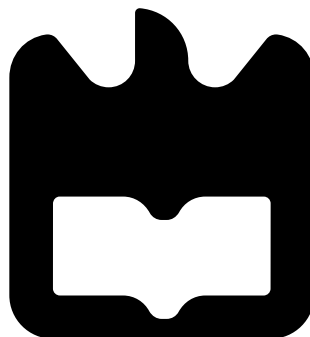




**Tatiana Filipa Gomes
Moura**

**Wireless Communication Support System for
Environmental Monitoring
Sistema de Comunicação sem Fios de Suporte à
Monitorização Ambiental**





**Tatiana Filipa Gomes
Moura**

**Wireless Communication Support System for
Environmental Monitoring
Sistema de Comunicação sem Fios de Suporte à
Monitorização Ambiental**

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Professor Doutor Paulo Monteiro, Professor Associado do Departamento de Electrónica Telecomunicações e Informática da Universidade de Aveiro e co-orientação do Professor Doutor Atílio Gameiro, Professor associado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro.

Este trabalho foi desenvolvido no âmbito do projeto de I&D SOCA – Smart Open Campus, (Ref. CENTRO-01-0145-FEDER-000010.)



Dedico esta dissertação à minha família, especialmente aos meus pais que me apoiaram sempre.

O júri / The jury

Presidente / President

Professor Doutor Arnaldo Silva Rodrigues de Oliveira

Professor Auxiliar da Universidade de Aveiro

Vogais / Examiners Committee

Professor Doutor Fernando José da Silva Velez

Professor Auxiliar, Dep. de Eng^a Electromecânica da Faculdade de Engenharia da Universidade da Beira Interior

Professor Doutor Paulo Miguel Nepomuceno Pereira Monteiro

Professor Associado do Departamento de Eletrónica Telecomunicações e Informática da Universidade de Aveiro

Acknowledgement

A special gratitude to my advisor Professor Paulo Monteiro for supervising the process of my dissertation, I appreciate your patience and guidance. I would also like to thank Dr. Fernando Guiomar, Paulo Santos, and João Prata for helping me during my long and hard journey in this dissertation in many ways.

To all of my family members and friends who supported me during these years in so many ways that I cannot mention all, I sincerely thank and love you!

There were essential experiments that could not have been completed if not for the help of Luís Félix, Kevin Ganhito and Tatiana Moreira, I am sincerely thankful to you all, you helped me in countless ways and made Aveiro a better place to study.

I would also like to thank to my dear friend John for all his patience, for always helping me with my english grammar and for never giving up on me.

Save the Best for last. Save the last for Love. Thank you Fábio, for always believing in me, making me smile and laugh, and keeping me alive.

Palavras Chave

Internet das coisas, Monitorização Ambiental, Controlo Remoto, Sensores, Multi-Tecnologias de Comunicação, Low Power Wide Area Networks, Aquisição de Dados, Qualidade do Ar no Interior

Resumo

A má qualidade do ar no interior das salas de aula pode levar à diminuição do desempenho dos alunos, uma vez que a qualidade do ar é um factor fundamental a ser controlado para garantir a saúde e o conforto dos ocupantes. Esta dissertação tem como objectivo desenvolver um sistema de suporte à monitorização ambiental através de tecnologias de comunicação sem fios e de redes de longo alcance.

O protótipo desenvolvido permite recolher medições contínuas de temperatura, humidade relativa, Compostos Orgânicos Voláteis (VOC), pressão do ar, oxigénio e dióxido de carbono.

Foram realizados testes em salas de aulas seleccionadas durante o semestre de inverno na Universidade de Aveiro usando o protocolo LoRaWAN. É demonstrado como recolher, integrar, analisar e visualizar em tempo real os dados obtidos.

Keywords

Internet of Things, Environmental Monitoring, Remote Control, Sensors, Multi-Technology Communication, Low Power Wide Area Networks, Data Acquisition, Indoor Air Quality

Abstract

Poor indoor air quality in classrooms can lead to decreased students' performance, and affect the health and comfort of the occupants. The purpose of this dissertation is to deploy a system for environmental monitoring support through wireless communications technologies and long range networks. The prototype developed allows to collect continuous measurement of temperature, relative humidity, Volatile Organic Compounds (VOC), air pressure, oxygen and carbon dioxide. Evaluations were done using LoRaWAN protocol in selected classrooms during the winter semester at University of Aveiro. It demonstrates how to collect, integrate, analyse, and visualize real-time air quality data collected.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Objective	1
1.3	Dissertation organization	2
2	State of the Art	3
2.1	Introduction to the Internet of things	3
2.1.1	IoT Architectures	4
2.1.2	Use cases	4
2.1.3	Potentialities	5
2.1.4	Communication Models	5
2.1.4.1	Device-To-Device Communications Model	5
2.1.4.2	Device-To-Cloud Communications Model	6
2.1.4.3	Device-To-Gateway Model	6
2.1.4.4	Back-end data-sharing model	7
2.2	Machine-to-Machine Transport Protocols	8
2.2.1	MQTT	8
2.2.2	LWM2M	9
2.2.3	Overview and comparison between the M2M Protocols	9
2.3	Long-Range Communication Protocols	10
2.3.1	SigFox	10
2.3.2	LoRa and LoRaWAN	10
2.3.2.1	LoRa	10
2.3.2.2	LoRaWAN	11
2.3.2.3	The Things Network	14
2.3.3	LTE (LTE-M and NB-IoT)	15
2.3.4	Overview of IoT Protocols	17
2.4	IoT Cloud Platforms	18
2.4.1	Nokia IMPACT	18
2.5	Node-RED	18
2.6	Data Storage	19
3	System Architecture	21
3.1	Architecture Overview	21

4	Implementation	25
4.1	Development Board	25
4.1.1	Microcontroller	26
4.1.1.1	Pycom FiPy	26
4.1.2	Data Acquisition	27
4.1.2.1	Bosch Sensortec BME680 Shuttle Board	28
4.1.2.2	Grove-Gas Sensor Oxygen (O_2)	30
4.1.2.3	Telaire 6713 - CO_2 Module	30
4.1.2.4	Gravity Analog Infrared CO_2 Sensor	31
4.1.3	Deployed Prototype	32
4.2	System Implementation With LoRaWAN Protocol	33
4.2.1	End Nodes	34
4.2.2	Gateway Connection	34
4.2.3	Server	35
4.2.3.1	Network Server	35
4.2.3.2	Application Server	35
4.2.4	Communication Protocol	38
4.3	Alternative Communication Protocols	43
4.3.1	Usage of the Wi-Fi Communication Protocol	43
4.3.2	Usage of the LTE NB-IoT Communication Protocol	44
5	Testing and Results	45
5.1	Sensors Calibration and Validation	45
5.2	Classrooms' Indoor Air Quality Evaluation	46
6	Conclusion	51
6.1	Future Work	52
A	Application Server	59
B	Registering the Gateway in TTN	61
C	User Interface	63

List of Figures

2.1	IoT solution Architecture [1].	4
2.2	Example of a device-to-device communications model [2]	6
2.3	Example of an Device-To-Cloud communications model [2]	6
2.4	Example of a Device-To-Gateway communications model [2]	7
2.5	Example of a back-end data-sharing model [2]	7
2.6	MQTT Publish/Subscribe Architecture	8
2.7	LoRaWAN network topology architecture [3]	11
2.8	LoRa protocol stack [3]	12
2.9	Packet structure of the packets exchanged on a join procedure using OTAA.	13
2.10	LoRaWAN network activated by ABP.	14
2.11	The Things Network architecture [4].	14
2.12	NB-IoT operation modes [5].	16
2.13	IoT low power wide area access technology landscape [6]	17
3.1	Typical system architecture of a LPWAN.	21
3.2	Use case diagram of the system deployed.	22
3.3	End Nodes state machine.	23
3.4	Gateway state machine.	23
3.5	Server state machine.	23
4.1	FiPy Front [7].	26
4.2	FiPy Back [7].	26
4.3	LoRa and Sigfox Antenna.	27
4.4	LTE Antenna.	27
4.5	Bosch Sensortec BME680 Shuttle Board [8].	28
4.6	BSEC air quality index [8].	29
4.7	Grove-Gas Sensor Oxygen [9].	30
4.8	Telaire 6713 Sensor [10].	31
4.9	Gravity Infrared CO ₂ Sensor V1.0 [11].	31
4.10	First deployed prototype.	32
4.11	Final deployed prototype.	33
4.12	LoRaWAN architecture overview.	33
4.13	Gateway prototype.	34
4.14	TTN Structure.	35
4.15	Node-Red Flow.	36
4.16	Example <i>'storeData'</i> document.	37

4.17	Communication between the end nodes and the gateway.	38
4.18	Keepalive signal between the gateway and the server.	39
4.19	Communication between the gateway and the server.	39
4.20	Packets received at the gateway.	41
4.21	Fragment of the packet details at the gateway.	41
4.22	Packets received at the Application Server.	42
4.23	Packet details at the Application Server.	42
4.24	MQTT Architecture overview.	43
4.25	LTE NB-IoT architecture overview.	44
5.1	Prototype used on tests.	45
5.2	Final prototype.	46
A.1	Application configuration example.	59
A.2	Device configuration example.	60
B.1	Gateway registration configuration example.	61
B.2	Gateway Connected.	62
C.1	Temperature, Humidity, Pressure and VOCs visualisation.	63
C.2	Air quality index, Oxygen and both CO_2 sensors visualisation.	64

List of Tables

2.1	LoRaWAN Regional Parameters for Portugal [12]	12
2.2	Overview LTE-M and NB-IoT [13].	16
4.1	FiPy main features [14].	27
4.2	BME680 specifications [15].	29
4.3	Gas Sensor Oxygen specifications [9].	30
4.4	Telaire 6713 specifications [10].	31
4.5	Gravity Analog Infrared specifications [11].	32
4.6	Description of each field from the packet received and associated metadata.	40
5.1	The maximum outdoor values measured in Aveiro on November 20 th [16].	47
5.2	Measurements from one day of lessons.	48
5.3	Average measurements from one day of lessons.	49

Glossary

I²C	Inter-Integrated Circuit
3GPP	3rd Generation Partnership Project
4G	4th Generation
5G	5th Generation
ABP	Activation By Personalization
AES	Advanced Encryption Standard
API	Application Programming Interface
AppKey	Application Key
AppSKey	Application Session Key
BSEC	Bosch Software Environmental Cluster
BSON	Binary Script Object Notation
CMAC	Cipher-Based Message Authentication Code
CoAp	Constrained Application Protocol
CPU	Central Processing Unit
CSS	Chirp Spread Spectrum
EC-GSM	Extended Coverage - Global System for Mobile Communications
GSM	Global System for Mobile Communications
hPa	hectopascal
IAB	Internet Architecture Board
IAQ	Indoor Air Quality
IMPACT	Intelligent Management Platform for All Connected Things
IoT	Internet of things

IP	Internet Protocol
JSON	JavaScript Object Notation
LoRa	Long Range
LoRaWAN	Long Range Wide Area Network
LPWAN	Low Power Wide Area Network
LTE	Long Term Evolution
LTE-M	Long Term Evolution-M
LTS	Long Term Support
LWM2M	Lightweight Machine-to-Machine
M2M	Machine-to-Machine
MAC	Medium Access Control
MIC	Message Integrity Code
MQTT	Message Queue Telemetry Port
NB-IoT	Narrowband IoT
NFC	Near-Field Communication
NIDD	Non IP Data Delivery
NoSQL	Not Only Structured Query Language
NwkSKey	Network Session Key
OAI	OpenAirInterface
OMA	Open Mobile Alliance
OSI	Open System Interconnection
OTAA	Over The Air Activation
PCB	Printed Circuit Board
PPM	Parts Per Million
QoS	Quality of Services
RF	Radio Frequency
SDKs	Software Developer Kits
SIM	Subscriber Identity Module
SMS	Short Message Service

SoC	System on a Chip
SOCA	Smart Open Campus
SPI	Serial Peripheral Interface
SQL	Structured Query Language
SSL	Secure Socket Layer
TCP	Transmission Control Protocol
TTN	The Things Network
UART	Universal asynchronous receiver/transmitter
UDP	User Datagram Protocol
UWB	Ultrawideband
VOC	Volatile Organic Compounds
VOLTE	Voice Over Long Term Evolution
Wi-Fi	Wireless Network

Introduction

1.1 Motivation

Smart Open Campus (SOCA) aims to build and bring smart intelligence into the University of Aveiro Campus. The main concept of smart campus is to develop the campus through efficient use of resources by applying several types of intelligence in order to provide high quality of living based on an open innovation ecosystem where data is gathered from multiple sources, processed, integrated, and made available for applications and users [17].

Based on the utilization of an appropriate set of technologies, the aim is to develop a digital campus with the future communications, for example as with the 5th Generation (5G) of wireless networks currently on research. At the application level, it considers the well-being of individuals and classrooms. The idea is to support different users (students, employees and visitors) and improve campus performance, graduate quality and also adapt and facilitate all stakeholders' environment; from learning, teaching, management and service.

It was proposed to implement a testbed in order to study how the air quality of a classroom can affect the productivity of a lesson, since air is vital and can affect health and performance. By so doing, we will be able to guarantee air quality, through identifying and reducing the decline of ambient air quality in indoor environments such as classrooms.

In a nutshell, the motivation of this dissertation is to build an Internet of things (IoT) system where users can analyse in real time, air quality in an enclosed space and make the air cleaner.

1.2 Objective

The purpose of this dissertation is to implement an experimental wireless communication support system for environmental data gathering on the University of Aveiro Campus with a remote access. Consequently, this dissertation has the following objectives:

- Research of emerging technologies related with IoT applications;
- Study the rise of Low Power Wide Area Network (LPWAN) technologies, in particular Long Range (LoRa) technology and 5G networks and their communications support between M2M and IoT devices;

- Research about the standardization techniques to improve traffic performance in IoT scenarios and management platforms to get a large number of IoT devices together;
- Research about the environmental sensors in the market;
- Design the overall elements of the architecture system for environmental data gathering in classrooms, from the sensory elements to the server;
- Deploy and implement a prototype capable of a data acquisition protocol that can cope with distinct technology requirements;
- Evaluate the functionality and overall performance of the developed solution in both real and laboratory environments.

1.3 Dissertation organization

This dissertation is divided into the following five chapters:

- **Chapter 1 - Introduction** - Presents the motivation and the main objectives behind this dissertation.
- **Chapter 2 - State of the Art** - Discusses the existing technologies related to smart environments and theoretical background.
- **Chapter 3 - System Architecture** - Represents the proposed system architecture, including a detailed explanation of communications protocol used and data acquisition process.
- **Chapter 4 - Implementation** - Describes the several steps to achieve the final solution of the proposed system.
- **Chapter 5 - Testing and Results** - Contains a set of experimental results of the evaluation tests done.
- **Chapter 6 - Conclusion** - Summarizes the challenges encountered and presents some concluding remarks with directions for further research.

State of the Art

2.1 Introduction to the Internet of things

IoT has brought us into a new era, as a modern topic, commonly discussed nowadays. The term IoT was used for the first time by a British technology pioneer called Kevin Ashton in 1999. According to him, IoT defines a system in which objects from the physical world could be connected to the Internet via a sensor [2].

Considering a standard concept, there is no single universally accepted definition for the term IoT, there is a variety of them. However, all of them describe scenarios in which there is a network protocol and computing capability of smart objects, sensors, devices and machines. Such devices are not considered to be computers. All of these sensors and devices require human intervention to be monitored, analysed and controlled as they are often connected to remote data storages.

2.1.1 IoT Architectures

A typical IoT solution has four stages, as presented in figure 2.1. The first stage is characterized by many devices, wireless sensors that gather data. In the second stage, that data may either be sent to directly the cloud or a gateway can be used to aggregate data from multiple sources before forwarding. In the third stage data will reach the cloud, which handles the processing and storage. In the fourth stage, the information acquired through processing is ready and may be used in various applications.

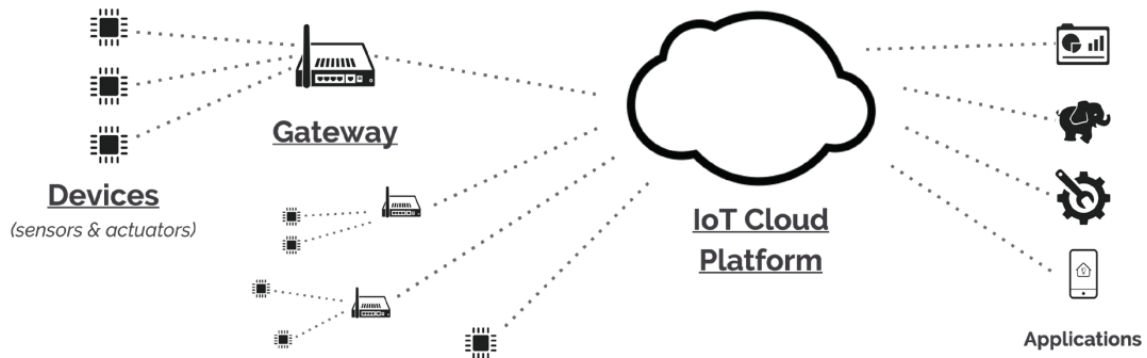


Figure 2.1: IoT solution Architecture [1].

2.1.2 Use cases

Interconnecting smart objects with the Internet enables exciting new use cases and consequently new products in the market. Different types of IoT applications and use cases have been developed by many companies, organizations and associations that could extend to nearly every aspect of our lives.

Some settings for IoT applications [18, 19] are:

- **Healthcare** - IoT devices can be used to monitor and preserve human health and wellness, disease management, remote monitoring of patients' health statistics and send all information to the nurses and doctors. As an example, a small wearable device can detect a person's vital signs and send an alert to a health care professional when a certain threshold has been reached or when a person has fallen down and cannot get up.
- **Home, offices and lodging** - IoT devices can be utilized to control connected devices like TVs, refrigerators, lights, thermostats, smoke detectors and other sensors, security systems, water quality monitoring and much more. At home, for example, embedded sensors can understand the human activities and thus adjust air temperature, humidity or lighting to reduce the waste of energy without compromising comfort.
- **Retailers and services provided** - It can be applied in improving stores, banks and restaurants operations, reducing theft, optimizing inventory management and enhancing

the consumer's satisfaction monitoring. The main goal is to regain lost market against online competitors and attract consumers into the store.

- **Logistics and fleet management** - IoT devices have multiple applications in geolocation tracking, on-board diagnostic or smart labels.
- **Industry** - IoT devices can assess repetitive work operations, such as production line remote monitoring, predictive maintenance or workforce tracking.
- **Agriculture** - Usage of devices that can be applied to climate monitoring and forecasting, livestock management, soil moisture levels to control irrigation systems and minimize water consumption, and wind monitoring.
- **Vehicles** - IoT can be used to connect all electronic systems present in vehicles, from engine diagnostics to GPS data, smart driving assistance that can reduce the vehicle collision rate and cloud-based infotainment solutions.
- **Cities** - Can be used in smart lighting grids automation, more efficient transportation and parking systems, improvement of traffic control, smart meter and bus networks, and efficient public services tracking.

2.1.3 Potentialities

The world is on the edge of revolutions in both quantity and quality, the power of IoT is improving the comfort level of humans' life by providing connectivity between objects with human beings [20, 21]. Nowadays, can be perceive that more things than humans are connected to the internet [22]. Another potential use of it, but with some drawbacks to human beings is all the data that can be collect from the world. The internet will cross all aspects of people's lives, and will allow information systems to capture their own personal information by themselves, for themselves, in a large array of connected sensors that will turn the world into data, and all of data captured can help automate processes and tasks for people.

2.1.4 Communication Models

In a practical context it is important to be aware of how IoT devices connect and communicate with each other. The Internet Architecture Board (IAB) published a document, in March 2015 for networking of smart objects (RFC 7452)[23], where it summarizes four common communication models used in the smart objects' environment, so they can connect and provide value to the user.

2.1.4.1 Device-To-Device Communications Model

Device-to-device communications model represents two or more devices which are connected and can communicate directly among themselves through shared network. These kind of devices can communicate over different networks, Internet Protocol (IP) networks or Ethernet, however they use protocols like Bluetooth, Long Term Evolution (LTE), ZigBee, Near-Field Communication (NFC), Ultrawideband (UWB), or Z-Wave that enables direct communication between them, as the example shown in Figure 2.2.

Although, interoperability between two devices manufactures by different sellers will need development efforts to implement the specific device formats instead of open approaches

standard data formats. Another consideration to take are the possible incompatibilities in communications protocols between device-to-device.



Figure 2.2: Example of a device-to-device communications model [2]

2.1.4.2 Device-To-Cloud Communications Model

Device-to-cloud communications model represents devices connected straight to cloud service. In establishing a connection between the device and the cloud service using IP network, the traditional communication mechanisms like Ethernet or Wireless Network (Wi-Fi) connections will be used, as the example in figure 2.3 shows. Moreover, standard Internet protocols are needed to communicate with the server infrastructure, such as Constrained Application Protocol (CoAp), User Datagram Protocol (UDP), IP, among others.

This communication model presents an efficient solution for IoT services and applications through managing data, dealing with the real world things in a more distributed and dynamic way.

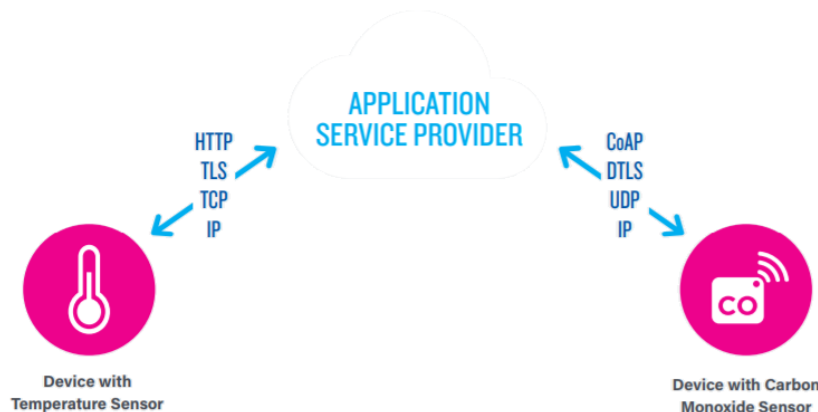


Figure 2.3: Example of an Device-To-Cloud communications model [2]

2.1.4.3 Device-To-Gateway Model

Device-to-gateway model, as the example in figure 2.4 shows, this model is convenient for IoT devices where they will be connected to an intermediary local application gateway, which is used to process data gathered by IoT devices before sending them to the Cloud that provides security and other functionality such as data or protocol translation.

This communications model is frequently used to integrate new smart devices into legacy systems with devices that are not natively interoperable with them.[2]

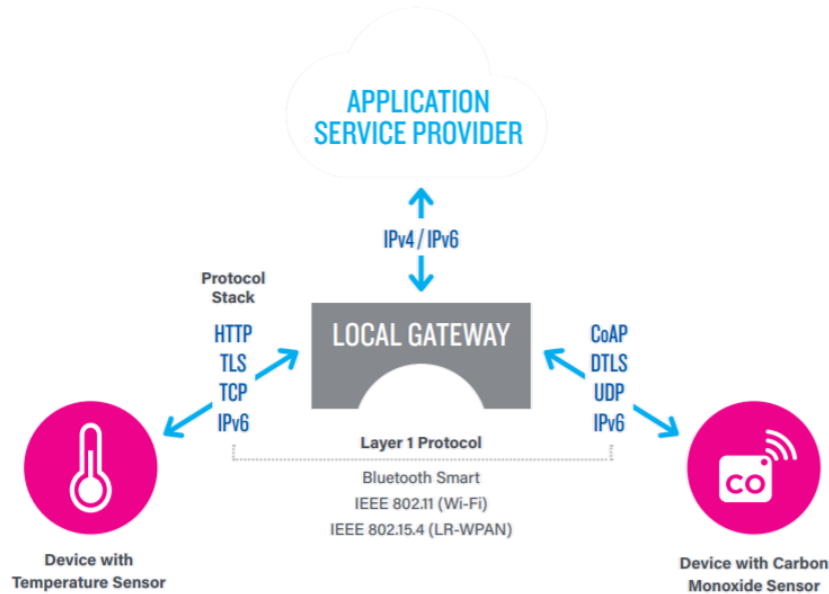


Figure 2.4: Example of a Device-To-Gateway communications model [2]

2.1.4.4 Back-end data-sharing model

Back-end data-sharing model refers to a communication architecture that enables IoT devices to transfer data to a cloud service or other destinations. A third party is required to access the uploaded sensor data. An example of it can be seen in figure 2.5.

This kind of model allows data collected from simple IoT devices to be uploaded to a single application service provider, where it is aggregated and can be analysed.

In interoperability back-end data-sharing architectures cannot cope with closed system designs.

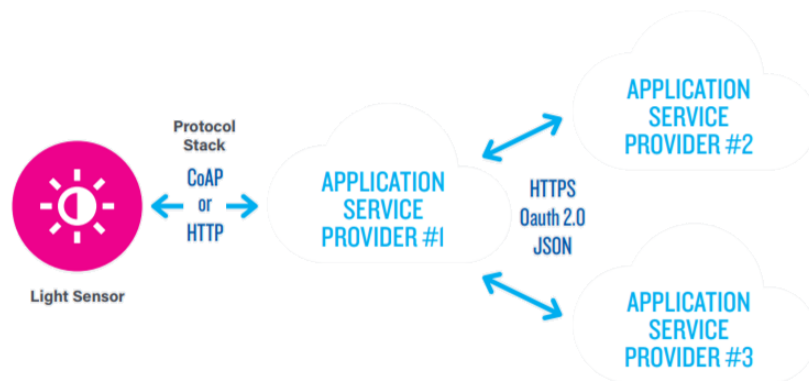


Figure 2.5: Example of a back-end data-sharing model [2]

2.2 Machine-to-Machine Transport Protocols

With IoT, everything around us is becoming connected and more interacting. The real question is what communication protocols seamlessly integrate thousands of nodes and enable data transfer process.

Machine-to-Machine (M2M) Transport Protocols are application layer protocols that facilitate the transfer of messages between two devices. It is basically a communication process of connected sensors, which will send real-time data to a remote machine which will store it in a server. Currently, some of the most widely used data transference based protocols are Message Queue Telemetry Port (MQTT) and Lightweight Machine-to-Machine (LWM2M) which focus on distributed messaging in a Publish/Subscription model. These protocols specifically target low power devices which have to conserve power so that they can operate for a long time. A deeper understanding of these protocols and their application requirements is important to properly elect which protocol is most suitable for the application at hand.

2.2.1 MQTT

MQTT is a lightweight protocol based in publish and subscribe messaging architecture, based around a central broker that runs on the top of TCP/IP networking stack [24]. Despite the word *Queue* in the name, MQTT is not a queue based protocol. It has low bandwidth particularly for constrained devices, as sensors and mobile devices on unreliable networks. The TCP/IP ports assigned for MQTT are 1883 without Secure Socket Layer (SSL) and 8883 for MQTT over SSL.

MQTT is divided into four stages: connection; authentication; communication; and termination.

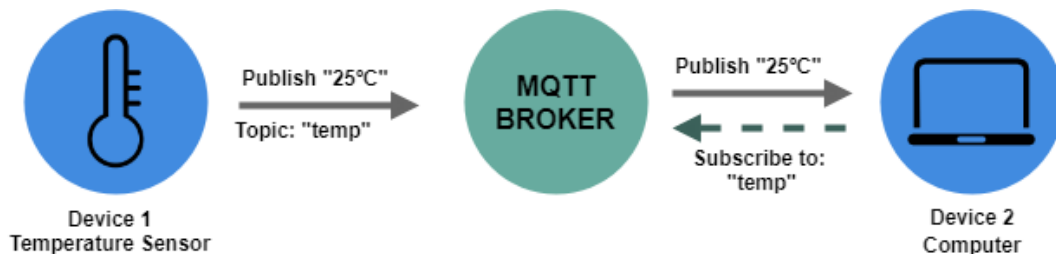


Figure 2.6: MQTT Publish/Subscribe Architecture

As seen in the above figure 2.6, **device 1** starts by creating a connection to the broker over Transmission Control Protocol (TCP), validates and authenticates it. Then the message will be published to an address, known as a topic. The broker forwards the message to **device 2**, the client, and after receiving it, will subscribe the message.

At the center there is always a **broker** which is responsible for filtering and transferring messages over the network, at the same time as deciding who is interested in the messages and then publishing it to all subscribed clients. There are several brokers that can be used, among the ones available online, such as Mosquitto [25] implemented by the Eclipse Foundation and also open-source.

The **client** is any devices or system that sends and receives the actual data. Therefore it could be a Raspberry Pi, Arduino, web browser or any server that can store the data into a

database for future analysis.

All MQTT messages have a Quality of Services (QoS) level, that supports three types:

- **QoS 0** - "*At most one*", when there's no particular guarantees that the message will be receive.
- **QoS 1** - "*At least one*", when it is ensured that the message survives a lost connection, so there is guaranteed transfer of at least one copy of the message, although multiple copies might be received.
- **QoS 2** - "*Exactly one*", when the message was stored and survived a connection loss, and it arrives to the destination.

2.2.2 LWM2M

LWM2M is a standard defined by Open Mobile Alliance (OMA) [26] constrained to IoT devices and adequate for low powered battery devices thanks to low client footprint. The aim is to address service and management needs for constrained M2M devices such as remote device actions, firmware updates, access controls, software management and connectivity monitoring.

The architecture defines a LWM2M Client and Server. The LWM2M Server is usually placed in a private or public data center and can be hosted by the M2M service provider to manage LWM2M Clients. The LWM2M Client are devices, so they are located on the device.

The communication bounded by Server and client are grouped in four interfaces: Bootstrap Interface; Registration Client Interface; Device Management and Service Enablement Interface; and the Information Reporting Interface. LWM2M protocol stack uses CoAp as the underlying transport protocol over UDP or Short Message Service (SMS) bearers. CoAp defines the message header, request response codes, message options and retransmission mechanisms.

2.2.3 Overview and comparison between the M2M Protocols

MQTT is best used for simple or trial IoT deployments. It is on based subscriber/publisher architecture and provides small network overhead [27]. MQTT has three Quality of Service policies to avoid packet loss, which can be a very important feature for real-world reliable applications. It is designed to be a lightweight protocol, implemented through many unmanaged devices generating high data volumes for IoT platform ingestion. However, MQTT is based on TCP/SSL which is not optimized for constrained devices and does increase the client-side computational requirements. MQTT's protocol allows support of data management and device management, although implementation of these features is an entirely platform or provider specific.

LWM2M is best used for complex and long-term IoT deployments and for implementations with managed devices. LWM2M's primary benefit is improving the interoperability between different device manufacturers and cross platforms of data management and devices management capabilities.

Unlike MQTT, LWM2M has a well-defined data and device management structure that enables a variety of standard objects.

While MQTT is the most common protocol used today, LWM2M is growing in the market for more complex and longer-lived deployments with managed devices. Furthermore, MQTT protocol is much easier to set up than LWM2M.

2.3 Long-Range Communication Protocols

Every IoT application has its own specific requirements such as range, data rate, energy consumption and cost effectiveness. The common short-range radio technologies, such as Bluetooth, do not fit scenarios which require long range transmission; on the other hand, the widely used cellular communications can give us a larger transmission, although they consume excess energy from the device.

Consequently, the network known as LPWAN was born, created for M2M and IoT networks, which operate at a lower cost with greater power efficiency than traditional mobile networks and connected devices. Below are presented some of these LPWANs namely: SigFox; LoRa and Long Range Wide Area Network (LoRaWAN); and LTE.

2.3.1 SigFox

SigFox is the pioneer in LPWAN technology [28] proposed in IoT market in 2009 by Christophe Fourtet and Ludovic Le Moan. Concerning the security aspects, there is not much documentation available about its specifications.

It uses Ultra Narrow Band modulation in which signals have a bandwidth of 100Hz. Furthermore it is a lightweight protocol, that supports bidirectional communications and claim to have achieved connection with up to a million devices per gateway and a range of 30 to 50 km in rural areas and 3 to 10 km in urban areas [29].

2.3.2 LoRa and LoRaWAN

LoRa is a proprietary protocol developed by Semtech and the LoRa Alliance [30], LoRaWAN protocol is also trademarked by the LoRa Alliance, and both of them are open source protocols. It can be said that LoRa consists of two parts, LoRaWAN and LoRa modulation. LoRaWAN involves a protocol stack with the LoRa wireless as the physical layer.

Both protocols are non-cellular LPWAN wireless communication network protocols and with low power consumption, with a battery lifetime up to 20 years [3].

2.3.2.1 LoRa

- Stands for the physical layer, layer one of the Open System Interconnection (OSI) model;
- Is based on a Chirp Spread Spectrum (CSS) ¹ modulation. Is a technique that uses wide band linear frequency modulated chirp pulses to encode information;
- As a long range capacity, only one gateway or base station can cover the whole city or hundreds of square kilometers.

¹The CSS was initially developed for radar applications in the 1940's and has since then been used for military and space communication.

2.3.2.2 LoRaWAN

- Is a protocol specification built on top of the LoRa, defines a MAC layer protocol (Media Access Control) with the second layer of OSI model and along some elements of the third layer of OSI.
- Figure 2.7 shows the star network topology of LoRaWAN architecture, where a central server is the root or center of the network. This improves battery lifetime.

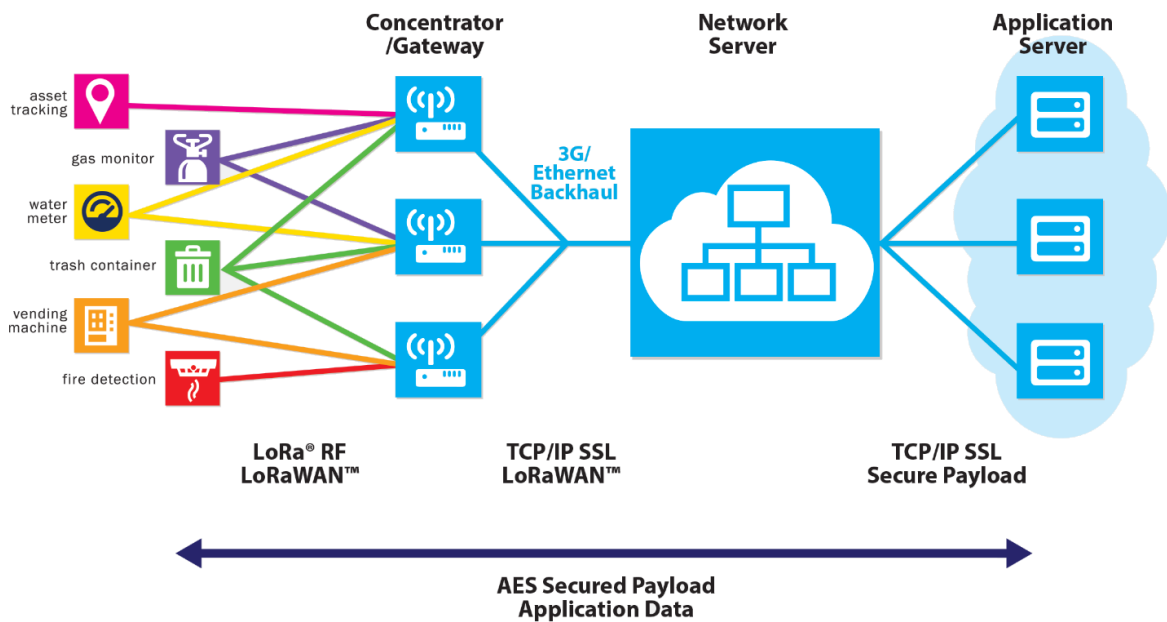


Figure 2.7: LoRaWAN network topology architecture [3]

- An **end node** corresponds to a LoRa embedded sensor, that means they have one or multiple sensors to detect some parameters, such as temperature, with a LoRa transponder chip to transmit the signal over LoRa.
 - First, an end node uses the LoRa Radio Frequency to communicate with one or multiple **gateways**. Each **gateway** will forward then the received packet from the end node to the network server via either cellular, Ethernet, satellite, or Wi-Fi.
 - The **Network Server** manages the network and will decode the packets sent by the end nodes, performing the security checks and then route them to the **Application Servers** [31].
 - The **Application Server** receives data from the **Network Server** and can typically be built over IoT platforms.
- LoRaWAN considers three classes of end nodes. As illustrated in figure 2.8, Medium Access Control (MAC) layer provides classes of communication style: Class A, Class B and Class C [32].

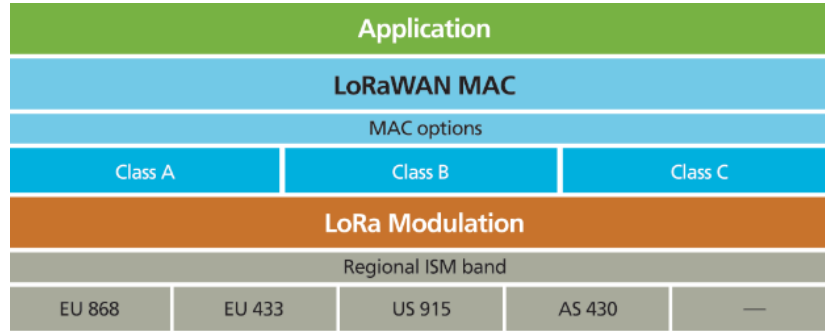


Figure 2.8: LoRa protocol stack [3]

- **Class A:** Is the default class which must be supported by all LoRaWAN devices. Allows bidirectional communication between an end node and the gateway. Each uplink transmission is followed by two downlink receive windows. This is an ALOHA type protocol, where downlink communication must always follow an uplink transmission schedules by the end nodes. In this way, any other downlink communications from the server will need to wait until the next scheduled uplink. The devices are battery powered and sleeping most of the time.
 - **Class B:** Adding up to Class A, in Class B, the end nodes have an additional scheduled downlink window, in order to receive a synchronized time beacon sent by the gateway, allowing the server to know when the end node is listening.
 - **Class C:** Devices in Class C are constantly listening, unless they are transmitting data. Devices should always be connected to the power source or have sufficient power available, since having a constant communication window costs more energy.
- LoRaWAN defines parameters that should be followed for each region in order to ensure interoperability between different LoRaWAN networks. For Portugal, the values in use are shown in table 2.1.

Channel Plan	Bandwidth [kHz]	Channel Frequency [MHz]	Nb Channels	Duty cycle
EU433	125	433.05 - 434.79	3	<1%
EU863-870	125	863 - 870	3	<1%

Table 2.1: LoRaWAN Regional Parameters for Portugal [12]

- Data over LoRaWAN is encrypted twice. It is encrypted by the radio link between the end node and the gateway, and it is encrypted again on the Internet between the gateway and the Application Server. Before a device can communicate on LoRaWAN network it must be activated. Devices have two ways to join the network, over **Over The Air Activation (OTAA)**, or **Activation By Personalization (ABP)**. In the two cases, both the network and the joined device, to establish connection must prove that they have the correct security keys [33]. This procedure is extremely important since LoRa gateways operate over frequency, so they can receive data from any sensor in the neighbourhood.

- In **OTAA** a device starts by performing a join request with the information in figure 2.9 to the network. The '**DevEUI**' (a unique Device Identifier), and the '**AppEUI**' (an Application Identifier) must be configured in the end node along with the **Application Key (AppKey)**. The '**DevNonce**' is a random number generated by the end node to prevent from replay attacks. The Application Server has a root key, only known by the application provider, that generates an '**AppKey**'. This '**AppKey**' is configured on the end node and is used to generate the **Network Session Key (NwkSKey)** key and the **Application Session Key (AppSKey)** for each join procedure.

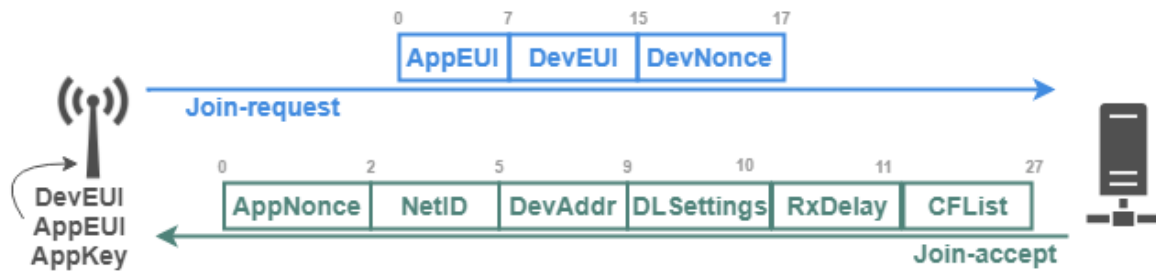


Figure 2.9: Packet structure of the packets exchanged on a join procedure using OTAA.

If the node is accepted to join the network, a join-accept message is sent downlink. This message is sent with a Message Integrity Code (MIC) as well as an Application Nonce. The Application Nonce is used to derive both the Network Session Key and the Application Session Key. The Network ID (NetID) identifies the network; the Device Address is the identifier of the end node within the network; the Downlink Settings (DLSettings) indicates the spreading factor used in the two receiving windows for downlink traffic; RxDelay indicates the delay between the end node transmission and the start of the first receiving window; and the Channel Frequency List (CFList), lists the frequency channels in use by that network, is optional [34].

When the activation is over the air, both the join request and accept messages include a MIC computed using the Cipher-Based Message Authentication Code (CMAC) algorithm with the '**AppKey**', which allows each end to verify that the other end knows the key, thus achieving mutual **authentication** [35]. The **integrity** of a message is also achieved by appending an MIC code at the MAC layer. The MIC is computed using the AES-CMAC algorithm with the '**NwkSKey**' which is used to interact between the end nodes and the LoRa network server in order to prove and verify the integrity and authenticity of the packets. Finally, end-to-end encryption using the Advanced Encryption Standard (AES) is performed for application payloads exchanged between devices and application servers ('**AppSKey**'). This means that the traffic is not only encrypted over the air interface, but it also remains encrypted until the operator's core network. The encryption scheme is based on AES with a symmetric key block cipher which encrypts each message in 128 bit blocks.

- In **ABP** end nodes skip the join procedure, which means the session keys become

static and do not change over different sessions.

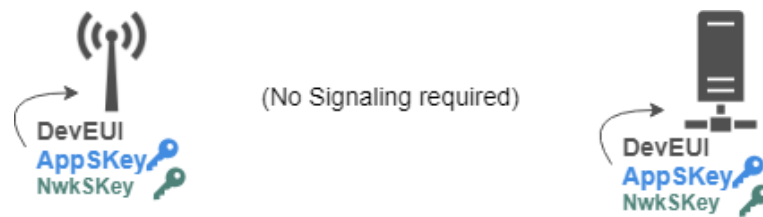


Figure 2.10: LoRaWAN network activated by ABP.

As seen in the figure 2.10, the end node must be configured with the '*DevEUI*' and the two session keys: the '*NwkSKey*' and '*AppSKey*' and is ready for exchanging application data with a specific LoRa network. Both keys are generated by the Application Server.

2.3.2.3 The Things Network

The Things Network (TTN) is an example of a Network Server platform that can be integrated with IoT cloud or platform services. It is a community driven project, so all the features provided are open and free for IoT data network. The initiative was launched in 2015 by Wienke Giezeman and Johan Stokking. At the moment are register around 4,100 LoRaWAN gateways in about 97 countries [36].

Anyone is allowed to build and complete its own IoT solution using the LoRaWAN Application Server and integrate it with existent cloud routing services and storage. It is possible also to register LoRa gateways that will run a TTN packet forwarder to redirect messages from the end nodes that are within its radio range to the Network Server and to the Application Server.

In order to receive messages from end nodes, a member must be registered and then create an application in order to register each device individually. When registering a device, the user selects the type of activation desired, OTAA or ABP, fills the parameters needed for the chosen option and the Application Server generates the keys that must be flashed to a particular device. After it, the user just needed to deploy the end nodes and start the communication between the gateway device and messages will start to appear in the TTN application dashboard.

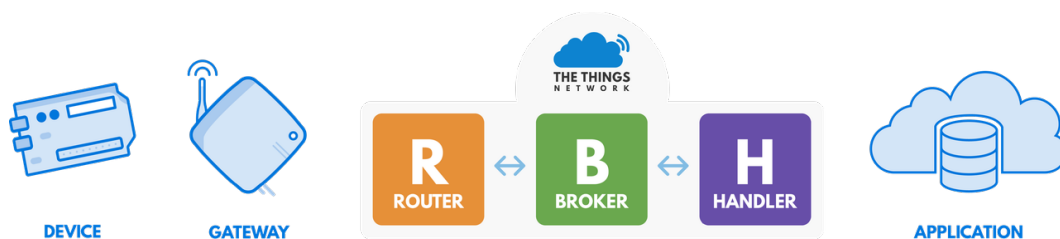


Figure 2.11: The Things Network architecture [4].

The architecture of TTN is showing in figure 2.11. Every gateway is connected to one **Router**, that is responsible for managing the gateway and for scheduling transmissions.

Consequently, when a message reaches a local router, it is redirected to one of the several **Brokers**. **Brokers** are the central part of TTN. Their responsibility is to duplicate the uplink messages and then forward them to the **Handler** where the application is registered. The Broker also forwards the downlink messages to the correct **Router**.

The **Handler** is responsible for encrypts and decrypts the uplink messages. Finally, after being decrypted, the payload and metadata will be published in **JavaScript Object Notation (JSON)** format and handed to the application over a **MQTT** broker. Each JSON packet will contain the information regarding the end node address of the sender packet, payload and frame counter; the time of transmission; frequency channel used; bandwidth; spreading factor; type of modulation; coding rate; airtime; and the gateway details used to transmit the packet. Any application that subscribes to the respective MQTT topic will receive incoming packets. There are multiple options to integrate applications with TTN, ranging from IoT clouds or platforms for possible data storage, analysis, visualisation, advanced device management or other services. In order to easily use TTN Application Programming Interface (API)s, **Software Developer Kits (SDKs)** are available for Go, Java, Python, Node-RED and Node.js.

The TTN website is a repository that includes crowd sourced tutorials in order to help the setup of a gateway and end nodes; documentation explaining the TTN architecture and LoRaWAN; and a forum community where ideas or questions can be discussed.

2.3.3 LTE (LTE-M and NB-IoT)

It has been a while since the first time the LTE cellular system developed by 3rd Generation Partnership Project (3GPP) came online [37]. Right now it has 13 releases and it is said that will deliver the LPWAN component of 5G. Nowadays all cellular devices support LTE, IoT devices support **LTE-M** and **LTE Narrowband IoT (NB-IoT)**. Both of them have been incorporated by 3GPP to LTE specifications to support complete M2M and IoT features, respectively[38]. They are capable of carrying high speed data rates through wireless communication for mobile devices and, similar to other LPWAN networks they have longer battery life.

The 3GPP standardized technologies will have a huge advantage in the market since they are already in place licensed LTE bands infrastructure which will help with a quick and cheap deployment of LPWAN devices.

Long Term Evolution-M (LTE-M) is the M2M version of LTE, is often viewed as the second generation of LTE chips built for IoT applications. It was released in 2016 by United States in order to support complete M2M and IoT features. It has only one in-band operation, it is similar to legacy LTE, so it is compatible with the existing LTE network, LTE stations only need a software update. Which is a great advantage for the operators once they do not need to spend money in building new antennas [39].

LTE-M can support a wide range of applications that need low data rates. It can be used for sensor monitoring in applications like building automation or asset tracking, but it can support a far wider range of IoT applications that need real-time communications (like voice, emergency data and precision tracking data), as well as those that need fixed and mobile communications [40].

NB-IoT, refers to narrowband IoT, is a part of release 13, and was launched on 2017. NB-IoT does not operate in LTE band as LTE-M, meaning that providers will have higher cost to deploy it. However, NB-IoT is still a potentially LPWAN with less expensive requirements

as it eliminates the need of a gateway. With NB-IoT, end nodes will send data directly to the main server [39].

It has been developed to operate in three operations modes: in-band, guard-band, and stand-alone, as show in figure 2.12 [41].

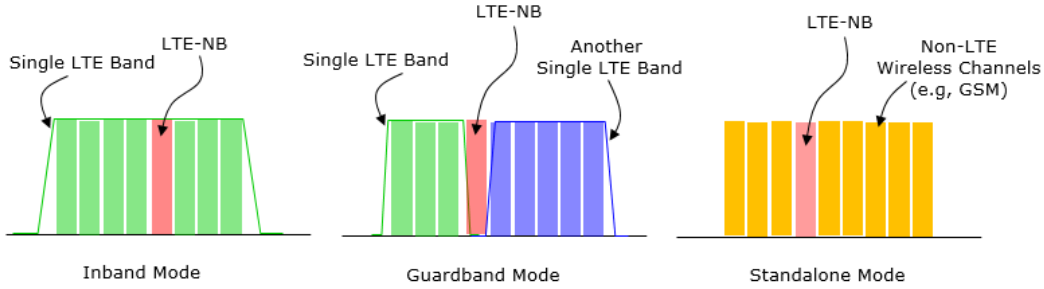


Figure 2.12: NB-IoT operation modes [5].

In in-band or guard-band case, a single NB-IoT carrier occupies the bandwidth of one LTE Physical Resource Block (180kHz). In stand-alone operation, a single NB-IoT carrier is deployed in a bandwidth of 200 kHz.

It is announced for 2020, even though it is far from being standardized. NB-IoT is expected to be more spectrally efficiency, support many more users, offer higher data rates, latency lower than 10 seconds and allowing to interconnect billions of objects into a single base station. This standard focuses specifically on higher distance coverage at optimized lower costs, ultra-low complexity, and indoor coverage improvement [41].

Low-cost narrowband modules should be available in a near future. Devices will be able to connect to narrowband networks and communicate using Non IP Data Delivery (NIDD) [42].

The table 2.2 shows an overview between **LTE-M** and **NB-IoT**.

Feature	LTE-M	NB-IoT
Bandwidth	1.4 MHz	200 KHz
Up-link data-rate	1 Mbit/s	250 kbit/s
Down-link data-rate	1 Mbit/s	250 kbit/s
Latency	10-15 milliseconds	1.6-10 seconds
Antennas	1	1
Voice Over Long Term Evolution (VOLTE)	Supports	Does not support

Table 2.2: Overview LTE-M and NB-IoT [13].

The biggest differences are bandwidth and voice support that is just supported by LTE-M. LTE-M has a maximum uplink and downlink data rate of 1 Mb/s while NB-IoT reaches only 250 kb/s. LTE-M has a maximum channel bandwidth of 1.4 MHz, and NB-IoT uses just 200 kHz.

In addition, NB-IoT will consume more power than LTE-M, since LTE-M allows devices to get to sleep faster between transmissions, thus saving power.

The latency aspect also plays a role in deciding whether LPWAN should be used or not. The accepted latency for NB-IoT uplink is 10 seconds, while the average latency for LTE-M is around 10 milliseconds.

In a nutshell, NB-IoT offers low bandwidth data connections at low cost and is currently Europe focused, while LTE-M is optimized for higher bandwidth and mobile connections, including voice [42] and its interest has been particularly strong in the United States.

2.3.4 Overview of IoT Protocols

The figure 2.13 shows an overview of IoT Protocols.







	Short-range 	SIGFOX 	LoRa 	LTE-M Rel. 13 	NB-IoT Rel. 13 	EC-GSM-IoT Rel. 13 
Range* MCL**	10cm to 200m	<12km 160 dB	<15km 155 dB	<100km 156 dB	<35km 164 dB	<35km 164 dB
Spectrum	Unlicensed	Unlicensed	Unlicensed < 1GHz	Licensed LTE bands in-band	Licensed LTE in-band guard-band stand-alone	Licensed GSM bands
Bandwidth	2.4 GHz	900MHz 100Hz	900MHz <500kHz	1.08 MHz (1.4 MHz carrier bw)	180 kHz (200 kHz carrier bw)	200 kHz
Max Data rate***	<100s Mbps	<100 bps	<50 kbps (DL/UL)	<1 Mbps (DL/UL)	<170 kbps (DL) <250 kbps (UL)	<140 kbps (DL/UL)

Figure 2.13: IoT low power wide area access technology landscape [6]

LPWANs are distinct from the short-range technology such as Wi-Fi, Zigbee, Bluetooth and WAVE, though they can be used for IoT applications, they are very limited in range and application.

Sigfox and LoRa are two LPWAN technologies using unlicensed spectrum to provide connectivity, with a strong presence in the market and already successfully installed in different countries.

However, these unlicensed spectrum technologies are going to have a very difficult time maintaining viability in the market since mobile operators have a huge advantage. They can reuse the existing mobile infrastructure, being easier and cheaper to develop and deploy IoT applications.

LTE-M, NB-IoT and Extended Coverage - Global System for Mobile Communications (EC-GSM) are 3GPP based cellular LPWAN technologies. EC-GSM-IoT is a 2G technology and likely to be strong in countries that are committed to keeping their 2G networks up for many years and where 4th Generation (4G) coverage is insufficient [43]. Although these technology trials have begun, the commercial timing of EC-GSM is hard to predict and will require a combined ecosystem adoption of the technology [43].

LTE-M allows higher bandwidths, mobility and voice calls. NB-IoT is an answer for applications requiring only limited data connections at low cost and is also the most flexible technology in terms of spectrum usage [44]. EC-GSM can be deployed on existing Global System for Mobile Communications (GSM) networks being a suitable option in the absence of

4G systems [44]. For instance, LTE-M is the most mature in terms of deployment, networks, and in hardware. Additionally, NB-IoT chipsets are still in early stages, with only a few operators testing it [44].

2.4 IoT Cloud Platforms

According to Botta *et al.* [45] having the IoT world and Cloud integrated brought a lot of advantages. Starting with the IoT side, in order to reward their technological constraints, like storage, it can benefit from the virtually unlimited capabilities and resources of the Cloud. On the other hand, the Cloud can extend its scope to deal with real world things.

Basically an IoT Cloud platform can remotely collect data, monitoring and managing all the connected devices from a single system, with unlimited storage. There are already in the market a bunch of IoT cloud providers, they act as an intermediate layer between the IoT devices and the applications.

2.4.1 Nokia IMPACT

Nokia launched in June 2016 an Intelligent Management Platform for All Connected Things (IMPACT), it is one of many software platforms created to make IoT into more than a bunch of sensor networks or automation systems [6].

It is an horizontal IoT platform that provides security to devices, collects and analyses the data from them and manages data gathered from any device, protocol or application. It can support over 1.5 billions device models and recognizes around 80.000 different types of devices [6]. Moreover, supports radio connectivity namely 4G/LTE, NB-IoT, LoRa and Wi-Fi.

IMPACT provides an API layer with an object model capable of extension which allows adding more devices and use cases without programmatic change.

2.5 Node-RED

Node-RED is an open source, flow-based programming tool built for IoT applications, to wire together hardware devices, created by IBM ² in 2013 [46].

A flow-based programming defines an application as networks of black-boxes or nodes, where each node has their own purpose [46]. The developer can connect and configure a few required parameters in the nodes to create an input, output or even just to process data.

In order to access to Node-RED editor a user only requires a web browser, connecting through the standard port 1880. Then to create application flows the user just drags a range of nodes from the pallet into the workspace and wires them together. Application flows are compiled and deployed in a single-click. Its runtime is based on Node.js [47] and the flows are stored using JSON. In addition to the existing nodes, others can be added by installing some packages to the Node-RED running space [46].

Moreover, due to Node-RED's built-in text editor, applications can be made using JavaScript. A dashboard can be also accessed to see some graphics with the processed data.

Node-RED can be run locally, on a device such as Raspberry Pi, or in the cloud.

²IBM's Emerging Technology Services: <https://emerging-technology.co.uk>.

2.6 Data Storage

In order to support all the incoming data from sensors a database is required to store data. There are two major types of databases Structured Query Language (SQL) (relational) and Not Only Structured Query Language (NoSQL) (non-relational). While relational databases use tables and rows made through structured query language, non-relational database contains multiple collections and its own documents format.

Many studies and tests have been done around which database has the better performance, yet the outcome obtained depends on the service requirements. The software architect must carefully weigh the pros and cons in order to choose the database model most suitable.

NoSQL database such as MongoDB, Apache, Redis, CouchDB, among others are simple to use and perceive. Although, Li and Manoharan [48] have tested MongoDB, RavenDB, CouchDB, Cassandra, Hypertable, Couchbase and MS SQL Express databases and proved that not all NoSQL performs are better than SQL databases. The performance varies with each operation. When reading operation requires 10 readings, MongoDB was faster, but when increasing that number to 1000 reading operations Couchbase was the fastest. On average, MongoDB and Couchbase were the fastest ones for the read, write and delete operations.

For instance, van der Veen *et al.* [49] in two proposed scenarios compared an SQL database called PostgreSQL with Cassandra and MongoDB, both of them NoSQL databases, and neither of them was the best. Even so, it is concluded that PostgreSQL was the best in multiple reading operations while MongoDB was the best in single write operations.

Additionally, Parker *et al.* [50] compared NoSQL MongoDB to an SQL database and found that MongoDB has better runtime performance when regarding to simple queries, inserts and updates. However, when it is necessary to update, querying non-key attributes and aggregate queries, SQL was better than MongoDB.

Handle with IoT data is challenging once, IoT environment generates millions of data volume and in different formats. Due to this constant data growth the information storage, support and maintenance have become a challenge while using the traditional data management approaches, such as structural relational databases. A way to manage this is by defining out of the box optimized storage schemas and then to store the data into NoSQL databases while abiding the format principles [51]. Therefore it is important to choose a database with scalability, ability to handle huge amounts of data at adequate speeds, flexible schema and portability with varied analytical tools.

After all, MongoDB was considerate the preferable database. MongoDB is free and open-source database, document-oriented, with high availability and automatic scaling [52]. It was released in 2009 by Kevin P. Ryan, Dwight Merriman and Eliot Horowitz. MongoDB does not have predefined schema per collection or per document and it has not values of types and size per document keys, thus enabling thus a faster development and an easier way of schema modification.

MongoDB database structures data into collections of JSON documents which are represented in a binary-encoded format called Binary Script Object Notation (BSON) for efficiency in encoding and decoding data.

System Architecture

Appropriate ventilation is essential to decrease indoor pollutants and make indoor air quality suitable for human health and comfort. Nowadays, with the availability of micro-controllers, new portable and small sensors, and modelling techniques, the quality of the air inside of a classroom can easily be monitored in real time and with full details.

The objective is to deploy a prototype to collect, integrate, analyse, and visualize real time air quality data from a classroom. For this purpose, we will present an architectural strategy that can be developed within different communication technologies, to add into a platform for environmental data gathering from the sensors to a backed up remote server.

3.1 Architecture Overview

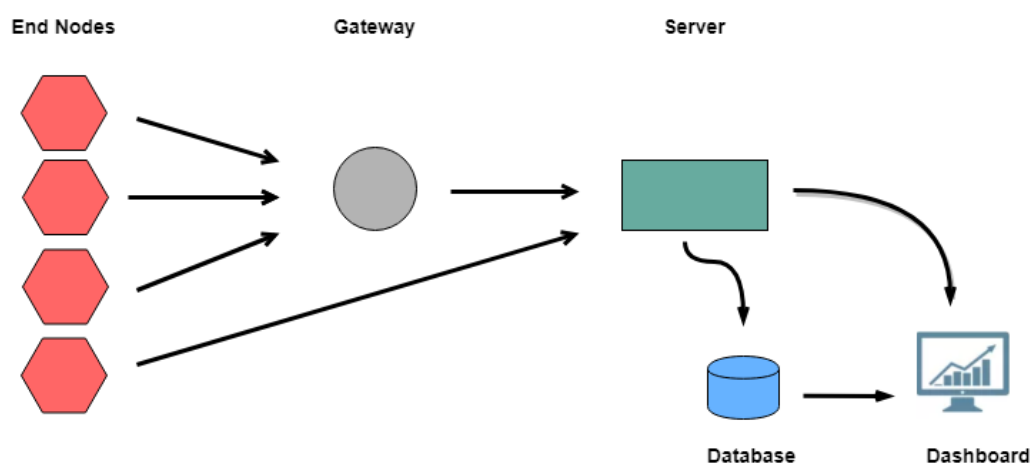


Figure 3.1: Typical system architecture of a LPWAN.

The main components of the system architecture, as shown in figure 3.1, are end nodes with data collecting equipment and monitoring sensors, multi technology communication, Wi-Fi for short range communications and LoRaWAN and LTE NB-IoT as an alternative for long range communications. Data may either be sent directly to the server or a gateway can be used to process data gathered from multiple sources before forwarding. The information will follow into the server which will present a dashboard in real time with the data received as well as store the data into a database.

Figure 3.2 depicts an Use Case diagram with the functionalities that the system architecture must satisfy. There are two actors: the **End Nodes** and the **Actor**. **End Nodes** are devices which can measure real-world conditions and convert these measurements into digital readings. The **Actor** is a person that is responsible for controlling the main activities of the system, *Enable System*, *Disable System* and can control the ventilation at the classroom, *Facilities Control*. The *Enable System* use case enables the actor to activate the system. This use case starts whenever the person responsible decides to enable the prototype and can also end when the person wishes.

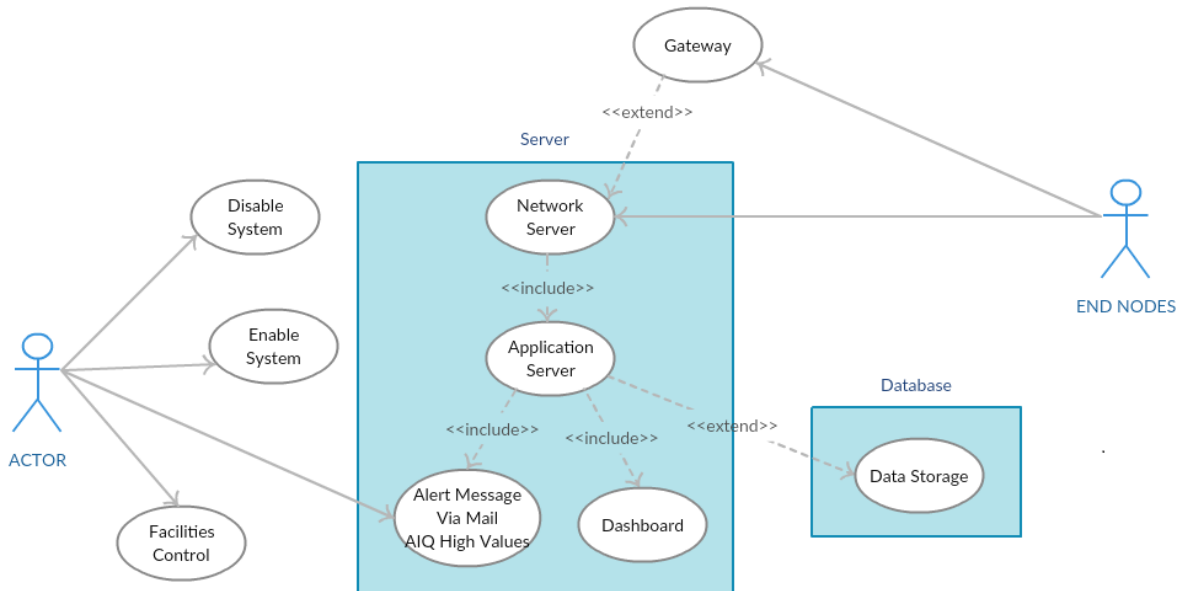


Figure 3.2: Use case diagram of the system deployed.

The system architecture is divided into three state machines, as shown below. To develop a system for environmental monitoring, it all starts with the data acquisition, obtained through the end nodes, as demonstrated in figure 3.3. Every end node must connect over a protocol communication in order to send their data. After connecting to it, a gateway can be used, or not, in order to cover a bigger area and provide additional security to the network. As demonstrated in figure 3.4, a gateway allows an end node to join it and receive their data if both prove to have the same secret keys. Then the gateway will just forward the data to the server, it just needs to connect through a network. Finally, in figure 3.5 the state machine of the server is represented. It is divided into two states: Network Server and Application Server. The gateway connects to the Network Server to forward the data received from the

end nodes, and consequently this data will be exchanged with the Application Server to have a real time visualization and to store it into a database for future analysis.

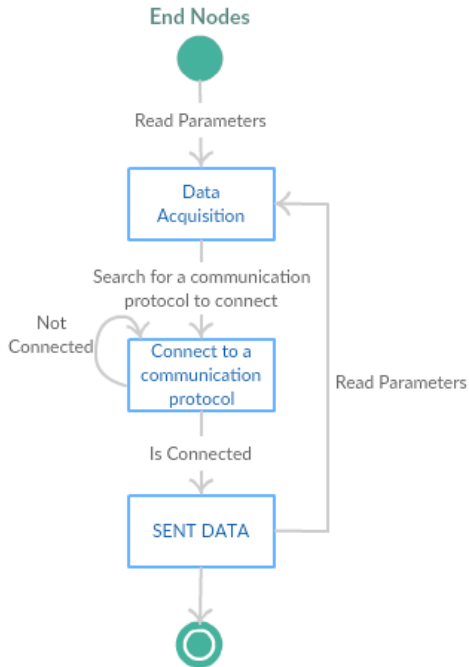


Figure 3.3: End Nodes state machine.

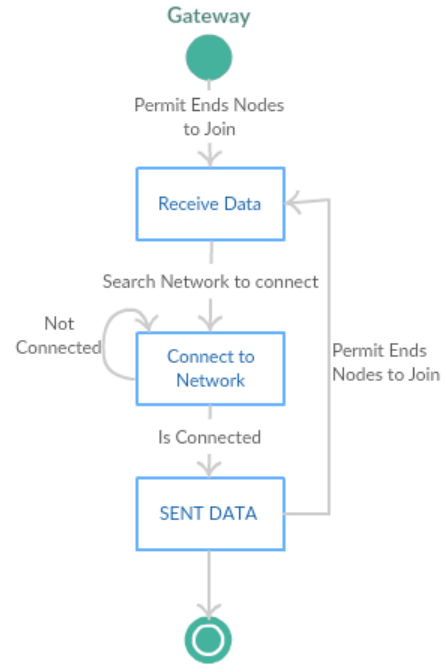


Figure 3.4: Gateway state machine.

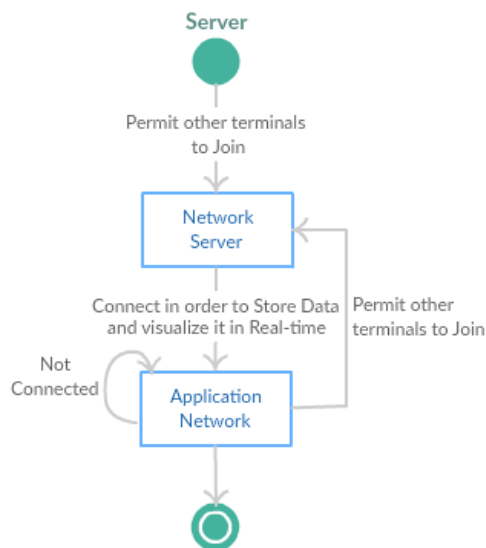


Figure 3.5: Server state machine.

Implementation

At this point, it is important to put into consideration that implementation requires a solid choice for prototyping, developing and testing. Earlier in section 3.1, three different communication protocols were mentioned: Wi-Fi for short range communications, LoRaWAN and LTE NB-IoT as an alternative for long range communications.

At the current level of this research, the Wi-Fi will not encrypt the data and it is also not a requirement yet, because the data does not contain anything sensitive, just the indoor air quality in a room. However, in the future encrypted data could be a requirement.

Additionally, LoRaWAN on one hand, has a much longer range than Wi-Fi, but on the other hand has a decreased signal when passing through buildings or walls. Despite these limitations, in our case building a LoRa network with nodes able to reach the device was achievable within a radius of 600 meters.

Furthermore, Alahi *et al.* [53] concluded that LoRaWAN optimizes the data exchange with the gateway, allowing for lower power consumption when compared to Wi-Fi.

After all these considerations, the architecture over LoRaWAN was the one chosen to be developed and executed in classrooms to monitor their indoor air quality.

Initially, in section 4.1 all the hardware used to implement the prototype will be characterized: sensors used, their characteristics and operating mode. Then, the system implementation is presented in section 4.2, the usage of Wi-Fi in section 4.3.1 and finally the usage of LTE NB-IoT in section 4.3.2.

4.1 Development Board

Connected hardware devices are the heart of IoT. It is important to take into consideration the full stack of hardware, software, and infrastructure.

Starting with the microcontroller that will provide data processing and storage capabilities, followed by sensors and actuators which will be connected to the microcontroller through digital or analog input/output pins or through a hardware bus. And finally, having in consideration portable power sources like a battery since IoT devices rely on power to work.

4.1.1 Microcontroller

Since the aim is monitoring environmental parameters in real-time, it was necessary to get to know better some of the microcontrollers in the market at this moment. The microcontroller will be the core of the communication between the sensor nodes, it will collect the data from the sensor nodes, process them and answer depending on the event type.

A microcontroller is an embedded system. A small, self-contained computer system based on electronic elements. They are, however, less expensive than computers, with lower power consumption and also more flexible in terms of development which makes them easy to handle. They contain in a single integrated circuit one or more computer processors, along with memory and programmable input/output peripherals.

In order to simulate the prototype, the microcontroller used will be a FiPy, from Pycom.

4.1.1.1 Pycom FiPy

The first reason to choose a microcontroller from Pycom was because they run MicroPython, a full Python compiler and runtime that runs on microcontrollers. It allows a much faster and simple development process than the ordinary C language. Another reason to select it was the fact that Pycom has a module that gives access to all of the world's LPWAN networks in one tiny board based on ESP32 System on a Chip (SoC). This module is called FiPy.

FiPy includes 5 networks in one board, Wi-Fi, Bluetooth, LoRa, Sigfox and dual LTE (LTE-M and NB-IoT) [14].

In figures 4.1 and 4.2, the front and back of FiPy are shown.

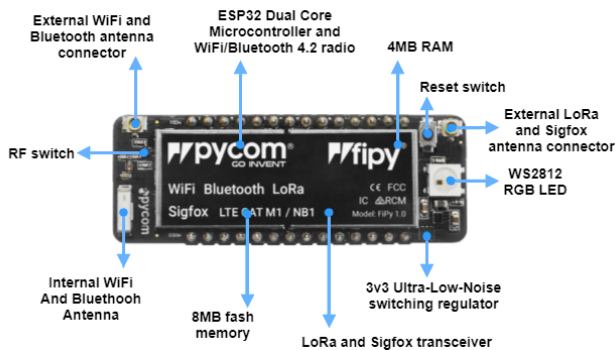


Figure 4.1: FiPy Front [7].

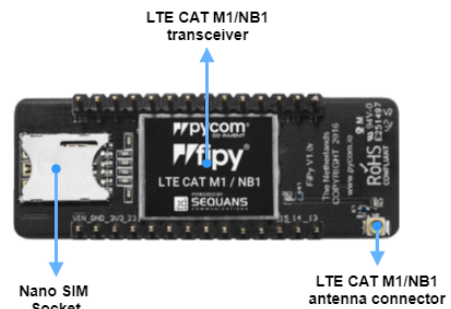


Figure 4.2: FiPy Back [7].

Table 4.1 presents a summary of FiPy features.



Figure 4.3: LoRa and Sigfox Antenna.

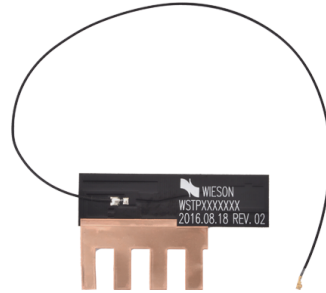


Figure 4.4: LTE Antenna.

Parameter	Technical Data
Memory	RAM: 520KB + 4MB; External flash: 8MB
Voltage Input	3.3V to 5.5V
Voltage Output	3.3V
Communication Interfaces	Inter-Integrated Circuit (I^2C), Serial Peripheral Interface (SPI), UART, micro SD card
Wi-Fi Range	Up to 15mbps
Sigfox Range	Up to 50km
LoRa Range	Node range: Up to 40km; Nano-gateway: Up to 22km
LTE Range	34 bands supports from 699Mhz to 2170Mhz; Supports narrowband LTE UE categories LTE-M and NB-IoT

Table 4.1: FiPy main features [14].

In order to use LoRa or Sigfox connectivity on the FiPy, an external antenna is required, such as the one shown in figure 4.3, otherwise it could damage the device. The same situation applies when using LTE-M and NB-IoT, the LTE antenna, shown in figure 4.4, must always be used with it.

4.1.2 Data Acquisition

Data acquisition is the process of measuring real-world conditions and converting these measurements into digital readings.

Sensors are the input components that allow data acquisition, measuring physical parameters and converting them to electrical signals.

That is why it is crucial to identify from thousands of types of sensors the ideal ones to

measure the desired ambient parameters and consequently it is necessary to study how they operate, their advantages and specifications.

In this prototype, the I^2C standard communication protocol connection intra-device was taken into consideration, this way it will become easier to add components to the bus.

Firstly, only one sensor, the *Bosch Sensortec BME680 Shuttle Board*, was used for tests. This sensor board combines a digital 4-in-1 sensor and was enough to calculate one of the most critical components of enclosed spaces, the indoor air quality in a room.

Posteriorly, three more sensors, a Grove-Gas Sensor Oxygen, a Telaire 6713 CO_2 Module and a Gravity Infrared CO_2 Sensor were added. Since in enclosed rooms with people inside the oxygen levels decrease and the carbon dioxide (CO_2) levels increase, it can be measured by these sensors.

4.1.2.1 Bosch Sensortec BME680 Shuttle Board

The Bosch Sensortec BME680 Shuttle Board (figure 4.5) is a Printed Circuit Board (PCB) with an integrated environmental sensor mounted, with temperature, humidity, pressure, and gas designed for wearable IoT nodes and other battery powered devices.

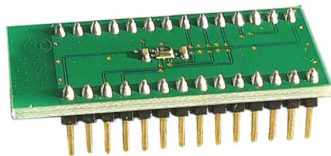


Figure 4.5: Bosch Sensortec BME680 Shuttle Board [8].

The gas sensor detects and measures a range of Volatile Organic Compounds (VOC) carried by nitrogen gas in the surrounding air. However, it can only give a resistance value, with overall VOC content, it cannot differentiate gases or alcohols.

Indoor Air Quality (IAQ) refers to the quality of air inside a closed place. Air Quality Index is an indicator for determining air quality level. It is possible to be measured by the concentration of all the air pollutants, such as VOC gases in this case. There is no official definition or international standard. The common indoor air quality index used in Europe ranges from 0 to 100 between 5 levels. In USA and China, the indoor air quality index has 6 levels and the scale ranges from 0 (clear air) to 500 (polluted air) [54].

The IAQ library from Bosch Software Environmental Cluster (BSEC) uses the raw data of the sensors and outputs an IAQ index shown in figure 4.6.

IAQ Index	Air Quality
0 – 50	good
51 – 100	average
101 – 150	little bad
151 – 200	bad
201 – 300	worse
301 – 500	very bad

Figure 4.6: BSEC air quality index [8].

The air quality index is a linear function of the pollutants' concentrations. To convert from concentration to IAQ index, the following equation 4.1 is used [55].

$$I = \frac{I_{high} - I_{low}}{C_{high} - C_{low}}(C - C_{low}) + I_{low} \quad (4.1)$$

Where:

I = the air quality index,

C = the concentration of the pollutant,

C_{high} = the concentration breakpoint that is greater than or equal to C ,

C_{low} = the concentration breakpoint that is less than or equal to C ,

I_{high} = the IAQ index value corresponding to C_{high} ,

I_{low} = the IAQ index value corresponding to C_{low} .

Table 4.2 lists the specifications of this sensor.

Parameter	Technical data
Communication Interfaces	I ² C and SPI
Supply Voltage Range	1.2V to 3.6V
Current Consumption	2.1 μ A at 1 Hz humidity and temperature; 3.1 μ A at 1 Hz pressure and temperature; 3.7 μ A at 1 Hz humidity, pressure and temperature; 0.09–12 mA for pressure, humidity, temperature and gas
Temperature Range	–40 ⁰ C ~ +85 ⁰ C
Humidity Range	0% to 100%
Pressure Range	300 hPa to 1100 hPa
VOC mixture	Ethane 5ppm; Isoprene 10ppm; Ethanol 10 ppm; Acetone 50 ppm; Carbon Monoxide 15 ppm
Accuracy	$\pm 1.0^{\circ}$ C for temperature; $\pm 3\%$ for humidity; ± 1 hPa for barometric pressure

Table 4.2: BME680 specifications [15].

4.1.2.2 Grove-Gas Sensor Oxygen (O_2)

The Grove-Gas Sensor (figure 4.7) is a device that detects and measures the concentration of oxygen present in air.

Having an oxygen sensor is relevant because depriving the brain of oxygen, exposure to high levels of carbon dioxide, carbon monoxide and other pollutant gases can lead to nausea, poor concentration and fatigue.



Figure 4.7: Grove-Gas Sensor Oxygen [9].

The sensor works on the principle of an electrochemical cell and the output voltage from the sensor is proportional to the concentration of oxygen in the vicinity. This sensor is an analog device that outputs analog signals [9]. Table 4.3 provides the main specifications about the Grove-Gas Sensor Oxygen.

Parameter	Technical data
Communication interfaces	Analog
Power Supply	3.3V to 5V
Measurement Range	0% to 25%
Operating conditions	Temperature: $-20^{\circ}\text{C}\sim+50^{\circ}\text{C}$; Humidity: 0%RH to 99%RH
Accuracy	$\pm 2\%$
Preheat Time	20 minutes
Lifetime	2 years

Table 4.3: Gas Sensor Oxygen specifications [9].

4.1.2.3 Telaire 6713 - CO_2 Module

CO_2 is always present in the air and its concentration can considerably grow in a short period of time due to human activities or just from being in an enclosed space.

Telaire 6713 CO_2 Module (figure 4.8) has a small compact design and uses significant less power compared to other devices available on the market which measure carbon dioxide, although it is as accurate and reliable as the larger sensors. It is a Non-Dispersive Infrared (NDIR) sensor.

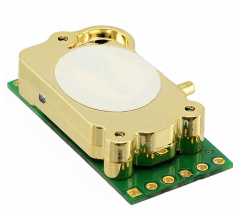


Figure 4.8: Telaire 6713 Sensor [10].

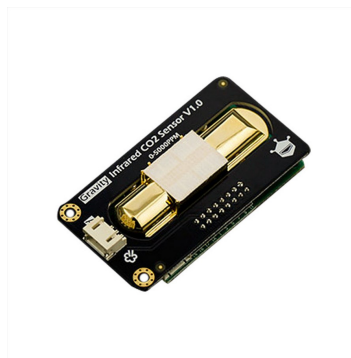
The accuracy measurements of CO_2 come factory calibrated in order to read concentrations up to 5000 ppm. A normal outdoor CO_2 concentration is around 400 ppm, but indoors it can easily grow several times, even with a good ventilation system [56]. Table 4.4 shows the main specifications of the sensor.

Parameter	Technical data
Communication interfaces	I^2C and Universal asynchronous receiver/transmitter (UART)
Power Supply	4.5V to 5.5V
Measurement range	0 to 5000 ppm
Operating conditions	Temperature: $-10^{\circ}C \sim +50^{\circ}C$; Humidity: 0%RH to 95%RH
Accuracy	$\pm 3\%$
Preheat Time	3 minutes;
Lifetime	15 years

Table 4.4: Telaire 6713 specifications [10].

4.1.2.4 Gravity Analog Infrared CO_2 Sensor

Gravity Analog Infrared CO_2 Sensor (figure 4.9) allows measuring the CO_2 air concentration in a range from 0 to 5000 ppm. The sensor operates on *NDIR* technology (non-dispersive infrared) [11], the most common type used to measure CO_2 .

Figure 4.9: Gravity Infrared CO_2 Sensor V1.0 [11].

In comparison to the CO_2 sensor introduced before, Telaire 6713 Sensor, this one is much larger, 37 mm x 69 mm. Table 4.5 shows the main specifications of the sensor.

Parameter	Technical data
Communication interfaces	Analog
Power Supply	4.5V to 5.5V
Measurement range	0 to 5000 ppm
Operating conditions	Temperature: $-10^{\circ}C \sim +60^{\circ}C$; Humidity: 0%RH to 95%RH
Accuracy	$\pm 3\%$
Preheat Time	3 minutes;
Lifetime	5 years

Table 4.5: Gravity Analog Infrared specifications [11].

4.1.3 Deployed Prototype

The first deployed prototype that just contains the *Bosch Sensortec BME680 Shuttle Board* is presented in figure 4.10 and the second one with the rest of the sensors added can be shown in figure 4.11.

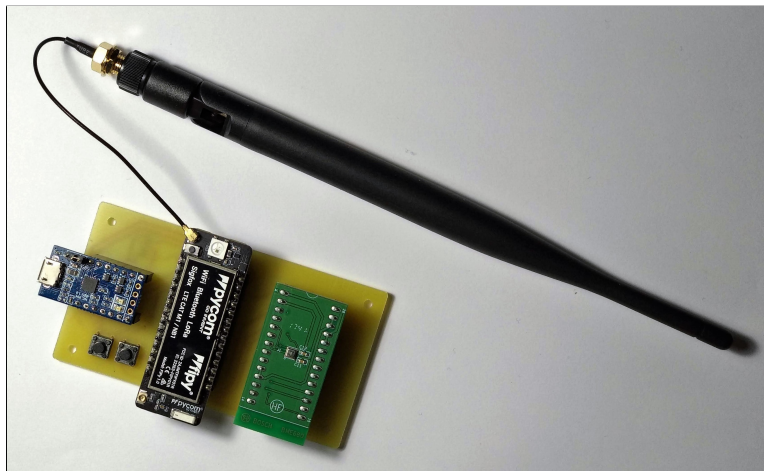


Figure 4.10: First deployed prototype.

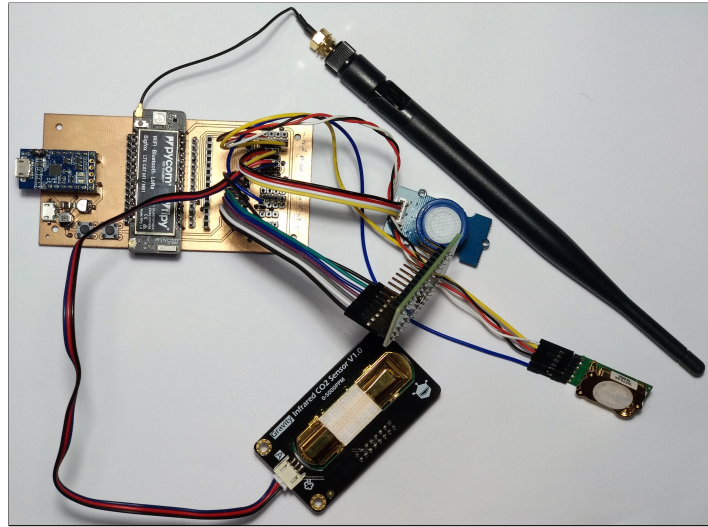


Figure 4.11: Final deployed prototype.

4.2 System Implementation With LoRaWAN Protocol

The prototype developed will work as a LoRaWAN Class C device. It is going to be constantly gathering data and for this reason, while testing it the prototype will be powered with mains electricity using an adapter to connect it to an outlet in the classroom. However this is not the most convenient and practical method since it requires a nearby outlet. Therefore in the future a battery should be added to the prototype.

In figure 4.12 the architecture with LoRaWAN is presented.

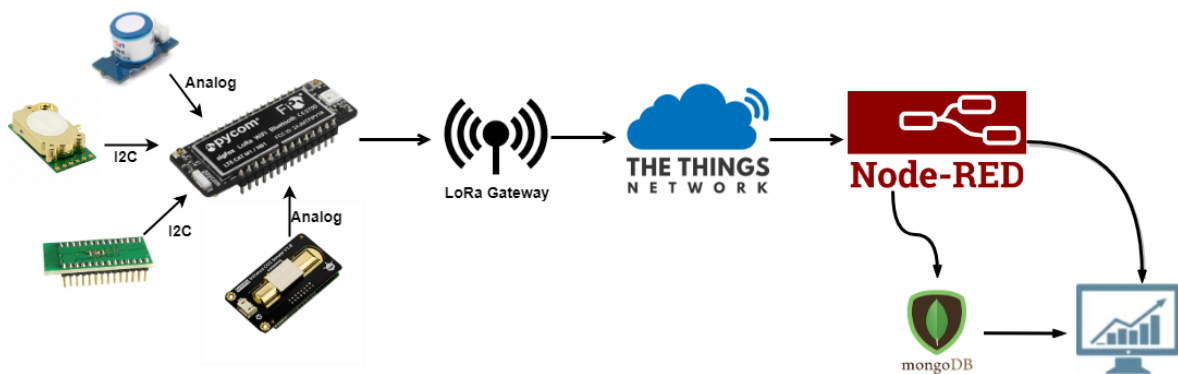


Figure 4.12: LoRaWAN architecture overview.

A desktop computer running Ubuntu 18.04 Long Term Support (LTS) served to start and monitor the LoRaWAN gateway, to run the Node-RED application, to show the graphical user interface and to store data using MongoDB.

4.2.1 End Nodes

In this research, the indoor air quality data was collected by continuously measuring temperature, relative humidity, Volatile Organic Compounds (mixture of gases), air pressure, the index of IAQ, oxygen and carbon dioxide. For this purpose we used a monitoring prototype system developed as mentioned earlier in section 4.1. A Pycom FiPy receives the data from the sensors, also introduced in section 4.1.

In the **Bosch Sensortec BME680 Shuttle Board** the temperature is represented in degrees Celsius, humidity in relative humidity percentage, barometric pressure in the international system units of Pascals, hectopascal (hPa) and gases as a resistance value in ohms. This is later converted to Parts Per Million (PPM). The index of IAQ is determined by the concentration of VOCs, as preciously mentioned. The value for VOCs take up to 20 minutes to stabilize. After it stabilizes, it is used as the baseline reading. If the VOC gas concentration reading increases, the air quality index value increases, and if the VOC gas concentration reading decreases, then the air quality index value will also decrease.

Additionally, the **Grove-Gas Sensor (O_2)** is used to test the oxygen concentration level in the air and returns it as percentage.

Furthermore, the **Telaire 6713 Module** and the **Gravity Analog Infrared Sensor** are both used to measure the CO_2 air concentration in a range from 0 to 5000 ppm. There are two CO_2 sensors. The purpose is to compare both of them and to conclude which one has more accuracy.

Finally all the measurements are encoded and transmitted over LoRa communications.

4.2.2 Gateway Connection

A LoRaWAN network requires at least one gateway, so another Pycom FiPy module was setup to act as gateway, and was used to receive data from the LoRa embedded sensors and then transfer it to the network services applications, TTN (a free LoRaWAN network) via Wi-Fi.

The figure 4.13 demonstrates how the gateway was deployed to transmit via LoRaWAN.

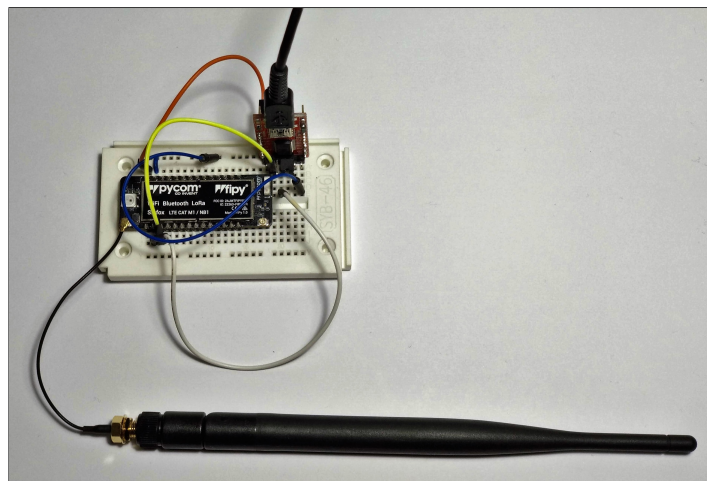


Figure 4.13: Gateway prototype.

In LoRaWAN, gateway devices are always connected to a power source. They connect

to the network server using a standard IP connection and acting as a transparent bridge, converting Radio Frequency (RF) packets to IP packets. On the other hand, end devices make use of LoRa to communicate with the gateway.

The gateway receives the packet and it is forwarded to a TTN application. Then all the data will also be decoded and sent to a Node-Red application. Finally, data will be stored using MongoDB and monitored through a Node-RED dashboard, as explained in the following sections.

4.2.3 Server

The server will be a computer running Ubuntu 18.04 LTS that will interact with the gateway. It comprises the network server and the application server, which in turn includes the dashboard and the database. An internet connection is required.

4.2.3.1 Network Server

TTN, as already described in Chapter 2, is about enabling low power **Devices** to use long range **Gateways** and connect with an open source **Network** to exchange data with **Applications**. The next figure 4.14 describes the structure of TTN.

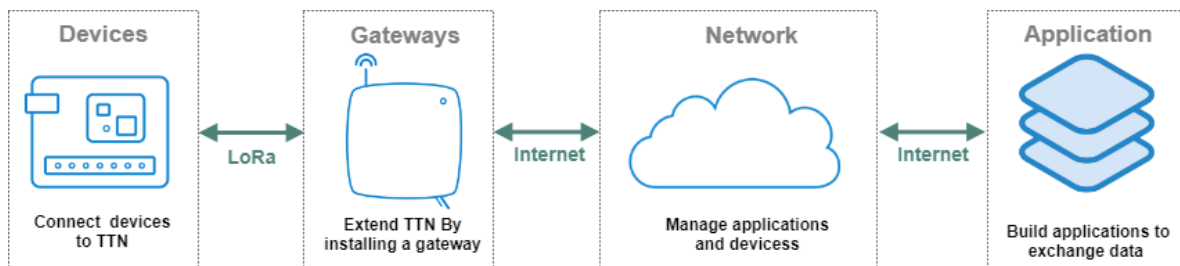


Figure 4.14: TTN Structure.

The fact that TTN is free and open source makes it very favourable for academic purposes and therefore was chosen for the present research. In order to communicate between an end node and the Network server, the device must be registered in TTN.

Following this, an **Application Server** was created in the TTN console and a device was registered, as explained in appendix A.

Furthermore, the **Gateway** was registered in the TTN console, as explained in appendix B, in order to forward the data received from the end node devices to the application.

4.2.3.2 Application Server

The last segment of this implementation is the visualization of the received data in the server. The TTN dashboard does not provide data persistence and only displays incoming packets in real time. All past events are eliminated and thus there is a need of using an available type of integration to store the data for later analysis.

For this purpose, the software programming tool chosen was Node-Red. Data is forwarded from TTN to the Node-RED application over an MQTT broker that subscribes to the specified *Application EUI* and *Device Addresses*.

4.2.3.2.1 Local Node-RED

Node-RED supports a number of extensions, among them an extension for getting data from TTN and a Dashboard user interface. Received packets from TTN not only contain the messages themselves but also metadata, which contains details on the gateway which received the message. In the next figure 4.15 the data flow produced in the Node-RED can be seen.

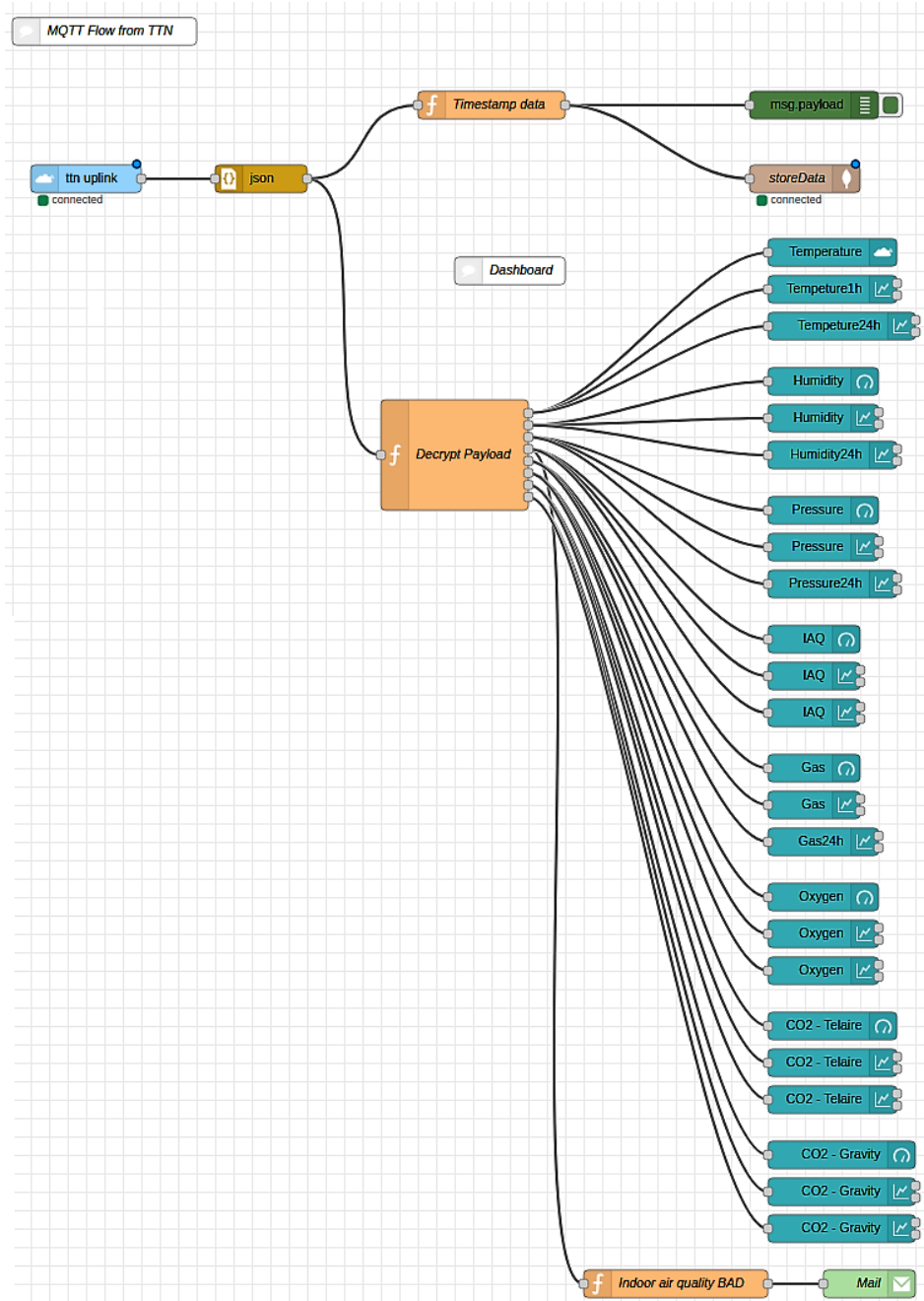


Figure 4.15: Node-Red Flow.

After a received uplink from TTN, the payload is decrypted from *base64* to *ASCII*. After that, an user interface is created in order to follow in real time all the measurements taken, the dashboard presented can be seen in appendix C. Moreover, all payloads received are stored into a database. An alert message is also sent via email when the IAQ index is high, informing that a window should be opened.

4.2.3.2.2 Database Management

The system uses a non-relational database management system based on MongoDB, a document oriented database. NoSQL solutions such as MongoDB have a flexible schema that enables processing and storing different data formats. In this research a structure database was used, titled '*sensors*' with only one collection, '*storeData*' that contains multiple documents.

Each document of the database is a free schema and an object in JSON format, as shown in figure 4.16.

```
{
  "_id": "5bd474c05ebb311bd6212e4c",           \\An unique identifier ObjectId type of the storeData
  "payload": {
    "app_id": "node",                         \\The same as in the topic
    "dev_id": "fipy",                         \\The same as in the topic
    "hardware_serial": "70B3D54992792798",    \\The Device EUI
    "port": 2,                                \\The LoRaWAN port
    "counter": 119,                           \\The LoRaWAN frame counter
    "payload_raw": "MjIuNjYsIDQ5Ljc5LCAxMDEwLjIzLCAwLCAwLjE1", \\The base64 encoded payload
    "payload_fields": {                       \\Object containing the results from the payload functions
      "Date": "2018-10-27T14:22:56.876Z",
      "Sensors": "22.66, 49.79, 1010.23, 1, 0.15"
    },
    "metadata": {
      "time": "2018-10-27T14:22:58.636230918Z", \\When the server received the message
      "frequency": 868.1,                       \\Frequency at which the message was sent
      "modulation": "LORA",                     \\Modulation used in LORA
      "data_rate": "SF7BW125",                 \\Data rate used in LORA modulation
      "airtime": 87296000,                     \\Air time in nanoseconds
      "coding_rate": "4/5",                   \\Coding rate that was used
      "gateways": [
        {
          "gtw_id": "eui-30aea4fffe2d5bfc",    \\The Device EUI of the gateway
          "timestamp": 494910804,              \\The gateway received the message
          "time": "2018-10-27T14:22:58.58841Z", \\The gateway received the message - left out when
          \\gateway does not have synchronized time
          "channel": 0,                         \\Where the gateway received the message
          "rssi": -46,                          \\Signal strength of the received message
          "snr": 6,                             \\Signal to noise ratio of the received message
          "rf_chain": 0                          \\Where the gateway received the message
        }
      ]
    }
  }
}
```

Figure 4.16: Example '*storeData*' document.

All the documents have a unique `_id` automatically generated and indexed by MongoDB. MongoDB guarantees it is unique by returning an `ObjectId` value, which contains a 12-byte BSON combination, where a 4-byte value represents seconds since the Unix epoch, a 5-byte random value, and a 3-byte counter starting with a random value [52].

The 'sensors' database was designed and structured to achieve optimal results by providing fundamental information without unnecessary additional queries or preventing server overload through conditional expressions and search algorithms in arrays. It also provides a fast response to a query.

4.2.4 Communication Protocol

Once everything is set up properly, the gateway is connected and running, and the end node plugged into the electricity and has also started transmitting, the gateway will start to receive packets. For this to happen, an Authentication By Personalisation (section 2.3.2.2) must be created both devices, the end node and the gateway. If the end node does not have the same authentication keys as the gateway, the latter will drop all packets received from the former.

The end node and the gateway communicate through a nonblocking socket, which means that if the gateway is not available to process an incoming packet, it is discarded. This reduces gateway load, allowing it to process real-time information instead of outdated packets. The gateway never sends packets to an end node. The communication between end nodes and gateway is shown in figure 4.17.

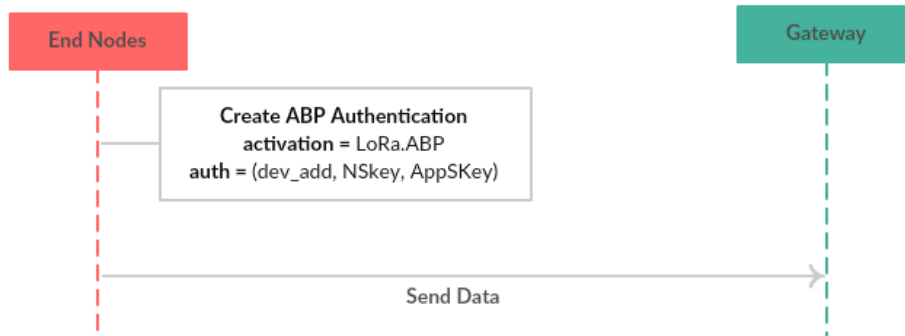


Figure 4.17: Communication between the end nodes and the gateway.

Upon gateway initialization, communication with the server is established, allowing packets to flow in both directions. Periodically, the gateway will send *keepalive* frames, containing a token and its MAC address, to check that the link between the two is operating and the server will respond with a *KA_ACK* in acknowledgement using the same token. This communication between gateway and server is shown in figure 4.18.

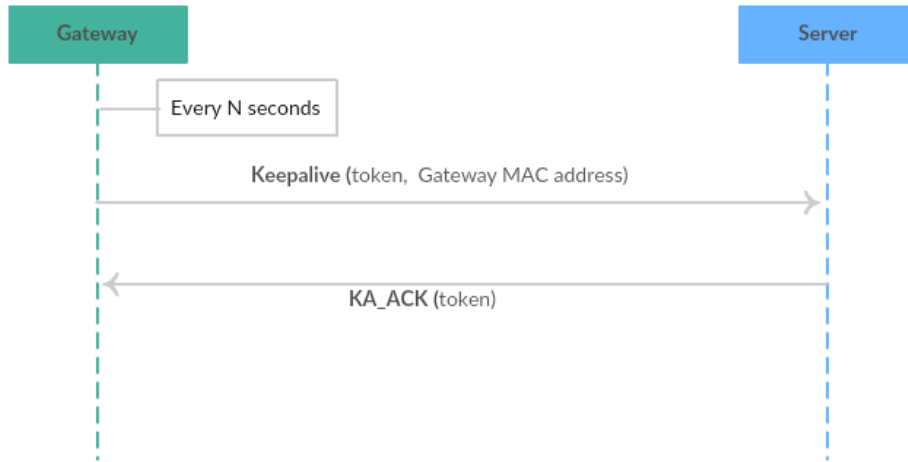


Figure 4.18: Keepalive signal between the gateway and the server.

When the gateway receives a packet from the end node, it appends metadata to it before forwarding it to the server in a *push_data* packet which also contains a token and the addresses of both the end node and the gateway (figure 4.19). The server responds with a *push_ack* in acknowledgement using the same token. The payload of the *push_data* packet is an array named *rxpk* with at least one JSON object. An example can be seen on the next listing 4.1 and a description of each field on table 4.6.

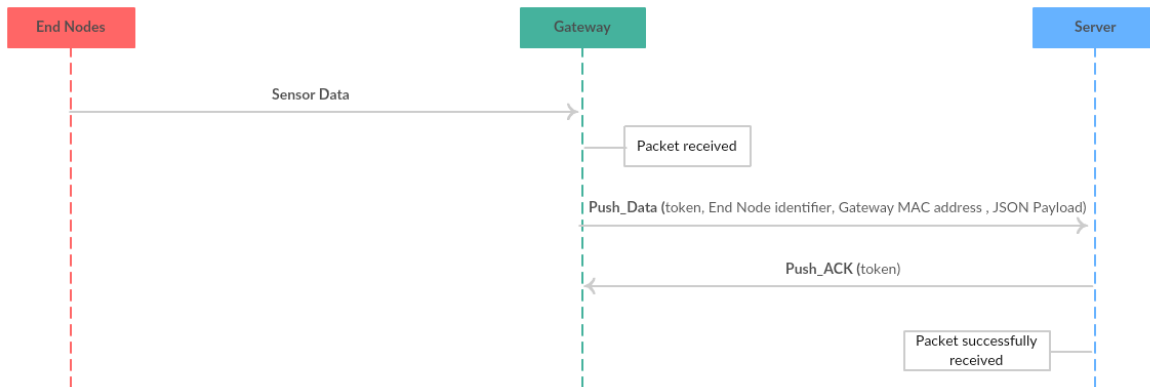


Figure 4.19: Communication between the gateway and the server.

```

1  {"rxpk": [
2      {
3          "data": "QAkTASYA+RsCoNKjkIa6b72INZkLH+
4              10ZusbUv5HkyGqfxb3x4YF114Th9lh YDoMK1sqdM3JUoiAJY=",
5          "time": "2018-11-20T10:58:22.974053Z",
6          "chan": 0,
7          "tmst": 2242018528,
8          "modu": "LORA",
9          "lsnr": -8,
10         "rssi": -119,
11         "rfch": 0,
12         "codr": "4/5",
13         "freq": 868.1,
14         "datr": "SF7BW125",
15         "size": 59
16     }
17 ]
18 }

```

Listing 4.1: Example of a Json data structure.

Name	Function
'data'	Base64 encoded payload
'time'	Time of delivery
'chan'	The channel where the message was received
'tmst'	The timestamps when the message was received
'modu'	The modulation used, in this case LoRa
'lsnr'	The LoRa signal noise ratio of the received message in dB
'rssi'	The received signal strength of the message
'rfch'	The radio frequency chain used
'codr'	LoRa coding rate identifier
'freq'	The frequency used (125 to 500 kHz)
'datr'	The data rate in bits per second
'size'	The payload size in bytes (0 to 255)

Table 4.6: Description of each field from the packet received and associated metadata.

Furthermore, it should also be possible to see the packets received using the TTN website. In figure 4.20 it is shown the traffic of the receiving packets from the end node to the gateway. The device which sent the packets can be recognized through the device address (*'dev addr'*).

time	frequency	mod.	CR	data rate	airtime (ms)	cnt	
▲ 15:34:42	868.1	lora	4/5	SF7 BW 125	87.3	236	devaddr: 26 01 14 E6 payload size: 43 bytes
▲ 15:34:24	868.1	lora	4/5	SF7 BW 125	87.3	233	devaddr: 26 01 14 E6 payload size: 43 bytes
▲ 15:34:06	868.1	lora	4/5	SF7 BW 125	87.3	230	devaddr: 26 01 14 E6 payload size: 43 bytes
▲ 15:33:48	868.1	lora	4/5	SF7 BW 125	87.3	227	devaddr: 26 01 14 E6 payload size: 43 bytes
▲ 15:33:30	868.1	lora	4/5	SF7 BW 125	87.3	224	devaddr: 26 01 14 E6 payload size: 43 bytes
▲ 15:33:11	868.1	lora	4/5	SF7 BW 125	87.3	221	devaddr: 26 01 14 E6 payload size: 43 bytes


Figure 4.20: Packets received at the gateway.

Moreover, as demonstrated in figure 4.21, by clicking on a packet, detailed information can be obtained.

time	frequency	mod.	CR	data rate	airtime (ms)	cnt	
▲ 15:31:05	868.1	lora	4/5	SF7 BW 125	87.3	200	devaddr: 26 01 14 E6 payload size: 43 bytes


Uplink

Dev Address

26 01 14 E6 

Network: The Things Network
Net ID: 0x13
Region: World

Physical Payload

40 E6 14 01 26 00 C8 00 02 67 1C CC 19 01 D5 CA B3 C0 50 C2 69 58 F3 A2 9C 50 4C 3A A9 33 8D D1 37 F4 C5 E6 E0 CF 12 8A F3 9A 22 

Event Data

```

1  {
2    "gw_id": "eui-30aea4fffe2d5bfc",
3    "payload": "QOYUASYAYAACZxzMGQHvYrPAUMJpWPOlnFBM0qkzjdE39MXn4M8Siv0aIg==",
4    "f_cnt": 200,
5    "lora": {
6      "spreading_factor": 7,
7      "bandwidth": 125,
8      "air_time": 87296000
9    },
10   "coding_rate": "4/5",

```

Figure 4.21: Fragment of the packet details at the gateway.

Additionally, it is also possible to see the packets received at the Application Server, figure 4.22.

	time	counter	port	
▲	15:34:06	230	2	payload: 32 32 2E 35 38 2C 20 34 39 2E 33 35 2C 20 31 30 31 30 2E 31 33 2C 20 31 2C 20 30 2E 31 36
▲	15:33:48	229	2	payload: 32 32 2E 36 30 2C 20 34 39 2E 32 31 2C 20 31 30 31 30 2E 31 34 2C 20 31 2C 20 30 2E 31 36
▲	15:33:30	228	2	payload: 32 32 2E 36 32 2C 20 34 39 2E 30 39 2C 20 31 30 31 30 2E 31 33 2C 20 31 2C 20 30 2E 31 36
▲	15:33:11	227	2	payload: 32 32 2E 36 32 2C 20 34 39 2E 32 34 2C 20 31 30 31 30 2E 31 33 2C 20 31 2C 20 30 2E 31 36
▲	15:32:53	226	2	payload: 32 32 2E 36 32 2C 20 34 39 2E 34 30 2C 20 31 30 31 30 2E 31 36 2C 20 31 2C 20 30 2E 31 36
▲	15:32:35	225	2	payload: 32 32 2E 36 33 2C 20 34 39 2E 32 39 2C 20 31 30 31 30 2E 32 30 2C 20 31 2C 20 30 2E 31 36

Figure 4.22: Packets received at the Application Server.

Similarly to the gateway, the packet details can all be seen by clicking on them. However, here the payload is decrypted, as shown in figure 4.23. Each packet sent from the gateway to the Application Server contains information regarding:

- The packet, which includes the end nodes address of the sender, payload and frame counter.
- The transmission, with time, frequency channel used, type of modulation, bandwidth, coding rate and airtime.
- The gateway that has received the packet, containing the location and identification of the gateway and reception details.

▲ 15:31:59 209 2 payload: 32 32 2E 36 32 2C 20 34 39 2E 35 38 2C 20 31 30 31 30 2E 31 36 2C 20 31 2C 20 30 2E 31 36

Uplink

Payload

32 32 2E 36 32 2C 20 34 39 2E 35 38 2C 20 31 30 31 30 2E 31 36 2C 20 31 2C 20 30 2E 31 36

Fields

no fields

Metadata

```
{
  "time": "2018-10-27T14:31:59.822795970Z",
  "frequency": 868.1,
  "modulation": "LORA",
  "data_rate": "SF7Bw125",
  "coding_rate": "4/5",
  "gateways": [
    {
      "gtw_id": "cui-38aee4fffe2d5bfc",
      "timestamp": 1830352809,
      "time": "2018-10-27T14:31:59.771846Z",
      "channel": 0,
      "rssi": -49,
      "snr": 0
    }
  ]
}
```

Estimated Airtime

71.936 ms

Figure 4.23: Packet details at the Application Server.

4.3 Alternative Communication Protocols

Besides the LoRaWAN protocol, the prototype was also developed to work with the Wi-Fi and LTE NB-IoT protocols. A brief overview of these architectures are presented in this section.

4.3.1 Usage of the Wi-Fi Communication Protocol

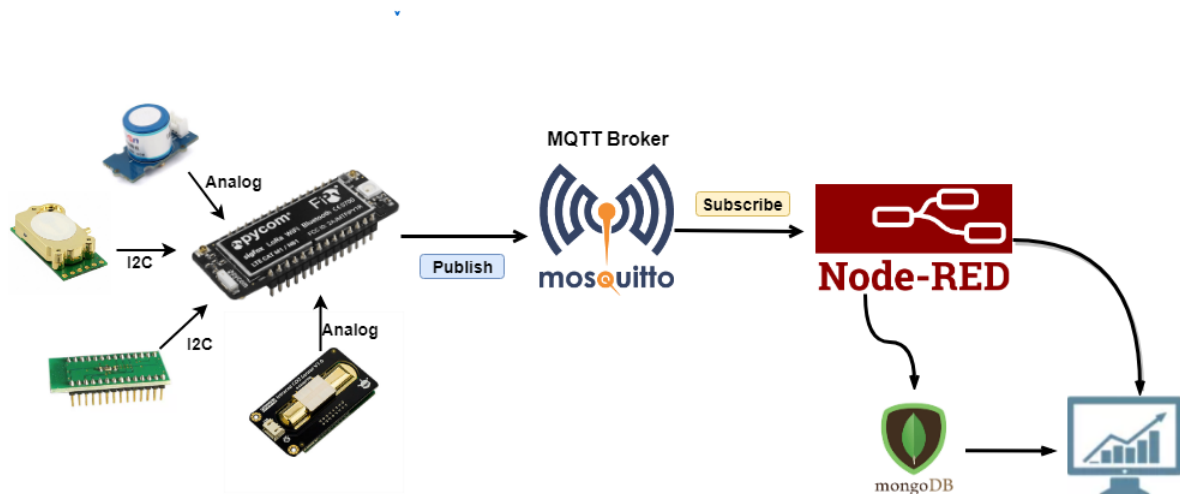


Figure 4.24: MQTT Architecture overview.

As can be seen in figure 4.24, it all starts with the data collected from the sensors. In order to send the data gathered to the server, basic communication through MQTT broker Mosquitto will be used. The broker will publish the data over a predefined messaging topic. Then the broker will forward the message to the MQTT client, which will receive all the sensor data in JSON format. But in order to access the information published, the client has first to act as a subscriber of the topic.

The MQTT client is a web server from Node-RED that runs on a Raspberry Pi 3. At this stage, the data is on the server side and will be saved to a database, MongoDB, for later use and, finally, it will also be possible to visualize and monitor the data in real-time in a Node-RED dashboard, both also running on the Raspberry Pi 3.

4.3.2 Usage of the LTE NB-IoT Communication Protocol

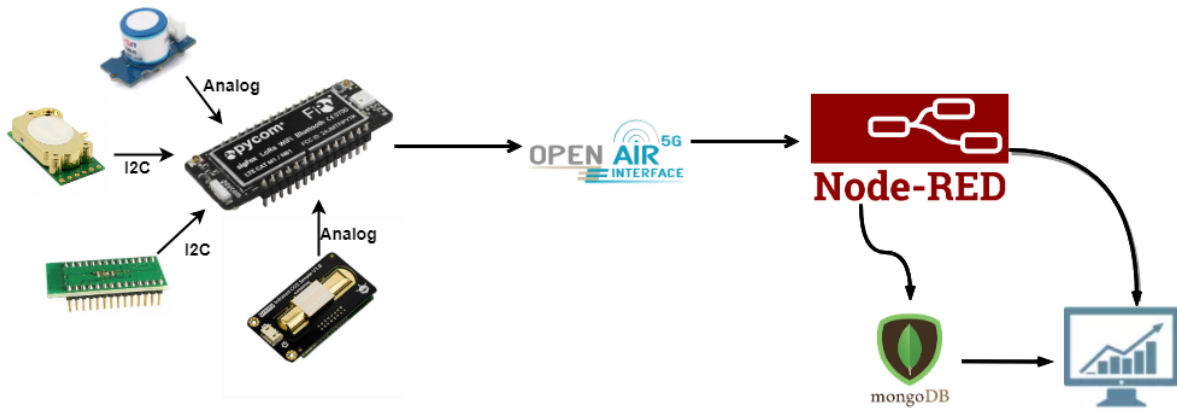


Figure 4.25: LTE NB-IoT architecture overview.

NB-IoT is a recent LPWAN N technology. In figure 4.25 is shown an high-level view of the NB-IoT architecture. As mentioned earlier in chapter 2, the NB-IoT communication protocol is based on the LTE protocol.

OpenAirInterface (OAI) was created by the Mobile Communications Department at EURECOM with the goal of democratizing access to 3GPP radio systems. It is the first implementation of an open software that makes up a 5G Wireless Research and Product Development [57]. It can be used to build a LTE base station and core network on a PC, which is connected to a commercial user equipment to test different configurations and network setups. It can also monitor the network and mobile devices in real-time.

The Institute of Telecommunications (IT) at the University of Aveiro has also been developing OAI. Since LTE NB-IoT is already incorporated in FiPy, it was only necessary to flash a different firmware, the Sequans modem. After that, an attempt to communicate with OAI setup was made using an IoT Subscriber Identity Module (SIM). Unfortunately there were difficulties establishing a connection to NB-IoT network, making it impossible to achieve communication between the FiPy and the OAI setup.

Testing and Results

In order to evaluate the performance of the implemented system architecture, tests were performed. The measurements were carried out in a classroom located in the Department of Electronics, Telecommunications and Informatics (DETI) of the University of Aveiro in 2018 during the winter semester.

The initial tests were executed in October with the first deployed prototype, which contained only the *Bosch Sensortec BME680 Shuttle Board*. Later, in November more tests were performed with the remaining sensor modules.

Section 5.1 presents the calibration and validation of the integrated sensors in this research.

In Section 5.2 are presented the obtained results about the tests performed in a classroom.

5.1 Sensors Calibration and Validation

Calibration of all sensors is one of the main challenges during the development phase.

During the first tests, the temperature readings were reaching 30°C. This was caused by the heat from microcontroller's Central Processing Unit (CPU) also warming up the circuit board. After moving the sensor out of the board and isolating it, as shown in figure 5.1, the temperature readings were pretty different and more accurate.

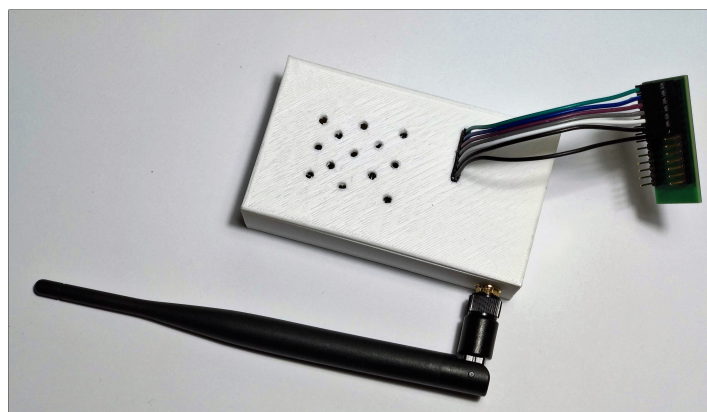


Figure 5.1: Prototype used on tests.

On deploying the prototype with more sensors, this problem was taken into consideration, and all the new sensors added were also placed out of the board, as can be seen in the figure 5.2.

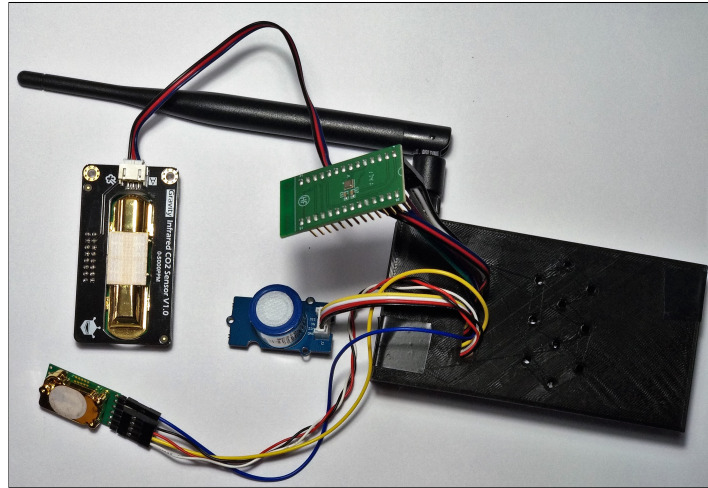


Figure 5.2: Final prototype.

Furthermore, the best way to understand how the device will perform over time is to use a laboratory test chamber to evaluate the accuracy of the measurements. In testing the performance of the device, the chamber was maintained at a temperature between from 11°C to 14°C. The test allowed to verify the sensitivity of the device to instantaneous temperature variations and it was found that the sensor has an error in the measurements of around 0.4°C.

Once there was no proper tool to measure the relative humidity variation, it was compared with an hygrometer ¹ in different days and also in different humidity conditions. It was concluded that the sensor is reading properly with a maximum error of 1 %RH. Until now, the rest of the measurements were not possible to validate, due to the lack of proper tools to do it.

The *Grove-Gas Sensor Oxygen* needs to preheat over more than 48 hours before the first use. In the same way, the *Telaire 6713 CO₂ Module* and *Gravity Infrared CO₂ Sensor* need a warm up time of 24 hours before reading data for actual calculations.

5.2 Classrooms' Indoor Air Quality Evaluation

A low indoor air quality in a classroom is cause for alarm, will decrease the ability of the students to concentrate and affect their health. An optimal comfortable indoor workspace will have a temperature range of 20°C to 26°C. Humidity may also, directly or indirectly, have an impact on the occupant. A high air humidity can create condensation, fungi or other micro-organisms, whereas a low humidity level can cause a sensation of dryness and irritation of skin and mucous membranes of some occupants. Ideal relative humidity should be around 30% to 70% [58].

Another measurement evaluated in this research is the barometric pressure, known as the weight of air pressing down on the atmosphere and Earth's surface. High pressure areas

¹An hygrometer is an instrument used for measuring the humidity.

are often associated with clear weather and low air pressure associated with rainy or snowy weather [59]. Normally, indoor and outdoor air pressure have the same values.

Furthermore, a large number of VOCs can be detected in indoor air. This value, according to guidelines set by the European Commission in 1992, the concentration should be around 0.3ppm [58]. The IAQ index has been developed to identify the quality of indoor air as well as the comfort level of the occupants in an inside area. In this research the IAQ index was determined according to the VOC concentration.

Additionally, the optimal range of oxygen concentration in the air required for an human breathing runs between 19.5 and 23.5 percent [60].

Moreover, high levels of carbon dioxide indicate a lack of fresh air and affects negatively the health, as well as the focus levels and learning ability of students. For outside air, the ideal levels of CO_2 are around 400 ppm, whereas an acceptable indoor CO_2 level in a classroom is 1000 ppm. As long as the classroom is empty this value is easy to maintain. However, humans are the main indoor source of carbon dioxide, so when a classroom is full of students the CO_2 level rises drastically [61]. To avoid adverse health effects, an acceptable range for CO_2 in indoor areas is up to 3500 ppm [58].

In conclusion, maintaining healthy indoor air quality levels requires bringing in fresh outside air and mixing it with inside air.

The setup was tested twice. The first experiment was with the deployed prototype that just contained the *Bosch Sensortec BME680 Shuttle Board*. However, the data collection process in these first tests were not accurate, as explained above. In the second round of tests the measurements obtained were much more reliable. The second environment was a prototype composed already of the four sensors.

The prototype was placed and activated in the classroom a day before starting the tests, so that sensor values could stabilize and be more accurate.

During the lessons, data was constantly being collected. Measurements were taken in the beginning and end of the lessons. Also, the number of occupants in the classroom was taken into consideration. Table 5.2 shows the results obtained in a full day of lessons on November 20th. The data visualization during that day, along with the respective tables from each measurement taken can be seen in appendix C.

The data collection was conducted between 10h00 and 18h00. The weather conditions in Aveiro during that day are presented in the next table 5.1.

Condition	Time
Temperature	17.1°C
Humidity	85%
Pressure	1003.6 hPa

Table 5.1: The maximum outdoor values measured in Aveiro on November 20th [16].

CASE	Time	Occupants	Temperature (°C)	Humidity (%)	Pressure (hPa)	VOCs (ppm)	IAQ	Oxygen (%)	Telaire CO_2 (ppm)	Gravity CO_2 (ppm)	
1	1h	30	Begin	21,31	57,25	1003,29	0,14	1	19,96	673,99	382,5
			End	21,66	58,99	1003,33	0,12	1	19,89	1230	943,75
2	1h30	40	Begin	21,66	58,99	1003,33	0,12	1	19,89	1230	943,75
			End	22,47	60,12	1002,92	0,11	0	19,78	2225	2356,25
3	1h	36	Begin	22,53	58,06	1002,18	0,12	1	20,13	1192,99	1106,25
			End	22,74	59,3	1002,16	0,11	0	19,92	1565,99	1356,25
4	1h	24	Begin	22,82	58,9	1002,24	0,11	0	19,92	1406	1215,62
			End	22,88	58,72	1002,21	0,11	0	19,93	1325,99	1100
5	1h30	26	Begin	22,88	58,72	1002,2	0,11	0	19,95	1327	1125
			End	23,27	58,84	1002,46	0,11	0	19,95	1366	1550

Table 5.2: Measurements from one day of lessons.

The averages of the measured values are summarized in table 5.3. It can be concluded that the classroom has a good ventilation since all the measurements are within the legal norms.

Measurement	Average
Temperature	22°C
Humidity	59.28%
Pressure	1011.27 hPa
VOCs	0.12 ppm
IAQ Index	0
Oxygen	20.15%
Telaire 6713 Module	1340.16 ppm
Gravity Infrared Sensor	1166.39 ppm

Table 5.3: Average measurements from one day of lessons.

Conclusion

The purpose of this dissertation was to develop an IoT system to monitor the indoor quality air in classrooms, to reduce the risk of exposure to contaminants and increase the comfort level of the occupants.

LPWAN technology can interconnect devices that need to stay for an extended period of time in the field and send data over a long range. For this reason becomes an ideal choice for indoor environmental monitoring. LoRaWAN was the LPWAN technology used in order to communicate between the end nodes and the server. In Chapter 4 was present the prototype developed, as well as, the explanation of the system implementation with LoRaWAN protocol, that comprises:

1. **End Node:** Collects the data and sends it to the gateway.
2. **Gateway:** Receives uplink messages from the end node and forward them to the application.
3. **Network server:** Manages several gateways.
4. **Application server:** Stores and processes the data.

The developed system provides a simple way to monitor and control the indoor air quality of enclosed spaces. It was deployed with four sensors for measuring different environmental parameters (temperature, humidity, pressure, VOCs, oxygen and carbon dioxide). It should be noted that more sensors can be added to the prototype.

The prototype was tested by monitoring one classroom located in the Department of Electronics, Telecommunications and Informatics (DETI) of the University of Aveiro during the winter semester. The results obtained were very promising, which represents a significant contribution to indoor quality studies. However, the sensors still require additional experimental validation in laboratory environment in order to verify the calibration of the sensors and to increase its accuracy.

Overall, the developed system provided reliable data, although the acquisition data from one of the CO_2 sensors, the *Gravity Infrared CO_2 Sensor*, fluctuated and was not as accurate when compared to the *Telaire 6713 CO_2 Module*.

The final result achieved during this dissertation met the initial main objectives and a prototype system to environmental monitoring is now available.

6.1 Future Work

Considering that this dissertation is the starting point for the development of a smart system for environmental monitoring, there are still several elements that need to be improved.

- Tests should be performed in order to verify the energy consumption of the developed prototype.
- A portable power source like a battery should be added to the prototype.
- There are sensors that must still to be validated and others that should undergo more tests, in order to validate them with proper conditions and certified meters.
- Deploy an interface to display, in an online web platform or mobile, so that everyone can observe the real time status anytime and anywhere.
- Develop a better enclosed box to fix all the sensors.
- Experiments with NB-IoT were not conducted in this dissertation. More investigation must be made in NB-IoT to understand why it was not possible to achieve communication between the Module Pycom FiPy and the OAI setup.

References

- [1] The Three Software Stacks Required for IoT Architectures. <https://iot.eclipse.org/resources/white-papers/Eclipse%20IoT%20White%20Paper%20-%20The%20Three%20Software%20Stacks%20Required%20for%20IoT%20Architectures.pdf>. [Online accessed 2018-06-28].
- [2] S. Eldridge K. Rose and L. Chapin. The Internet of Things (IoT): An Overview– Understanding the Issues and Challenges of a More Connected World. *The Internet Society*, 2015.
- [3] SEMTECH. <https://www.semtech.com/technology/lora/what-is-lora>. [Online accessed 2018-06-28].
- [4] The things network architecture. <https://www.thethingsnetwork.org/article/the-things-network-architecture-1>. [Online accessed 2018-11-11].
- [5] Lte quick reference. http://www.sharetechnote.com/html/Handbook_LTE_NB_LTE.html. [Online accessed 2018-11-16].
- [6] Nokia IMPACT (Intelligent Management Platform for All Connected Things) IoT Solutions. *Solution sheet Nokia IMPACT IoT Solutions*, pages 1–7, 2016.
- [7] Pycom. specification sheet for the fipy five network development board. http://www.openairinterface.org/?page_id=72. [Online accessed 2018-10-12].
- [8] Datasheet - BME680. https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BME680-DS001-00.pdf. [Online accessed 2018-09-30].
- [9] Grove - Gas Sensor(O_2). http://wiki.seeedstudio.com/Grove-Gas_Sensor-02/. [Online accessed 2018-09-30].
- [10] Tellaire 6713 Series CO_2 Module. <https://www.mouser.com/pdfdocs/AmphenolAdvancedSensorsTellaireT6713CO2Module.PDF>. [Online accessed 2018-09-30].
- [11] Gravity: Analog infrared co2 sensor for arduino sku: Sen0219. https://www.dfrobot.com/wiki/index.php/Gravity:_Analog_Infrared_CO2_Sensor_For_Arduino_SKU:_SEN0219. [Online accessed 2018-11-16].
- [12] LoRa Alliance Technical committee. Lorawan 1.0.2 regional parameters. February 2017.

-
- [13] <https://pycom.io/state-lte-m/>. [Online accessed 2018-06-29].
- [14] Pycom fipy. <https://pycom.io/product/fipy/>. [Online accessed 2018-09-30].
- [15] Bosch Sensortec BME680. https://www.bosch-sensortec.com/bst/products/all_products/bme680. [Online accessed 2018-09-30].
- [16] Current weather in aveiro, portugal. http://climetua.fis.ua.pt/legacy/main/current_monitor/cesamet.htm. [Online accessed 2018-11-20].
- [17] SOCA. <http://soca.av.it.pt/abstract>. [Online accessed 2018-09-30].
- [18] Fredric Paul. Top IoT use cases. <https://www.networkworld.com/article/3208867/internet-of-things/iot-is-everywhere.html>, 2017. [Online accessed 2018-02-19].
- [19] Jacques Bughin Richard Dobbs Peter Bisson Alex Marrs Dan Aharon James Manyika, Michael Chui. The Internet of Things: Mapping the value beyond the hype. *McKinsey Global Institute*, (June):144, 2015.
- [20] K. Rabadiya, A. Makwana, and S. Jardosh. Revolution in networks of smart objects: Social internet of things. In *2017 International Conference on Soft Computing and its Engineering Applications (icSoftComp)*, pages 1–8, Dec 2017.
- [21] Y. Chen. Challenges and opportunities of internet of things. In *17th Asia and South Pacific Design Automation Conference*, pages 383–388, Jan 2012.
- [22] Dave Evans. The Internet of Things: How the Next Evolution of the Internet is Changing Everything. *CISCO white paper*, pages 1–11, 2011.
- [23] H. Tschofenig, J. Arkko, D. Thaler, and D. McPherson. Architectural Considerations in Smart Object Networking. Technical report, 2015. [Online accessed 2018-06-23].
- [24] MQTT. <http://mqtt.org>. [Online accessed 2018-06-26].
- [25] Mosquitto. <http://mosquitto.org>. [Online accessed 2018-06-28].
- [26] OMA. <https://www.omaspecworks.org/what-is-oma-specworks/iot/lightweight-m2m-lwm2m/>. [Online accessed 2018-06-29].
- [27] M. H. Elgazzar. Perspectives on m2m protocols. In *2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*, pages 501–505, Dec 2015.
- [28] Sigfox. <https://www.sigfox.com/en/sigfox-iot-technology-overview>. [Online accessed 2018-06-28].
- [29] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi. Long-range communications in unlicensed bands: the rising stars in the iot and smart city scenarios. *IEEE Wireless Communications*, 23(5):60–67, October 2016.
- [30] LoRa Alliance. <https://lora-alliance.org>. [Online accessed 2018-06-28].

- [31] J. de Carvalho Silva, J. J. P. C. Rodrigues, A. M. Alberti, P. Solic, and A. L. L. Aquino. Lorawan — a low power wan protocol for internet of things: A review and opportunities. In *2017 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech)*, pages 1–6, July 2017.
- [32] A technical overview of LoRa® and LoRaWAN™. https://www.tuv.com/media/corporate/products_1/electronic_components_and_lasers/TUeV_Rheinland_Overview_LoRa_and_LoRaWANtmp.pdf. [Online accessed 2018-06-28].
- [33] Ensuring device and radio security in low-power iot wireless connections. <https://www.digikey.com/en/articles/techzone/2017/aug/ensuring-device-radio-security-lorawan>. [Online accessed 2018-10-07].
- [34] LoRa Alliance Technical Committee. Lorawan specification. October 2017.
- [35] J. Navarro-Ortiz, S. Sendra, P. Ameigeiras, and J. M. Lopez-Soler. Integration of lorawan and 4g/5g for the industrial internet of things. *IEEE Communications Magazine*, 56(2):60–67, Feb 2018.
- [36] The things network. https://de.wikipedia.org/wiki/The_Things_Network. [Online accessed 2018-10-07].
- [37] 3GPP. <http://www.3gpp.org>. [Online accessed 2018-06-29].
- [38] M. Elsaadany, A. Ali, and W. Hamouda. Cellular lte-a technologies for the future internet-of-things: Physical layer features and challenges. *IEEE Communications Surveys Tutorials*, 19(4):2544–2572, Fourthquarter 2017.
- [39] Cellular iot explained - nb-iot vs. lte-m vs. 5g and more. <https://www.leverage.com/blogpost/cellular-iot-explained-nb-iot-vs-lte-m>. [Online accessed 2018-11-16].
- [40] What is lte-m? <https://5g.co.uk/guides/what-is-lte-m/>. [Online accessed 2018-11-16].
- [41] W. Ayoub, A. E. Samhat, F. Nouvel, M. Mroue, and J. Prévotet. Internet of mobile things: Overview of lorawan, dash7, and nb-iot in lpwans standards and supported mobility. *IEEE Communications Surveys Tutorials*, pages 1–1, 2018.
- [42] What are the differences between lte-m and nb-iot cellular protocols? <https://www.digi.com/videos/what-are-the-differences-between-lte-m-and-nb-iot>. [Online accessed 2018-11-11].
- [43] What is lpwa for the internet of things? https://www.sierrawireless.com/iot-blog/iot-blog/2016/08/lpwa_for_the_iot_part_3_a_guide_to_decoding_lpwa_technologies/. [Online accessed 2018-11-16].
- [44] Altice Labs. Iot cellular networks. Oct 2017.
- [45] A. Botta, W. de Donato, V. Persico, and A. Pescapé. On the integration of cloud computing and internet of things. In *2014 International Conference on Future Internet of Things and Cloud*, pages 23–30, Aug 2014.

- [46] Node-RED. <https://nodered.org>. [Online accessed 2018-09-17].
- [47] Node.js. <https://nodejs.org/en/about/>. [Online accessed 2018-09-17].
- [48] Y. Li and S. Manoharan. A performance comparison of sql and nosql databases. In *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, pages 15–19, Aug 2013.
- [49] J. S. van der Veen, B. van der Waaij, and R. J. Meijer. Sensor data storage performance: Sql or nosql, physical or virtual. In *2012 IEEE Fifth International Conference on Cloud Computing*, pages 431–438, June 2012.
- [50] Zachary Parker, Scott Poe, and Susan V. Vrbsky. Comparing nosql mongodb to an sql db. In *Proceedings of the 51st ACM Southeast Conference, ACMSE '13*, pages 5:1–5:6, New York, NY, USA, 2013. ACM.
- [51] Nadeem Qaisar Mehmood, Rosario Culmone, and Leonardo Mostarda. "modeling temporal aspects of sensor data for mongodb nosql database". *Journal of Big Data*, page 8, March 2017.
- [52] Mongodb. <https://www.mongodb.com/what-is-mongodb>. [Online accessed 2018-10-08].
- [53] M. E. E. Alahi, N. Pereira-Ishak, S. C. Mukhopadhyay, and L. Burkitt. An internet-of-things enabled smart sensing system for nitrate monitoring. *IEEE Internet of Things Journal*, pages 1–1, 2018.
- [54] Air quality index. https://en.wikipedia.org/wiki/Air_quality_index#cite_note-38. [Online accessed 2018-10-23].
- [55] U.S. Environmental Protection Agency. Technical assistance document for the reporting of daily air quality – the air quality index (aqi). September 2018.
- [56] D. Lohani and D. Acharya. Smartvent: A context aware iot system to measure indoor air quality and ventilation rate. In *2016 17th IEEE International Conference on Mobile Data Management (MDM)*, volume 2, pages 64–69, June 2016.
- [57] Openairinterface. <https://pycom.io/wp-content/uploads/2018/08/fipySpecsheetAugust2017n2-1.pdf>. [Online accessed 2018-10-12].
- [58] Commission of the European Communities. Indoor air quality and its impact on man', report no. 11: Guidelines for ventilation requirements in buildings. 1992.
- [59] What is the range of barometric pressure? <https://sciencing.com/range-barometric-pressure-5505227.html>. [Online accessed 2018-11-20].
- [60] Minimum oxygen concentration for human breathing. <https://sciencing.com/minimum-oxygen-concentration-human-breathing-15546.html>. [Online accessed 2018-11-24].
- [61] Sick classrooms caused by rising co2 levels. <https://energyalliancegroup.org/sick-classrooms-require-energy-efficient-solutions-2/>. [Online accessed 2018-11-24].

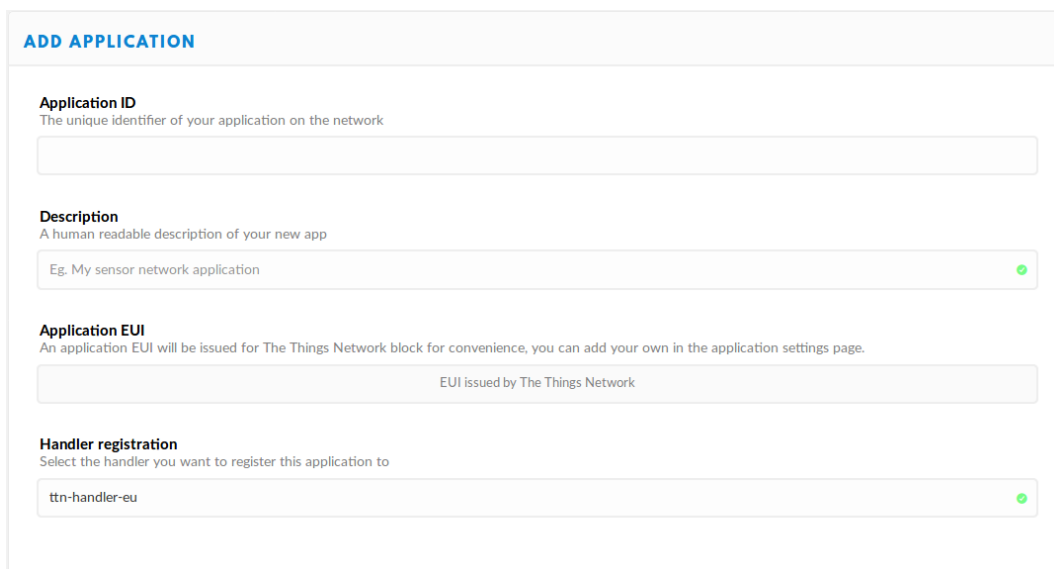
- [62] B. A. Mozzaquatro, R. Jardim-Goncalves, and C. Agostinho. Situation awareness in the internet of things. In *2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 982–990, June 2017.
- [63] Partha Pratim Ray. A Survey of IoT Cloud Platforms. *Future Computing and Informatics Journal*, 1(1-2):35–46, 2017.
- [64] M. B. Yaakop, I. A. A. Malik, Z. bin Suboh, A. F. Ramli, and M. A. Abu. Bluetooth 5.0 throughput comparison for internet of thing usability a survey. In *2017 International Conference on Engineering Technology and Technopreneurship (ICE2T)*, pages 1–6, Sept 2017.
- [65] A. J. Wixted, P. Kinnaird, H. Larijani, A. Tait, A. Ahmadiania, and N. Strachan. Evaluation of lora and lorawan for wireless sensor networks. In *2016 IEEE SENSORS*, pages 1–3, Oct 2016.
- [66] T. Szydlo, R. Brzoza-Woch, J. Sendorek, M. Windak, and C. Gniady. Flow-based programming for iot leveraging fog computing. In *2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, pages 74–79, June 2017.
- [67] B. Al-Shargabi and O. Sabri. Internet of things: An exploration study of opportunities and challenges. In *2017 International Conference on Engineering MIS (ICEMIS)*, pages 1–4, May 2017.
- [68] G. Nivetha; M. Udhayamoorthi; K. Theebak Kumar; C. Senthil Kumar. Challenges and opportunity in internet of things (iot). *International Journal for Innovative Research in Science Technology*, pages 166–173, 2017.
- [69] E. Balandina, S. Balandin, Y. Koucheryavy, and D. Mouromtsev. Iot use cases in health-care and tourism. In *2015 IEEE 17th Conference on Business Informatics*, volume 2, pages 37–44, July 2015.
- [70] S. Din, M. M. Rathore, A. Ahmad, A. Paul, and M. Khan. Sdiot: Software defined internet of thing to analyze big data in smart cities. In *2017 IEEE 42nd Conference on Local Computer Networks Workshops (LCN Workshops)*, pages 175–182, Oct 2017.
- [71] Brian Russell and Drew Van Duren. *Practical Internet of Things Security*. 2016.
- [72] P. Thota and Y. Kim. Implementation and comparison of m2m protocols for internet of things. In *2016 4th Intl Conf on Applied Computing and Information Technology/3rd Intl Conf on Computational Science/Intelligence and Applied Informatics/1st Intl Conf on Big Data, Cloud Computing, Data Science Engineering (ACIT-CSII-BCD)*, pages 43–48, Dec 2016.
- [73] M. B. Yassein, M. Q. Shatnawi, and D. Al-zoubi. Application layer protocols for the internet of things: A survey. In *2016 International Conference on Engineering MIS (ICEMIS)*, pages 1–4, Sept 2016.
- [74] J. Oh and H. Song. Study on the effect of lte on the coexistence of nb-iot. In *2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 610–612, July 2018.

- [75] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne. Understanding the limits of lorawan. *IEEE Communications Magazine*, 55(9):34–40, 2017.
- [76] R. Ratasuk, J. Tan, N. Mangalvedhe, M. H. Ng, and A. Ghosh. Analysis of nb-iot deployment in lte guard-band. In *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, pages 1–5, June 2017.
- [77] M. Elsaadany and W. Hamouda. The new enhancements in lte-a rel-13 for reliable machine type communications. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–5, Oct 2017.
- [78] Joseph Finnegan and Stephen Brown. A comparative survey of lpwa networking. *CoRR*, abs/1802.04222, 2018.
- [79] G. Marques and R. Pitarma. Health informatics for indoor air quality monitoring. In *2016 11th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6, June 2016.
- [80] A. Caron, N. Redon, B. Hanoune, and P. Coddeville. Unsupervised k-means learning applied to the investigation of indoor air quality events with electronic gas sensors networks. In *2017 ISOCS/IEEE International Symposium on Olfaction and Electronic Nose (ISOEN)*, pages 1–3, May 2017.
- [81] C. Lee and J. Lee. Development of indoor air quality supervision systems using zigbee wireless networks. In *2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 95–98, May 2018.
- [82] D. Lohani and D. Acharya. Smartvent: A context aware iot system to measure indoor air quality and ventilation rate. In *2016 17th IEEE International Conference on Mobile Data Management (MDM)*, volume 2, pages 64–69, June 2016.
- [83] Emekcan Aras, Gowri Ramachandran, Piers Lawrence, and Danny Hughes. Exploring the security vulnerabilities of lora. In *2017 3rd IEEE International Conference on Cybernetics*, pages 1–6, 06 2017.
- [84] Silvia Vilcekova, Peter Kapalo, Ludmila Meciarova, Eva Krídlová Burdová, and Veronika Imreczeová. Investigation of indoor environment quality in classroom - case study. *Procedia Engineering*, 190:496–503, 12 2017.
- [85] Dirk Ahlers, Frank Kraemer, Anders Bråten, Xiufeng Liu, F Anthonisen, Patrick Driscoll, and John Krogstie. Analysis and visualization of urban emission measurements in smart cities. In *21st International Conference on Extending Database Technology*, 03 2018.
- [86] Olaf Kolkman. Introducing Collaborative Security, our approach to Internet security issues — Internet Society. <https://www.internetsociety.org/blog/2015/04/introducing-collaborative-security-our-approach-to-internet-security-issues/>, 2015. [Online accessed 2018-04-10].

Application Server

In order to send data over LoRaWAN, first it is necessary to select a network that supports the data from the end nodes. In this implementation, the public network *The Things Network* was used. To acquire the network keys to connect the network with the TTN application, the following procedures need to be followed:

1. Register an account in TTN website - <https://www.thethingsnetwork.org>.
2. Navigate to the TTN console where options can be found to either register a gateway or create an application.
3. Starting by creating an application, it would be required to fill-in some fields to help in describing and identifying the application as shown in figure A.1.



The screenshot shows a web form titled "ADD APPLICATION" with the following sections:

- Application ID**: The unique identifier of your application on the network. An empty text input field is provided.
- Description**: A human readable description of your new app. A text input field contains the example text "Eg. My sensor network application" and has a green checkmark on the right.
- Application EUI**: An application EUI will be issued for The Things Network block for convenience, you can add your own in the application settings page. A text input field contains the text "EUI issued by The Things Network".
- Handler registration**: Select the handler you want to register this application to. A dropdown menu shows "ttn-handler-eu" with a green checkmark on the right.

Figure A.1: Application configuration example.

Enter a unique *Application ID* as well as a *Description and Handler Registration*.

- To connect end nodes to TTN at least one device must be linked to the application, in this case a *FiPy* device was added. As shown in figure A.2, the *Device ID* must be in lowercase. It is unique per device, and so is the *Device EUI* that identifies the device on the LoRaWAN network during the join request. This 16-character long string is assigned to every device by the chip manufacturer, in this case we read it by using the following code listing A.1.

```

1  from network import LoRa
2  import ubinascii
3  lora = LoRa()
4  print(ubinascii.hexlify(lora.mac()).decode('ascii'))

```

Listing A.1: Sample code to get the Device EUI.

The rest of the fields can be generated automatically. The default authentication of the device is OTAA, the user just needs to input the *Application EUI* and *Application Key*.

REGISTER DEVICE [bulk import devices](#)

Device ID
This is the unique identifier for the device in this app. The device ID will be immutable.

Device EUI
The device EUI is the unique identifier for this device on the network. You can change the EUI later.

App Key
The App Key will be used to secure the communication between you device and the network.

App EUI

Figure A.2: Device configuration example.

There were however some problems on trying to connect to *FiPy* over the OTAA authentication network. It was unable to connect to it, therefore ABP was used instead of OTAA to connect to the network server. In this way, some parameters in the settings about the activation method of the device needed to be added to the TTN application and configured manually. The *Device Address*, *Network Session Key* and *Application Session Key*.

Registering the Gateway in TTN

The gateway must be registered on TTN before it is able to forward packets to TTN console.

1. Navigate to the TTN web site.
2. At the TTN Console, click on *Gateways* and then *Register gateway*.
3. Fill-in the fields as required, as shown in figure B.1. The *Gateway EUI* is unique per device, generated by the chip manufacturer and may be obtained as the *Device EUI*.

The screenshot shows the 'REGISTER GATEWAY' form in the TTN console. It contains the following fields and options:

- Gateway ID**: A text input field with the placeholder text 'A unique, human-readable identifier for your gateway. It can be anything so be creative!'.
- I'm using the legacy packet forwarder**: A checkbox that is currently unchecked. Below it is the text 'Select this if you are using the legacy [Semtech packet forwarder](#)'.
- Description**: A text input field with the placeholder text 'A human-readable description of the gateway'.
- Frequency Plan**: A dropdown menu with the placeholder text 'The [frequency plan](#) this gateway will use'. The selected option is 'Europe 868MHz'.
- Router**: A dropdown menu with the placeholder text 'The router this gateway will connect to. To reduce latency, pick a router that is in a region which is close to the location of the gateway.' The selected option is 'ttn-router-eu'.

Figure B.1: Gateway registration configuration example.

4. Selecting the frequency plan is very important. It depends on the region where LoRa is operating, in this case is European ISM band, which is 860 MHz band. In Portugal, LoRa works in two channel plans, **EU863-870 MHz** and **EU433 MHz**. However,

TTN does not have frequency plan EU433 MHz yet. Having selected the frequency, TTN automatically changes the Router selection to the appropriate location.

5. If everything has been set up correctly and the gateway is plugged in, it should be seen the status of the gateway as *connected*, as in figure B.2.

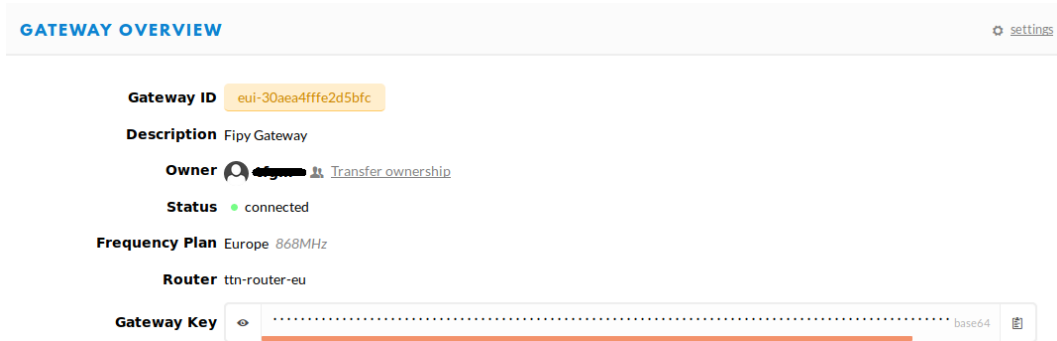


Figure B.2: Gateway Connected.



User Interface

In the next two figures, figure C.1 and figure C.2, is shown an user interface, a real time dashboard for data visualisation. A gauge graphic is used to display the most recent value for each environmental parameter measured; Charts graphics are used to display values measured in the last hour and for the last 24 hours.

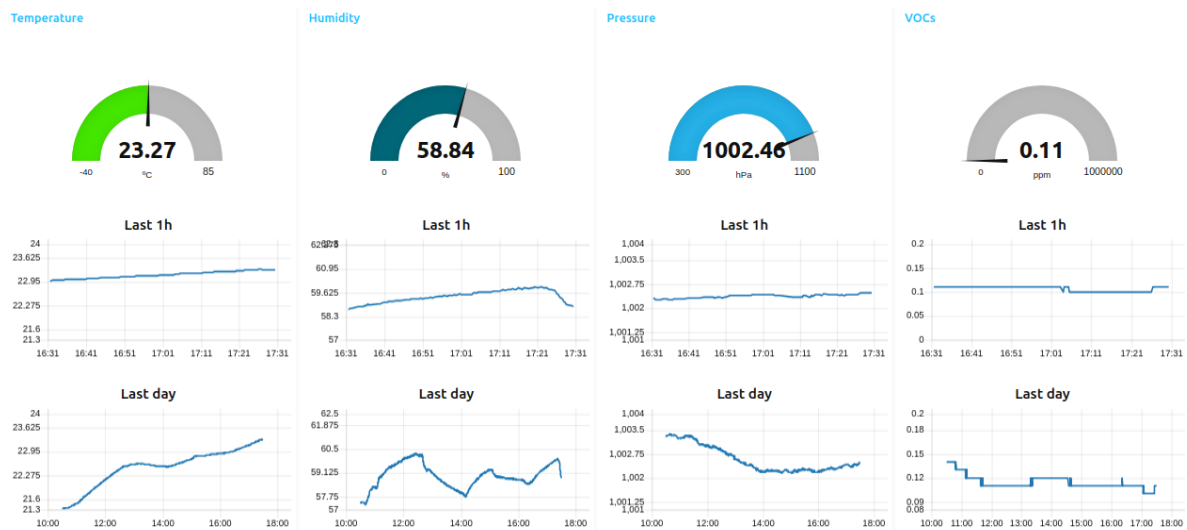


Figure C.1: Temperature, Humidity, Pressure and VOCs visualisation.

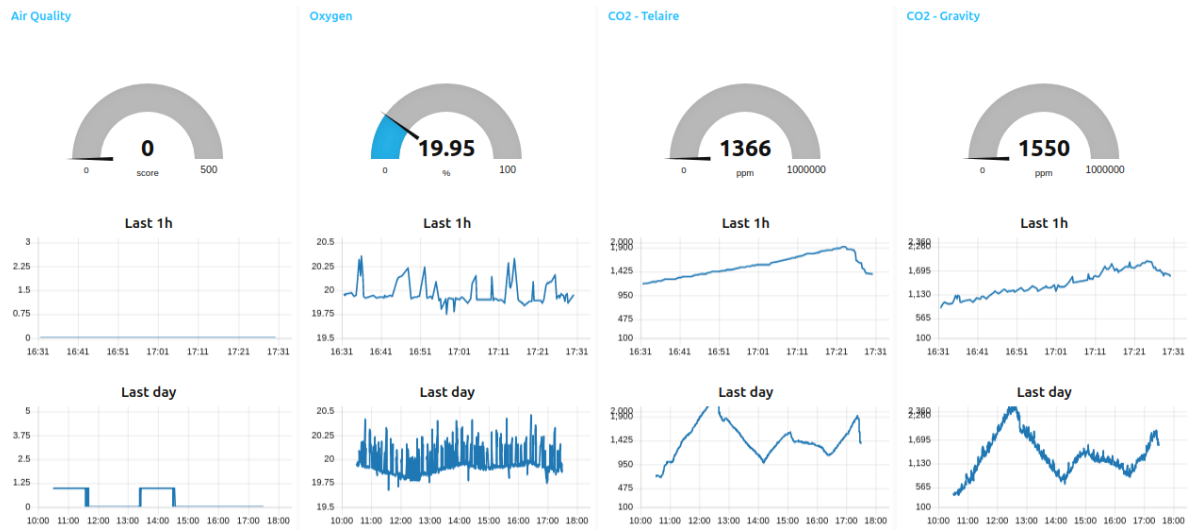


Figure C.2: Air quality index, Oxygen and both CO_2 sensors visualisation.