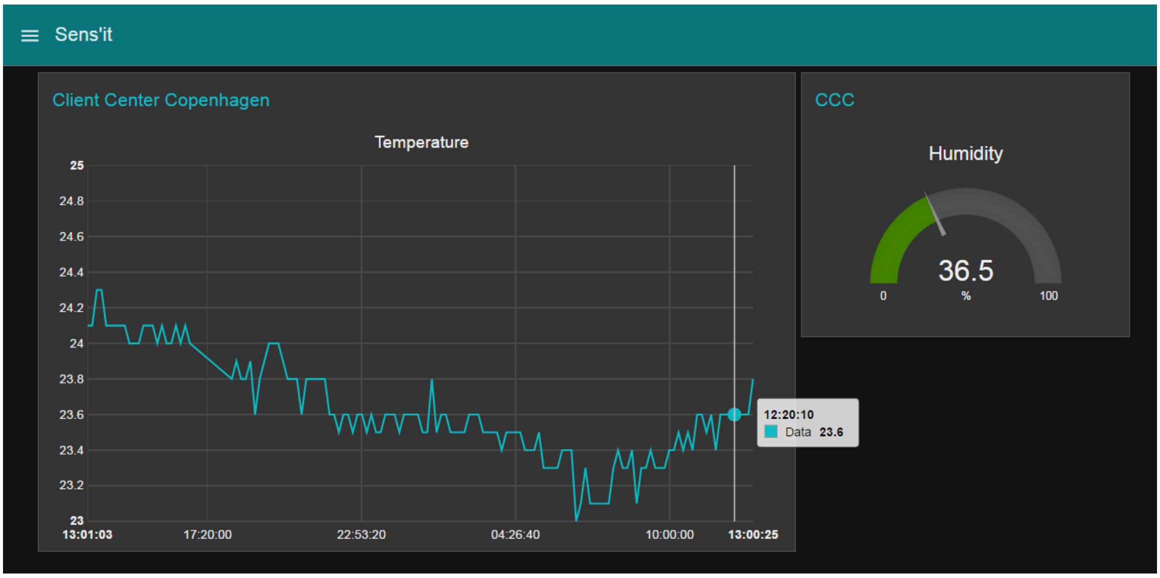

Figure 47: IBM Watson IoT platform visualization graphic. Source: [19]

7.2. Comparison of the IoT platforms

In this section, the different chosen IoT platforms are explained, with their pros and cons, so the reader would make up his mind of what platform can smoothly suit his needs and intentions when using an IoT platform for an IoT application.

Some criteria that have been born in mind when comparing the different platforms and how the IoT architecture has been developed in each one are explained below:

- Flexibility: flexibility on the context of accepting different points of view or protocols. Depending on the platform the service integration, that is to say the capability to integrate with other external systems, will be greater.

- Scalability: when there are few devices connected to the IoT platform the system functions correctly, the problem arises when the number of devices connected drastically increases. An IoT platform should be able to deal with the complexity associated to this increment

- Services: What kind of services each platform has as strength. The most important services used during the development of this project have been data store, data visualization, the device gateway— i.e. the connection between the devices that are sending data and the IoT platform— and how the different services interact.

- Easiness of connection: The difficulty in connecting a Thing, and specifically a Sigfox device. This is measured on the time and difficulty of creating the gateway.

- Easiness of analysis: How the data is stored and the easiness with which it can be accessed and transferred to visualization analysis.

- Pricing: Although all the platforms studied in this report have a free trial account for a period of time, and for example AWS can be used without paying in case the usage do not exceed a limit, it is interesting to point out the different types of payment for the same type of services used.

- Application environment: The possibility to develop new functions and applications in according to the needs of the business strategy.

7.2.1. AWS

The connection between AWS [https://aws.amazon.com] and the Sigfox device has been configured in cooperation between the two companies in order to be easier for the user. It is easy to parse and store the data using a DynamoDB service. The visualization has been achieved using the Redash platform, which is a specialised visualization service that can be easily connected to the DynamoDB service and it is able to make analysis of the data stored there maintaining the attributes structure that had been previously configured.

Amazon's Rules Engine continuously processes data collected by the device and routes to other Amazon services.

7.2.2. Azure Microsoft

The connection between the Sigfox device and the Azure Microsoft platform [https://azure.microsoft.com/en-us/] is achieved without problems. However, some difficulties arise when the data are sent to a storage and visualization service. The used storage service is difficult to consult and the data cannot be downloaded for further analysis. However, other storage services can be used in the Azure platform such as SQL Database as it has been attempted in other IoT projects. [26]

In order to visualize the data, it is necessary to use Power Bi, which is an annex service provided too by Microsoft. Some problems have been found when connecting the data sent to Azure Microsoft to the Power Bi visualization using the stream analytics service.

7.2.3. IBM Watson IoT platform

IBM IoT [https://www.ibm.com/internet-of-things/spotlight/watson-iot-platform] is the hub of all the things where developer can setup, build, and manage the connected devices so that APPs can access their real time data. RESTful and real time APIs help in connecting the data coming from the devices to the IBM Bluemix where applications can be created by the developer. [22]

The connection between the Sigfox device and the IBM Watson IoT platform is very long and it is required to have strong insight of how Node-Red works in order to make an analysis of the data that the device is sending. It is necessary to have different accounts related to IBM to make the whole configuration of the architecture.

7.2.4. thethings.io

thethings.io [https://thethings.io] platform provides a complete backend solution for IoT makers and IoT APP developers through an easy and flexible API. [22] It is specialised in fast and scalable IoT and focuses on device management and data analytics. One of the main benefits of this platform is that any device can be connected with an IP, so it is hardware agnostic, any brand that develops technological device can be connected to the platform.

The connection with the Sigfox device is achieved easily and quickly only by following a clear tutorial and the visualization and analysis can be deployed in the same platform just after the device has been connected. Only by using a parser the data is split in the attributes necessary to make visualization. All the necessary actions, from connection of the device and the platform from visualization and analysis can be done in the same platform. Real time data storage facilities are also provided to have interoperable access of the devices for fast and low cost product development. [22]

thethings.iO has been specially designed for Sigfox connectivity, having developed a Sigfox library: [https://github.com/theThings/thethings.iO-Sigfox-SDK]. [27]

7.2.5. Comparison table of different IoT platforms

	Microsoft Azure	AWS	IBM Watson IoT	thethings.iO
Protocols	HTTP, AMQP, MQTT	HTTP, MQTT, WebSockets	HTTP, MQTT	HTTP, MQTT, CoAP, Websockets
Certified Hardware	Intel, Raspberry Pi2, Freescale, Texas Instruments	Broadcom, Marvell, Renesas, Texas Instruments, Microchip, Intel	ARM mbed, Texas Instruments, Raspberry Pi, Arduino Uno	Hardware agnostic
SDK/ Language	.Net and UWP, Java,C, NodeJS	Java,C, NodeJS	C#, C Python, Java, NodeJS	Python, Node.js MQTT, Node.js HTTP, Node.js CoAP, Jailed Node
Connection	Easy configuration	Easy configuration	Difficult configuration	Easy configuration
Analysis	Medium configuration	Easy configuration	Difficult Configuration	Easy Configuration
Pricing	Paying for IoT Hub unit related to number of devices and messages per days	Paying millions messages traffic	Paying related to number of device, data traffic and data storage.	Paying related to the number of device and the platform maintenance.

Table 1: Comparison of different IoT platforms. Source: [28], [29]

8. Budget

This section explains the budget for the project.

Firstly, the time spent to make this project will be detailed. The hours in configuring the platforms mean the time spent understanding how to make the configuration, the actual configuration and the after analysis of how each platform internally works.

Concept	Hours	Revenue(€/h)	Total price (€)
Project research	300	8	2400
Configuring AWS	20	8	160
Configuring Microsoft Azure	25	8	200
Configuring IBM Watson IoT	30	8	240
Configuring thething.iO	10	8	80
Total	385	8	3080

Table 2: Working hours of the Project. Source: Own

Secondly, the cost of the material used that correspond to the TD1208 EVB and the Sigfox subscription is 162 €.

Finally, all the platforms have been used with the free trial account so they have not charged for sending the messages and storing them. For example, to have an idea of how the services provided by these platforms does cost, the pricing of thethings.io have been explained. If the user pays 29 € per month, he can use up to 10 things, send 50K monthly API calls / messages, have 10K monthly cloud code execution and have 1 admin user. [30]

9. Environmental Impact

The environmental impact of this project is minimal due to the fact that a low-energy consumption device such as the TD1208 EVB and a computer were used. However, some minor downsides have to be taken into account in order to minimize the carbon footprint of a higher deployment of the architecture designed in the project.

The batteries used to power the IoT sensor in a real case are low energy consumption, they would require maintenance in order to change them when the batteries are wasted. Another problem is that the dangerous materials with which the batteries are made should be recycled, which costs money and energy.

One solution to this problem would be the use of energy harvesting systems attached to the sensors and transmitting devices so as to easily power those remote devices using clean energy. Energy harvesting technologies use power generating elements such as solar cells, piezoelectric elements, and thermoelectric elements to convert light, vibration, and heat energy into electricity, then use that electricity efficiently. [31]

The messages sent from the TD1208 device produce radio electric contamination due to the use of Radio Frequency technologies. However, during the execution of this project, the emission of RF waves has been produced inside acceptable parameters, according to the current Spanish legislation on radio electric spectrum.

So far, only the negative impacts have been mentioned, but the potential in optimizing processes derived from the use of the IoT technology and of this project in particular is huge. The IoT technology can help to optimize industrial, agriculture production, hence increasing productivity and reducing the energy consumption of these processes.

The IoT technology can be used to create services that reduce the carbon footprint and other environmental impact. In particular, a possible future development of this project, could mean using the IoT architecture detailed in this project for monitoring temperature to create a service that handles the improvement of the building insulation in the university.

10. Conclusions

The aim of this project has been testing different IoT platforms to see what different approaches they did use. The configuration process aim was to connect the Sigfox Device to each and every of the IoT Platforms, then store the data in a data base and ultimately visualizing the data so as to get useful information.

The contributions made by this project are described below.

Regarding the Sigfox Device, TD 1208 EVB, it has been configured in order to send the data provided by a temperature sensor included in the TD 1208 chip automatically by means of a python programme. Then, the data reaches the Sigfox Cloud, from where it has been delivered to the IoT Platforms.

With respect to the AWS platform, it has been configured to receive the messages that are stored in the Sigfox Cloud. Moreover, it has been configured the DynamoDB service in order to store the data for a long period of time. The data stored in the DynamoDB service has been connected to the Redash visualization platform that has accomplished the visualization of the temperature lectures. Using Boto3 module and by means of a python programme, the consultation of the data stored in the DynamoDB service by external users has been made possible.

Concerning the Azure Microsoft platform, the connection between the Sigfox Cloud and the Azure Platform has been carried through. Then the data has been stored in a Storage Account service. Moreover, a Stream Analytics service has been configured in order to send the data to a visualization service like Power Bi.

With reference to the IBM Watson IoT platform, the connection between the Sigfox Cloud and the platform has been achieved. The messages sent from the Sigfox Cloud reach to the gateway application created with Node-Red and, there, the data is displayed in a table.

In relation to the thethings.iO platform, it has been configured in order to receive the data from the Sigfox Cloud. Then, by means of a parser, the data has been addressed and given an attribute. Finally, a table and a visualization graphic have been created in order to watch the trend of the temperature during a period of time.

All the above mentioned contributions have been explained in detail in this report.

Finally, a comparison of the features of each platform has been made in order to give the reader a broad vision of what platform to choose in function of its needs and intentions regarding an IoT project.

A future line of work of this project is trying to add different type of sensors and make different visualization analysis accordingly, although using the current IoT arquitecture defined in this project as a basis.

Another future line of work is trying to use the existent infrastructure of temperature readings to measure the differences of temperature in the university building and trying to monitor temperature and other variables to improve building insulation.

Other service that would completely fit the service developed with the study of the connection between the Sigfox device, which sends temperature readings, and the IoT platforms is monitoring the conditions of temperature, humidity, atmosphere of an ancient archaeological site. [26]

Acknowledgments

First and foremost, I would like to offer my sincerest gratitude to my advisor Manuel Moreno Eguílaz, first for conceding me the opportunity to do my degree final project about a field that so deeply interests me; and, above all, for the ever constant guidance and counselling throughout this period.

I would also like to thank the team of the IoT platform thethings.iO for giving us access to use his platform during the experimentation process on a free price for more than the 15 days of the free trial period.

Special thanks to César de la Rosa, from AVNET, for the continuous support in the world of IoT with free samples, evaluation boards and help with Telecom Design and Sigfox.

ETSEIB

Bibliography

[1] Israel Terán, Internet de las cosas: una primera aproximación utilizando la tecnología SIGFOX, [http://upcommons.upc.edu/handle/2117/82979]

[2] Albert Torres, Internet de les coses aplicat a la millora del servei de Bicing de Barcelona, [https://upcommons.upc.edu/handle/2117/79589]

[3] Edgar Tamayo, Internet de les coses aplicat a una motocicleta elèctrica, [http://upcommons.upc.edu/handle/2117/84871]

[4] Lluís Casanovas, Desenvolupament d'una GUI en Python per controlar una placa de desenvolupament de d'Internet de les coses, [http://upcommons.upc.edu/handle/2117/110620]

[5] Pol Sarmiento, Development of an Excel VBA library for the Internet of Things, [http://upcommons.upc.edu/handle/2117/110195]

[6] [https://www.ibm.com/developerworks/library/iot-lp201-iot-architectures/index.html]

[7] [http://ignasisayol.com/en/machine-learning-in-a-logistics-company/].

[8] Andrea Zanella ; Nicola Bui ; Angelo Castellani ; Lorenzo Vangelista ; Michele Zorzi, *Internet of Things for Smart Cities*, IEEE Internet of Things Journal (Volume: 1, Issue: 1, Feb. 2014)

[9] [http://www.businessinsider.com/four-elements-driving-iot-2014-10].

[10] [https://iot-analytics.com/top-10-iot-project-application-areas-q3-2016/].

[11] [http://ignasisayol.com/en/the-keys-to-understand-the-forth-industrial-revolution/].

[12] [https://www.ottomotors.com/blog/industry-4-0-connected-factory]

[13] Ramon Sanchez-Iborra ; Maria-Dolores Cano, *State of the Art in LP-WAN Solutions* for Industrial IoT Services

[14] [https://www.sigfox.com/en/sigfox-iot-technology-overview] .

[15] [http://www.radio-electronics.com/info/wireless/sigfox/basics-tutorial.php].

[16] TD1208 EVB User's Guide, [http://rfmodules.td-next.com/modules/td1208/].

[17] TD 1208 Reference Manual, [http://rfmodules.td-next.com/modules/td1208/].

[18] [https://techbeacon.com/4-stages-iot-architecture].

[19] [https://developer.ibm.com/recipes/tutorials/connecting-sigfox-backend-to-watson-iot-platform/].

[20] [https://developer.ibm.com/recipes/tutorials/connect-a-sens039it-2-0-device-to-the-watson-iot-platform/]

[21] [https://csrc.nist.gov/publications/detail/sp/800-145/final]

[22] Partha PratimRay, A survey of IoT cloud platforms; Future Computing and Informatics Journal (Volume 1, Issues 1–2, December 2016, Pages 35-46)

[23] [https://sysadmins.co.za/interfacing-amazon-dynamodb-with-python-using-boto3/].

[24] [https://stephanefrechette.com/upload-multiple-files-recursively-azure-blob-storage-azure-cli-2-0-macoslinux/#.WliBOqjibIV]

[25] [https://github.com/aureleq/sigfox-azure-iothub]

[26] [https://microsoft.github.io/techcasestudies/iot/2016/12/21/PolimiMitreo.html]

[27] [https://partners.sigfox.com/products/thethings.io]

[28] [https://paolopatierno.wordpress.com/2015/10/13/an-iot-platforms-match-microsoft-azure-iot-vs-amazon-aws-iot/]

[29] [https://thethings.io/libraries/]

[30] [https://thethings.io/pricing-iot-dashboard-tool/]

[31] [http://core.spansion.com/article/energy-harvesting-devices-replace-batteries-in-iot-sensors/#.Wl81-KjibIU]