UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Barcelona Est

TREBALL FI DE GRAU

Grau en Enginyeria Electrònica

WIRELESS SENSOR NETWORK

Memòria i Annexos

Autor: Jordi Calvo Cervera
Jose Antonio Gil Laso
Carlos Gonzalez Ortega
Director: John Rode

Convocatòria: Juny 2017

Resum

Aquest document és l'informe final per al projecte Lora realitzat per tres estudiants Erasmus internacionals a la Universitet d'Aarhus com el seu projecte de llicenciatura. L'objectiu d'aquest projecte és aprofitar la tecnologia d'Internet of Things mitjançant el desenvolupament d'un dispositiu sensor de baix consum. Aquest dispositiu consistirà en una porta d'enllaç i un node. On el node serà el responsable de la recopilació de dades i el trametrà a la porta d'enllaç. Després, la porta d'enllaç guardarà totes les dades en una base de dades per fer estadístiques en futures aplicacions.

Resumen

Este documento es el informe final para el proyecto LoRa realizado por tres estudiantes Erasmus internacionales en la Universitet de Aarhus como su proyecto de licenciatura. El objetivo de este proyecto es aprovechar la tecnología de Internet of Things mediante el desarrollo de un dispositivo sensor de bajo consumo. Este dispositivo consistirá en una puerta de enlace y un nodo. Donde el nodo será el responsable de la recopilación de datos y lo enviará a la puerta de enlace. Luego, la puerta de enlace guardará todos los datos en una base de datos para hacer estadísticas en futuras aplicaciones.

Abstract

This document is the final report for the LoRa project made by three international Erasmus students in the Aarhus Universitet as their bachelor's project. The goal of this project is to take profit of the Internet of Things technology by developing a low consuming sensing device. This device will consist in one gateway and one node. Where the node will be the responsible of the data gathering and will send it to the gateway. Then, the gateway will save all the data in a database in order to make statistics in future applications.

Índex

1. INTRODUCTION

The internet of things is a concept that is based on the interconnection of any object with any other around it with the goal of making all these devices communicate with each other and therefore be more intelligent and independent. Here it comes the name of “Internet of things”.

A technology created by this propose is LoRa, which is a technology designed to have a long range and a low bit rate to connect objects such as sensors.

This project makes use of this technology to develop a low consuming sensing device which can be used, between other applications, as an environmental controller. This device has two distinguished parts: a node and a gateway.

The node gathers the data, in this case, this data will consist in the sensing of temperature, humidity, pressure and relative position, but new features can be added. When collecting this data, the sensor is also sending it to the gateway, which will save it and upload it in a database in order to have a tracking of our data gathering. Figure 1 shows an overview of the system explained in the requirements specification [3].

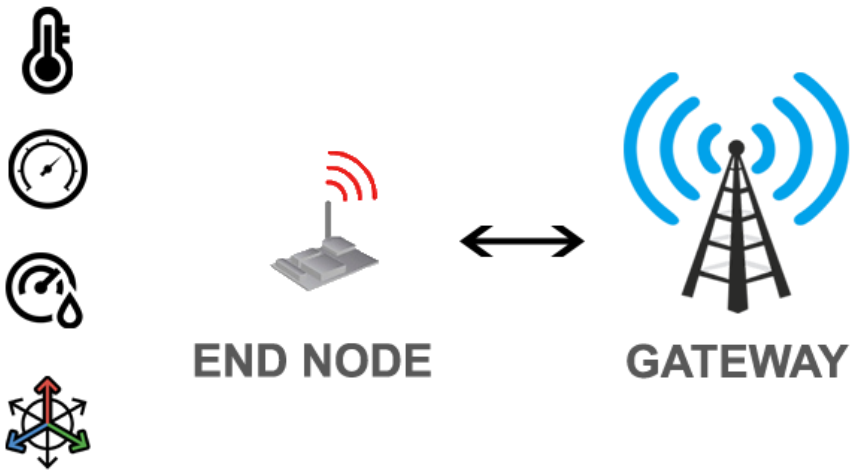

Figure 1: Overview of the system

1.1. Tasks

The schedule of the tasks are also shown in the Schedule of the project document [22]

1.1.1. Hardware tasks

The development of the hardware has three distinguished parts: the analysis of the components stated in the project description, the selection of the components to satisfy the requirements of the project description and the design of the hardware.

1.1.1.1. Analysis of components

The project description states that the platform for the gateway and the platform for the node has to be Raspberry Pi 2 and an Arduino ATmega2560 [1]. An analysis about their functioning and interfaces has been necessary in order to take as much profit as possible of them.

1.1.1.2. Selection of components

As explained previously, the node has to be capable of recording temperature, humidity, relative position and pressure, as well as sending it via LoRa. So that, sensors and a LoRa daughter has to be selected previously in order to make sure of selecting the components which fits best to the project.

1.1.1.3. Hardware design

A selection of the interfaces and a PCB design has been necessary to develop in order to make a project as smarter as possible.

1.1.1.4. Division of tasks

	Carlos	**Jordi**	**Jose**
Analysis			x
Selection	x	x	
Hardware design		x	
Veroboard soldering		x	

Table 1: Division of hardware task

1.1.2. Software tasks

1.1.2.1. Sensing

Program the node microcontroller in order to get data from the sensors.

1.1.2.2. Timers

Program the node microcontroller in order to synchronize the data gathering of the sensor, as well as the creation of a package and the sending of the packages.

1.1.2.3. LoRa communication - Node

Program the node microcontroller in order to make it able to send packets via LoRa communication.

1.1.2.4. LoRa communication - Gateway

Program the gateway microcontroller in order to make it able to send packets via LoRa communication.

1.1.2.5. Database

Creation and integration of the database in the gateway.

1.1.2.6. Division of tasks

	Carlos	**Jordi**	**Jose**
Sensing	x		
Timers	x		
LoRa node		x	
LoRa gateway			x
Database			x

Table 2: Division of software tasks

2. Requirements

2.1. Objective

This project's goal is to realize a system what It can send some relevant information from a place, as environment information. The end node system is responsible for capturing important information by the temperature and humidity sensor, the pressure sensor and accelerometer. By the other hand, there is the Gateway what provides a database for storing.

The wireless LoRa communication is the main part of the project because it is the link between those subsystem which have to provide data from end node to gateway. Both microcontrollers have LoRa cards settled on shield connected to an antenna.

The end node will receive data every 5 seconds from every sensor, placed in a PCB, via I2C for pressure and accelerometer and serial data for temperature and relative humidity. Besides, there is one buttom what is going to reset the accelerometer for settling a new origin.

About the gateway, it consists in a microcontroller with a LoRa shield that is receiving every 5 seconds this gathered data from node and putting it in a database to save new information about the node's environment.

2.2. System overview

As it was shown in the figure 1, the system overview consists in two parts: a gateway which will receive all the data and save it in a database and an end node which will collect data and send it to the gateway. The communication will be, as It was mentioned before, via LoRa, which is the main feature of this project.

The end node will be composed by the following sensors:

- Temperature sensor
- Pressure sensor
- Humidity sensor
- Accelerometer

2.3. Context diagram

Figure 2: Context diagram

2.4. Actors

It is relevant to explain who are acting in this system and how they are setted. The primary actor is the environment, because It is going to give us new data every time. While from the gateway, the user will be able to get data from the database, who is secondary actor in this entire system.

Actor name	Environment
Type	Primary
Description	It is where the data is collected from.

Actor name	User
Type	Secondary
Description	The user is the actor which will watch the

	collected data

Table 3, 4. Explicative tables of the system actors

2.5. Use case diagram

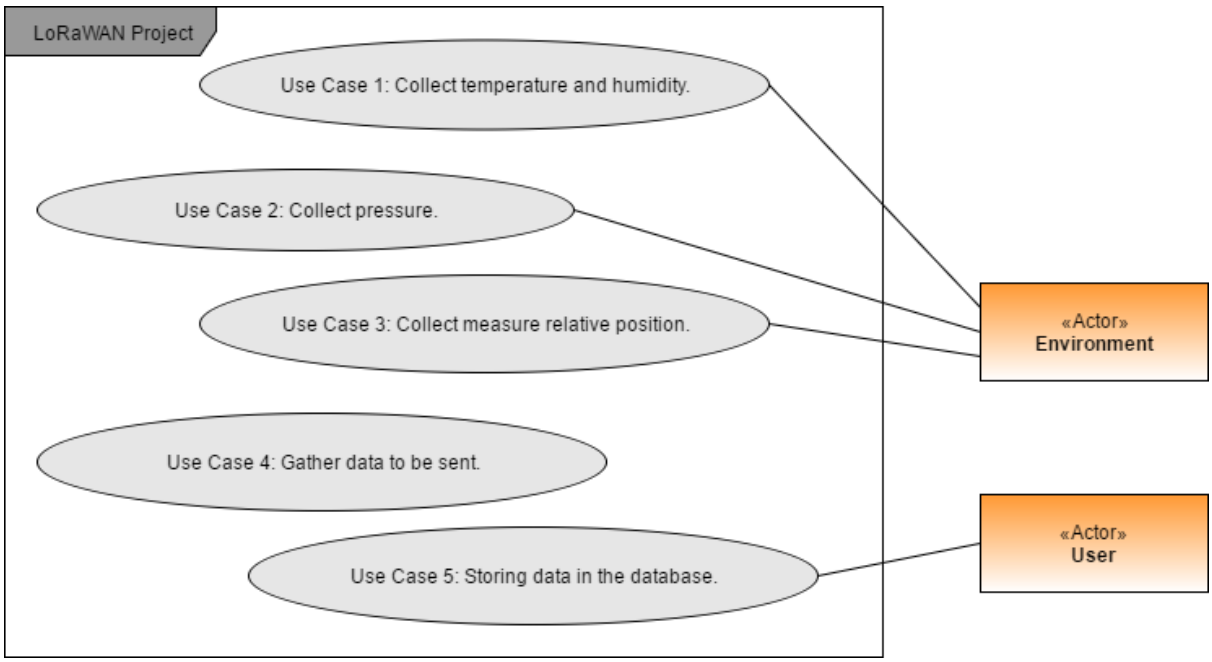

Figure 3. Summary of use cases.

2.6. Use cases

The intention of this part is to describe briefly all use cases for getting an overall idea. Anyway, more detailed information can be found in the Requirement Specification document [3].

2.6.1. Use case 1: Collect temperature and humidity

The DHT22 provides relative humidity and temperature every 5 seconds, requested by a timer and set it up for sending.

2.6.2. Use case 2: Collect pressure

The BMP280 is on charge of pressure, which is going to capture pressure values and send it every time that timer calls it.

2.6.3. Use case 3: Collect relative position

The MPU6050 will detect initial position and it will calculate accelerations in every move for transforming it in coordinates to send to microcontroller every 5 seconds.

2.6.4. Use case 4: Gather data to be sent

All obtained information will be packaged as a string for being transmitted to the gateway via wireless LoRa communication.

2.6.5. Use case 5: Storing data in the database

Once both microcontrollers are paired, the gateway will be ready for received those information packages.

3. Limits

- The accelerometer is not a great tool for short tracking because its accuracy is not really well in low-cost sensors. These sensors always have an offset error which will be increased by time. In general, accelerometer-based position and velocity estimates from low-cost sensors are poor because the orientation of the sensor must be known with a high degree of accuracy. For further details, check "Accelerometer calculations" referenced in "Software Architecture and Design" document.

-There are no ACK at wireless communication, so one packet could be sent but no received.

-The gateway must be initialized manually; the system does not start at root.

-The LoRa settings should be modified by coding.

-The LoRa statistics are not stored at mongoDB.

-The database is working in a local server; the gateway is not connected to internet.

-MongoDB can store a maximum of 2GB archive at raspberry pi (in a Local mode).

4. Realization

4.1. Methods

4.1.1. Gantt chart

The gantt chart has been the method used for the project scheduling. This diagram consist in bars that show the time that will take every step for the project, with deliverables and relations with other activities.[22]

4.1.2. UML

The using of Unified Modeling Language (UML) has been used for the software development (sequence and class diagrams) and the requirement specifications(Use cases). It consist in providing a standard way to visualize the design of a system.

4.1.3. SysML

The System Modeling Language (SysML) has been used for the hardware development. It has been used the Block Definition Diagram (BDD) to define the system and the Internal Block Diagram (IBD) to show how every part of the system is connected.

4.2. Development tools

4.2.1. Google drive

Google drive is a free cloud storage provided by Google that allows sharing documents and files between various users. This tool has been used for developing the documents of the project simultaneously by each member of the group.

4.2.2. Draw.io

Draw.io is a free cloud schematic editor provided by Google that allows to save the works done in google drive. This tool has been used for developing of the IBD and BDD schematics, as well as sequence diagrams and static diagrams.

4.2.3. Fritzing

Fritzing is an open-source software tool that allows the user to carry out hardware designs with the import of components. This tool has been used for the development of hardware schematics and the PCB design.

4.2.4. Arduino IDE

Arduino IDE is the tool provided by Arduino for coding its platforms in a custom version of C++. This has been the main tool for the node microcontroller programming.

4.2.5. Atmel Studio

Atmel Studio is the tool provided by Atmel for coding microcontrollers in C language. This tool has been used specifically for some parts of the node's code.

4.2.6. IDLE

IDLE is an integrated development environment for Python. This IDE is pre-installed on raspbian OS. This tool has been used for store JSON object in mongoDB and manage all the system of the raspberry pi.

4.2.7. Nano

Nano is an edit text tool pre-installed on Raspbian OS. This tool has been used for modify and create Python and C++ code when the LoRa board was mounted on raspberry pi and the Screen could not be used.

4.2.8. Terminal (Linux console)

Terminal is a system console internal to the Linux kernel and the main feature to manage the raspberry pi when the it has mounted a LoRa board and a SSH communications is needed to control this devices.

4.2.9. PuTTY

PuTTY is an SSH and telnet client, developed for Windows platform. PuTTY allows to establish a communication between raspberry pi and our computers by Windows OS without compatibility problems.

4.3. Analysis

In this section it is described briefly some of the conclusions made after the analysis. For further details in the analysis document is all described deeply [2].

4.3.1. Node platform

The Microcontroller platform for the node stated by the project description was the Arduino Mega 2560[1]. And after the analysis [2] these are some of the remarkable features.

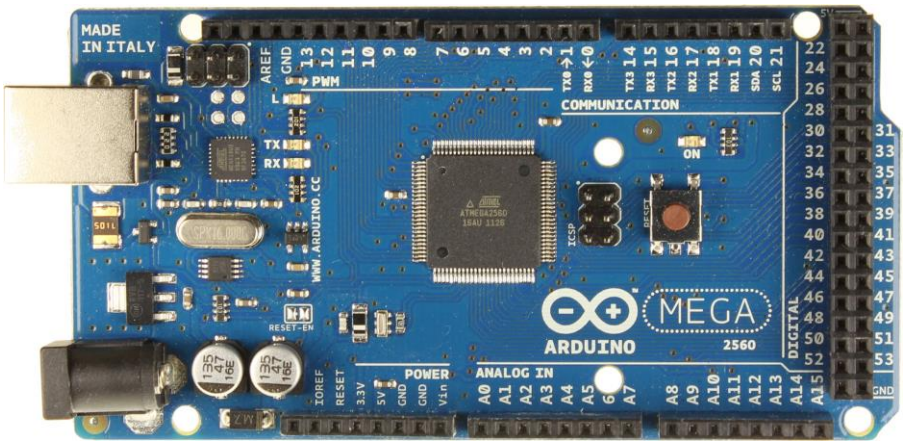

Figure 4: Arduino Mega 2560

	Arduino Mega 2560
Microcontroller	ATmega2560
Clock speed	16MHz
Input voltage	7-12 V
Flash memory	256 KB of which 8 KB used by bootloader
Digital I/O Pins	54 (14 PWM output and 16 analogic inputs)
Pin description	Each 54 pin could be use like digital pins. In addition some pins have

	specialized functions: ● 4xSerial function: there are 8pins for this purpose. ● 6xExternal interrupts function: 1 pin per external interrupt. ● 14xPWM: 14 pins which provide 8-bit PWM output. ● 1xSPI: this function has 1 pin for MISO, 1 for MOSI, 1 for SCK, 1 SS. ● 1xLED: at pin 13. ● 1x I2C: 1 pin for SDA and another 1 pin for SCL ● 4 pins for Tx and others 4 pins for Rx
SRAM	8 KB
EEPROM	4 KB
Reference	[17][18]

Table 5. Arduino Mega 2560 characteristics

4.3.2. Gateway platform

As the the Microcontroller platform for the node, the gateway's platform was also stated by the project description[1], but in this case it was a Raspberry Pi 2. And during the analysis [2] these features have been found among others.

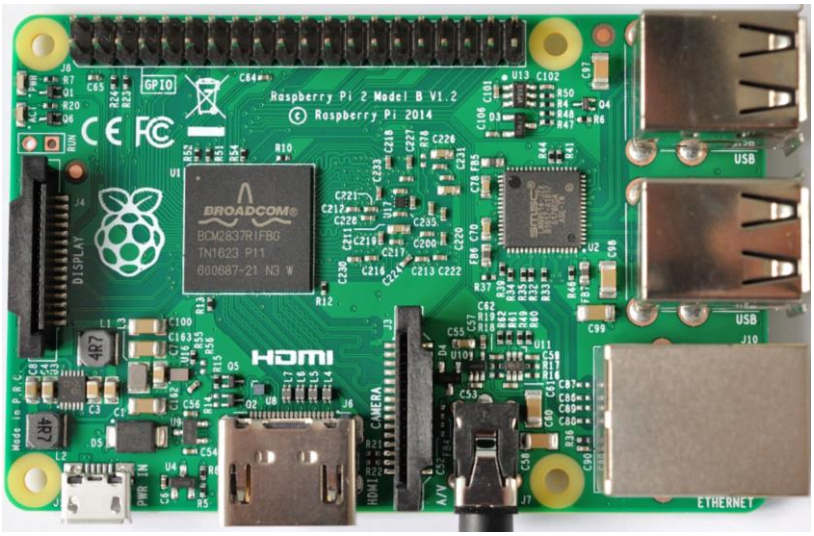

Figure 5: Raspberry Pi 2b

	Raspberry Pi 2 model B

CPU	ARM11 ARMv7 ARM Cortex-A7 900 MHz
RAM	1 GB
Data storage	microSD
Consumption	5v, 900mA
GPIO Connectors	40-pins expansion header (2x20 strip) Pin description: ● 2x5 Volts ● 2x3.3 Volst ● 4xGND ● 1xSDA ● 1xSCL ● 1xTXD ● 1xRXD ● 2xSCLK ● 2xMOSI ● 2xMISO ● 3xCE ● 1xID_SC ● 1xID_SD ● 26xGPIO (some of them have share some of the before functionalities) Pin Reference: [15] [16]
Other Connectors	HDMI, 10/100 BaseT Ethernet Socket, 4xUSB 2.0

Table 6. Raspberry Pi 2b characteristics

4.3.3. LoRa card

The reason of the chose for the Libelium SX1272 as the LoRa card was, mainly because, it has a low power consumption, and that is the point of using LoRa as way of communication. But also because of the connection bridges that can be used for Arduino and/or Raspberry which fits with their interfaces. More details in the analysis document[2].

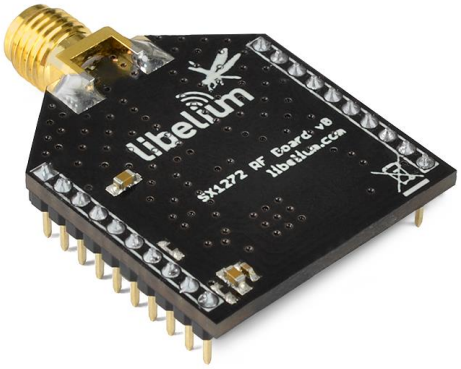

Figure 6: Libelium SX1272

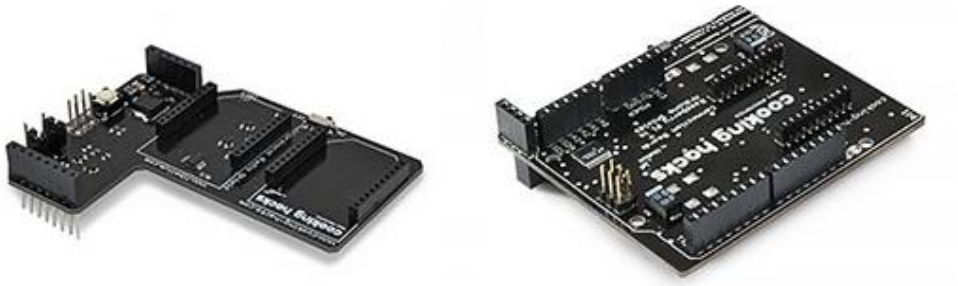

Figure 7 & 8: Multiprotocol shield. ArduPi platform for Raspberry Pi.

	Libelium LoRa 868
Frequency range	868 MHz and 900 MHz ISM frequency bands
Tx-Power	Up to +14 dBm
Chipset consumption (Transmitting data)	35 mA

Sensitivity	down to -134 dBm
Transceiver chip	SX1272
Adapter/Socket compatible	YES/YES
References	[14]

Table 7. Libelium SX1272 characteristics

4.3.4. Humidity and Temperature sensor

The DHT22 has been the selected sensor for the temperature and the humidity. As it is a sensor which can measure both temperature and humidity it reduces costs. Another important reason is that it it is low power consuming, and it is really remarkable as the main thing is to develop a low power consuming node. More details in the analysis document [2].

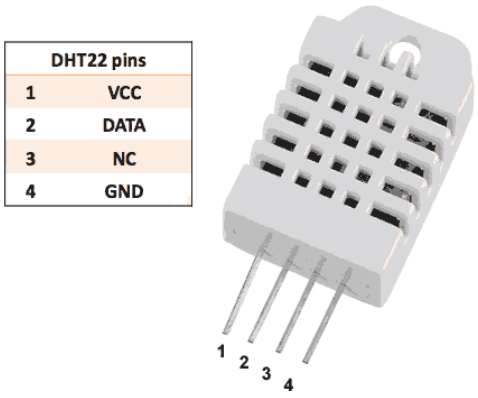

Figure 9. DHT22

	Temperature	**Humidity**

Measurement range	-40°C to 80 °C	0 - 100 %
Measurement accuracy	<±0.5 °C	±2 %
Resolution or sensitivity	0.1 ºC	0.1 %
Current Consumption	1.5 mA	1.5 mA
Supply Voltage	3.3 ~ 6 V DC	3.3 ~ 6 V DC
Communication Interface	Serial data	Serial data
References	[19]	[19]

Table 8. DHT22 characteristics

4.3.5. Pressure sensor

The BMP280 has been the selected pressure sensor because of its low power consumption. Another important reasons to take in count are the large measuring range (see table below) and the price. More details in the analysis document [2].

Figure 10. BMP280

	BMP280
Measuring range	300 - 1100 hPa
Supply voltage	1.71 ~ 4.25 V
Maximum pressure	20000 hPa
Supply current	2.7 µA
Resolution	0.16 Pa
Pressure accuracy	±1 hPa
Communication interface	I2C bus and SPI
References	[20]

Table 9. BMP280 characteristics

4.3.6. Accelerometer

The accelerometer sensor selected for the project has been the MPU6050 because of its precision in order to satisfy the project description constraints[1]. As the node has to measure with +/-5 cm precision. Also, another important characteristic is that the power consumption can be reduced to 10uA using the low power mode. More details in the analysis document[2].

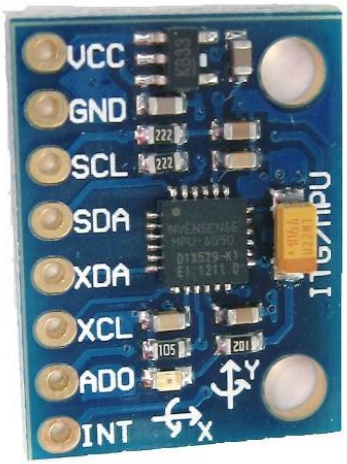

Figure 11. MPU6050.

	MPU6050
Input voltage	3.3 / 5V
Cross-axis Sensitivity	+/- 2%
Output resolution	16 bit (two's complement format)
Supply current	500 uA (Accelerometer only) 10 uA (Accelerometer low power mode at 1.25Hz rate)
Communication interface	I2C with an INT(interrupt)
Datasheet	[21]

Table10. MPU6050 characteristics

4.3.7. Programming language

4.3.7.1. Gateway

Raspberry platform was programmed in C++ mainly, but some features regarding to the database are programmed in Python.

4.3.7.2. Node

The node is programmed mostly with the Arduino IDE, so the language used is the arduino IDE language which is a modified C++. But some features are programmed with the Atmel Studio in C++ language.

4.4. Architecture and design

4.4.1. Hardware

In this section is described a brief summary about the hardware of this project, but it it is explained with more details in the Hardware architecture and design document [4].

4.4.1.1. Hardware architecture

SysML language has been used in order to explain how the parts interact with each others. First of all, the Block Definition Diagram to define the system and place it with the outside parameters. Explicative table of the BDD in the Hardware architecture and design document [4].

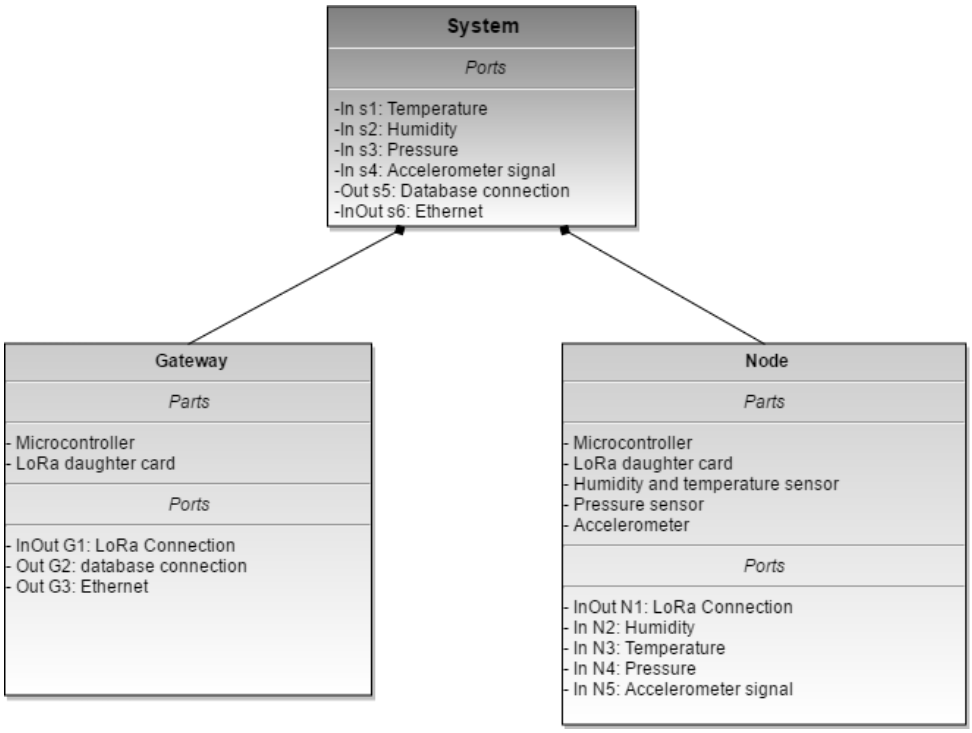

Figure 12. BDD of the system

The system is divided in two parts, the node and the gateway. Its interfaces with the outside are the temperature, humidity, pressure and relative position (measurements to take), as IN ports, as an OUT port it has the database connection, and, finally, as an INOUT port the Ethernet.

Also, an Internal Block Diagram (IBD) has made in order to show how every part of the system is connected. Explicative table of the IBD in the Hardware architecture and design document [4].

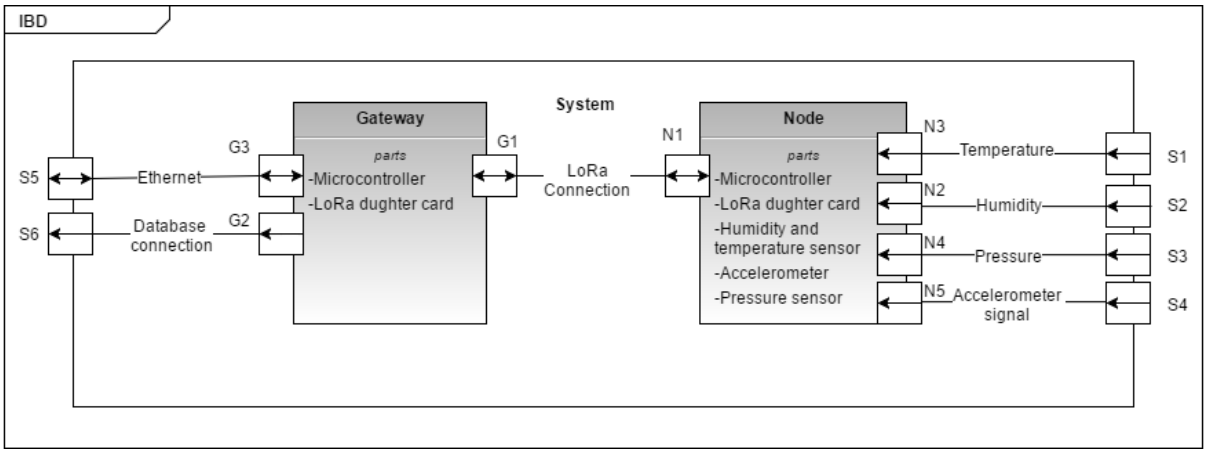

Figure 13. IBD of the system

The node and the gateway are connected each other via LoRa connection. The measurements signals go to the node and the ethernet and database connection to the gateway.

4.4.1.2. . Hardware design

All the design of the hardware has been done using Fritzing, not only for the schemes but also for the PCB design. Detailed description in the Hardware architecture and design document [4].

4.4.1.2.1 Gateway

The gateway is connected by interfaces are connected with the ArduPi bridge, and it is connected at the same time with the LoRa card.

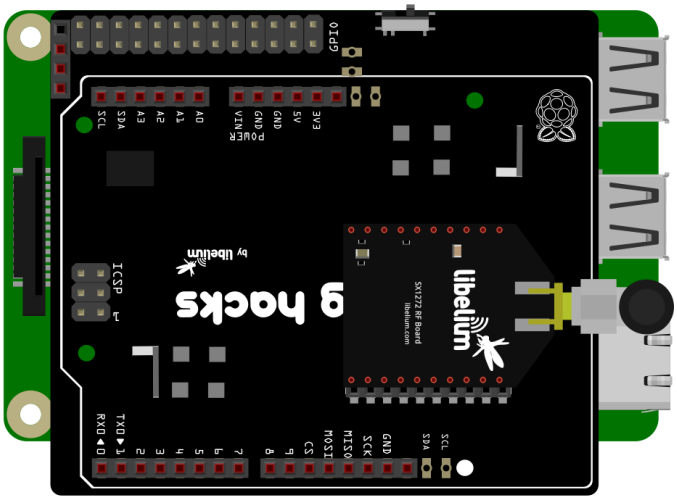

Figure 14. Graphic diagram of raspberry Pi with LoRa card.

Basically, the ArduPi bridge, acts as an intermediary between the LoRa card and the Raspberry platform. And it uses the following interfaces:

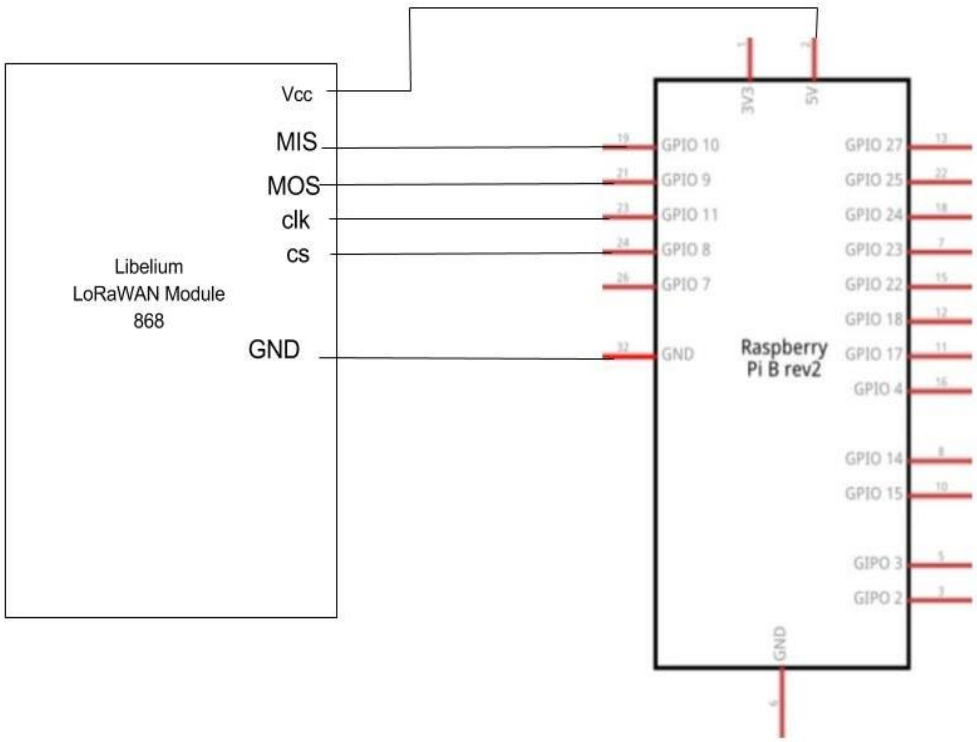

Figure 15. Schematic Raspberry Pi & LoRa card.

4.4.1.2.2 Node

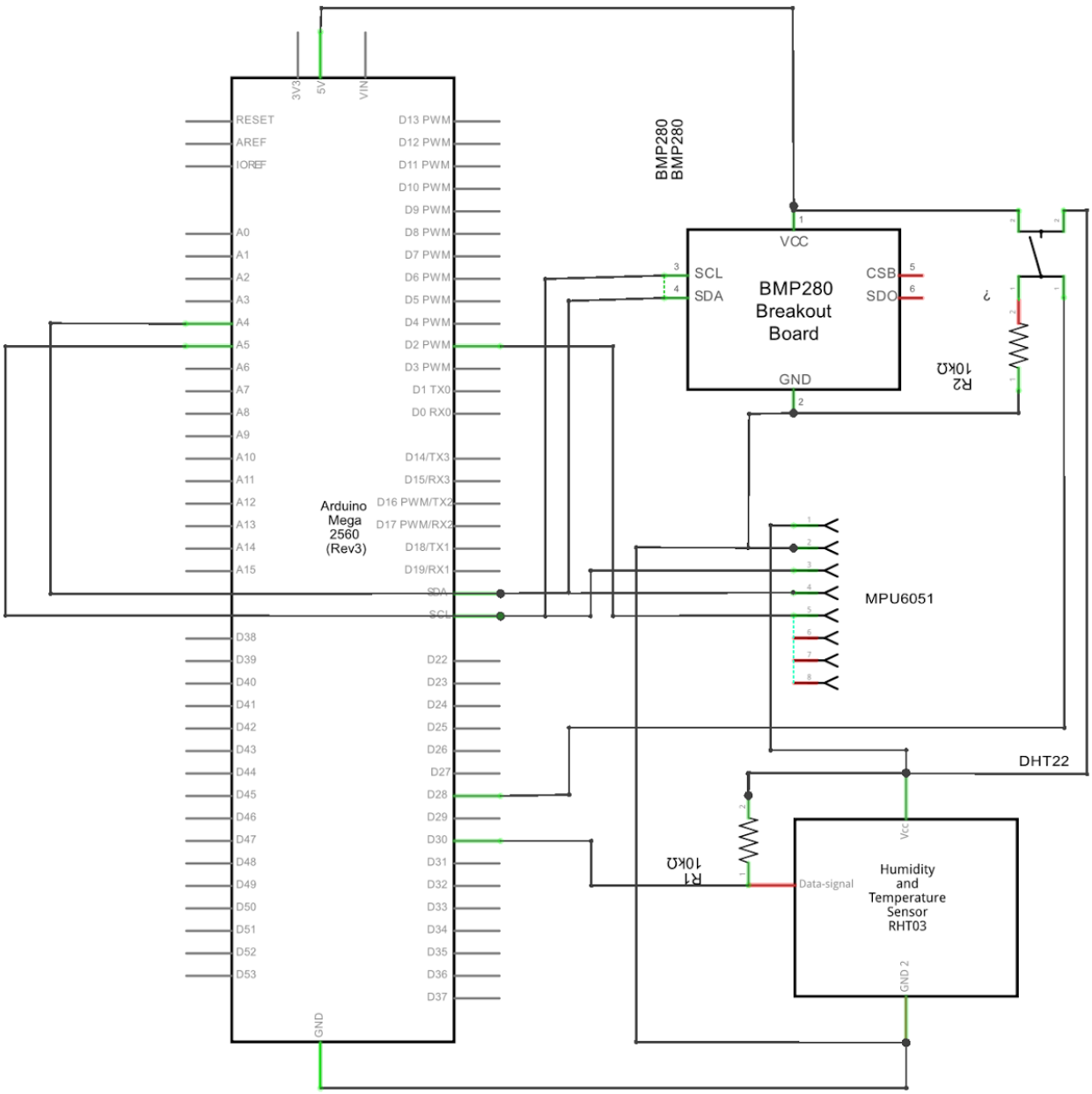

Figure 16. Schematic of the node

On the scheme above can be seen the schematic design of the node. It consists in the Arduino platform, the three sensors, one pull-up resistor, one pull-down resistor and a button.

Some of the remarkable features of the design are that all sensors are connected to the 5V supply but can also be connected at 3.3V(as they are prepared for that), the pressure sensor and the

accelerometer also share the i2c interfaces. A pull-up resistor of 10k is connected between the power supply and the data signal as recommended by some distributors[8], also a capacitor of 100nF for wave filtering can be added between power supply and ground, as the manufacturer states[19]. Something to remark is the button, where pull-down resistance makes the logic input to be zero while it is LOW state. All the details about the interfaces are explained in the Hardware architecture and design document [4].

4.4.2. Software

4.4.2.1. Method used

The method used to describe the software has been de N+1 model, talking about the 5+1 views: Logical, Process, Data, Deployment, Implementation and the Use Case view.

4.4.2.2. Logical view

In the logical view is where the the behaviour of the project is described, based on the use cases. It is divided in two distinguished parts: the gateway and the node, and both described in the same way.

First of all, the empty classes needed for each use case are given, then the sequence diagram is shown and, finally, a static class diagram.

4.4.2.3. Gateway architecture

After analysing the use case for gateway, here is the static diagram for entire gateway.

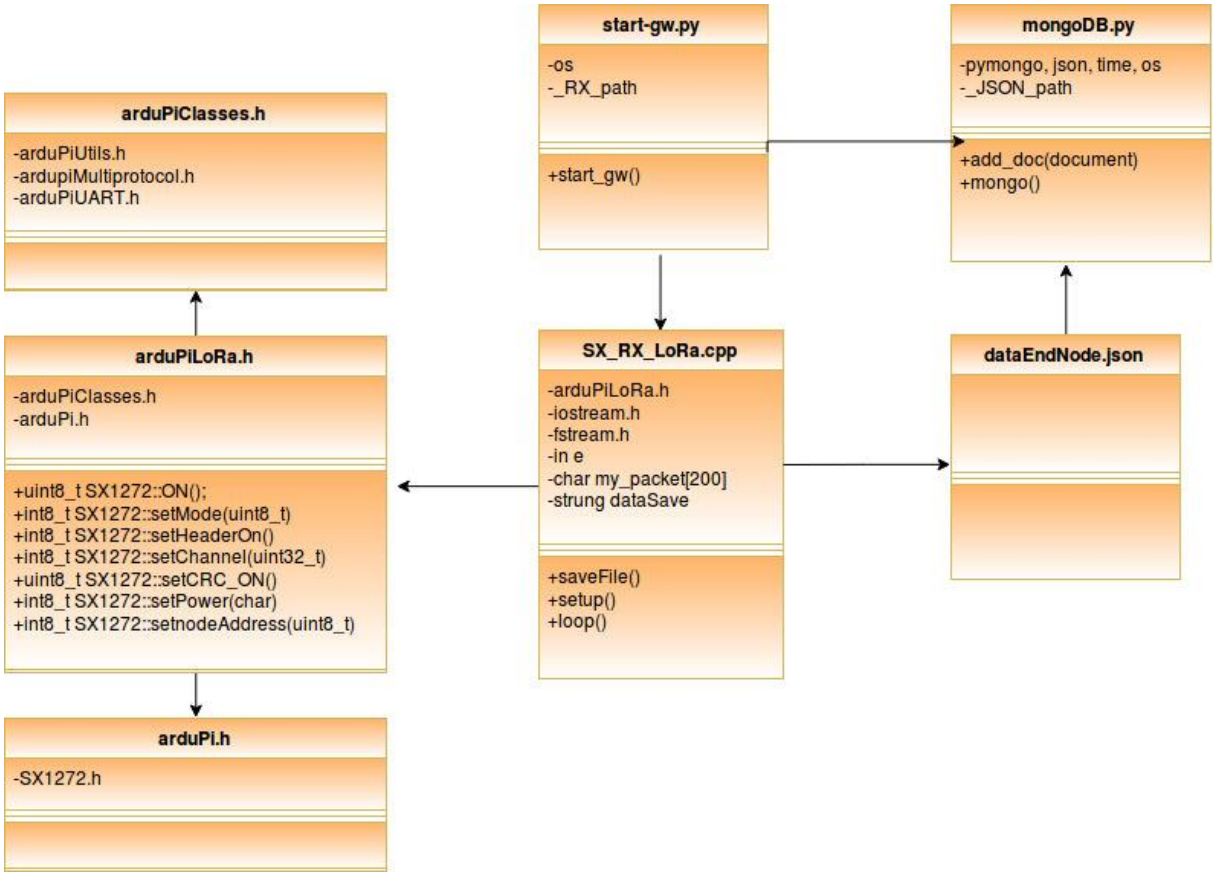

Figure 17. Static diagram gateway

The entire descriptions of those function are in SW document [5].

4.4.2.4. Node architecture

After analysing all the use cases in the end node, here it is the static diagram for the whole node.

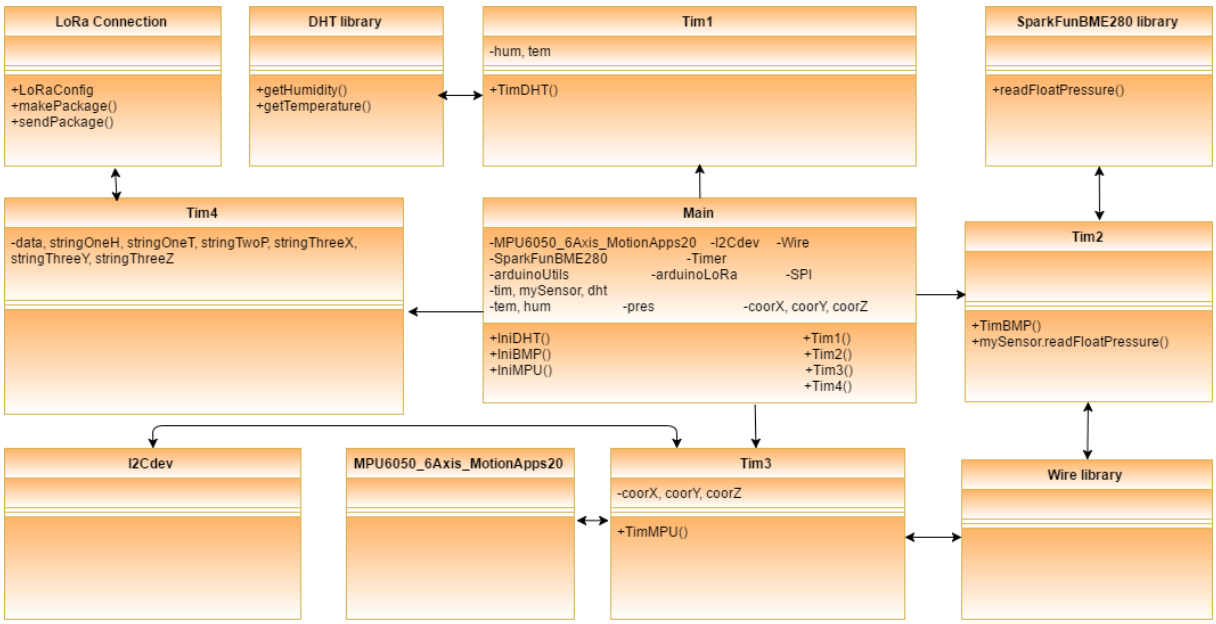

Figure 18. Static diagram End Node

4.4.2.5. Process view

Once every component is explained, it was detailed the processes inside of these project's components and then, how each process between node and gateway interact for exchange data. End node sends data from sensors which is received in the gateway for storing in database.

4.4.2.6. Data view

Storing data at mongo database is done by the two functions defined at mongoDB.py script which is developed in the project. To store this data, the system must create a JSON for this purpose. MongoDB works with BSON object (a Binary JSON object) but the commands of mongoDB demand a JSON structure and the mongoDB's librerie create this BSON from a JSON object automatically. (For more information about the command uses at mongoDB.py, see [5])

4.4.2.7. Deployment view

Then, the technology of wireless was setted, we analysed what will be the productive chips for our needs to establish the LoRa communication. We realized the sx1272 LoRa chip will complete our

objective to communicate both microcontrollers. One we had all components that we needed, the fact to create a programmer's view about how everything will be linked and how these components are going to work together as a system was simple. This below picture explains graphically how the system is linked.

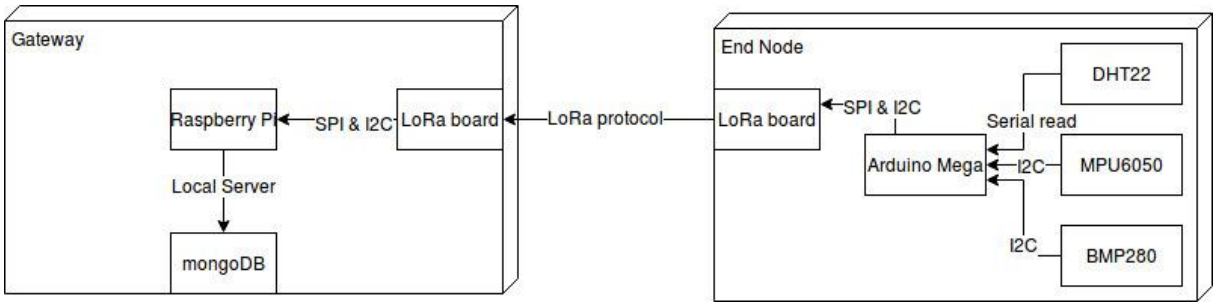

Figure 19. Schematic of whole system.

4.4.2.8. Implementation view

The implementation view describes which tools are used to develop this project. The tools are:

-Arduino IDE:

To program the Arduino Microcontroller, it will be used Arduino IDE.

-AtmelStudio:

To modify libraries of the different sensors.

-IDLE:

To program on Raspberry Pi. This IDE is pre-installed on raspbian OS.

-Terminal:

To check and control all actions of Raspberry Pi and mongoDB.

-Nano:

To support IDLE on Raspberry Pi. It is a basic Editor text.

4.5. Implementation

In this section the assembly of this project will be explained part by part.

4.5.1. Gateway

The implementation of the gateway is the one with less cost because the boards selected fits in the interfaces and no wiring is required.

The main part is the microcontroller, so that, the board in which everything is going to be plugged is the Raspberry Pi.

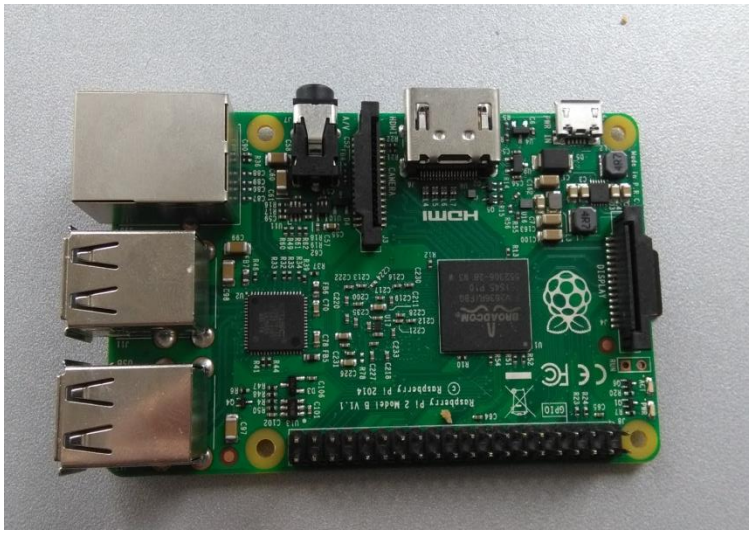

Figure 20. Raspberry Pi.

In the Raspberry Pi, the Raspberry Pi to Arduino Bridge is placed in and, the Raspberry Pi is also connected to the power.

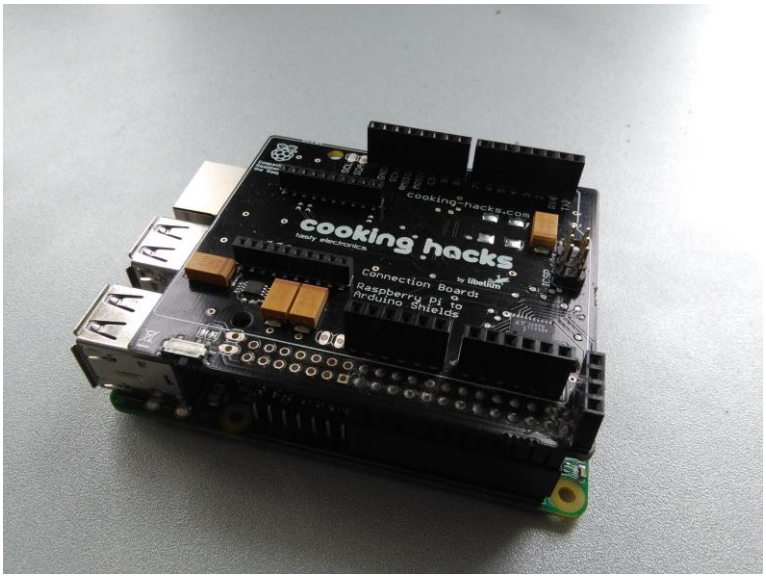

Figure 21. Raspberry Pi with the bridge

Finally, the LoRa daughter card is plugged in the Xbee socket of the shield and the antenna in the LoRa card.

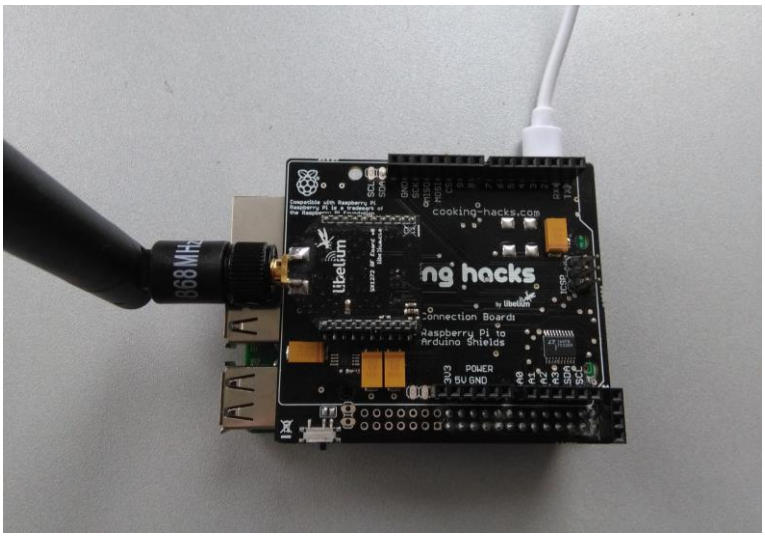

Figure 22. Gateway completely assembled

4.5.2. Node

The implementation of the node is similar with the gateway, but has some extra steps. As in the previous assembly, the main part is the microcontroller board, so, the Arduino Mega 2560. The PCB has to be plugged in, but in this case the lack of time has not made possible to develop this pcb. Instead of that two Veroboards has been developed for the interfaces of the sensors.

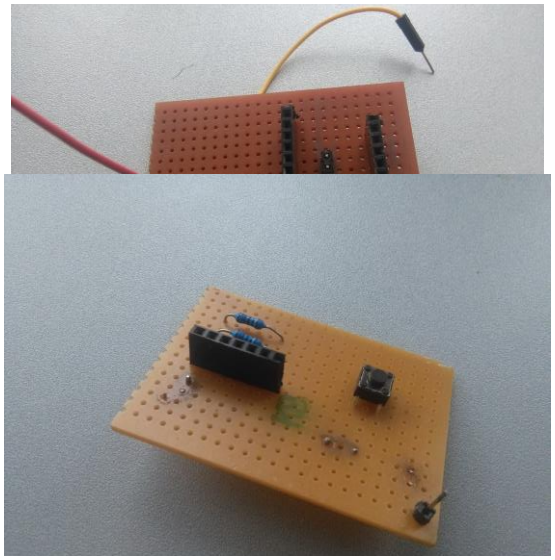

Figure 23 & 24. Veroboards developed

After developing the Veroboards, these ones have to be placed, as well as the multiprotocol board in the Arduino.

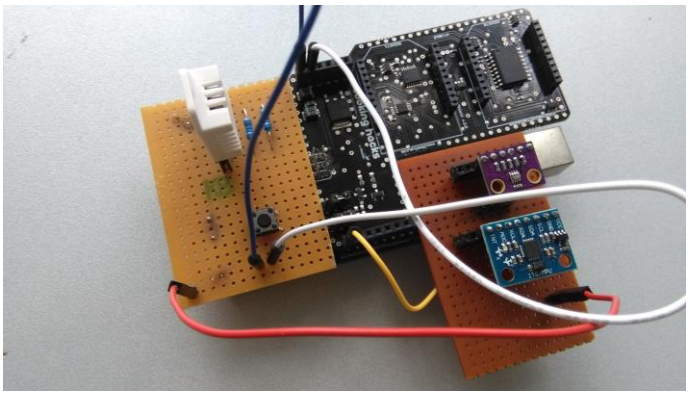

Figure 25. Boards plugged in the Arduino platform

Finally, the LoRa daughter card is plugged in the Xbee socket of the shield and the antenna in the LoRa card.

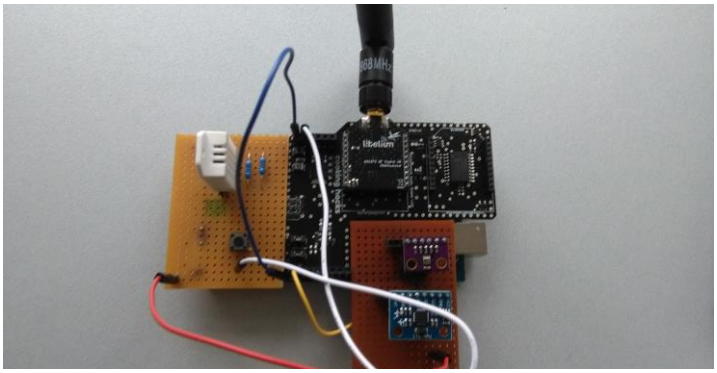

Figure 26. Node completely assembled

4.6. Test

In this section the procedure and the results of test are explained, but for further details about it, the Acceptance test document can be checked [6].

4.6.1. Unit test

A breadboard has been used for the unit test because the Veroboards had also to be tested and its functioning can't affect the units test, if not, the test is not not valid.

4.6.1.1. Humidity and temperature sensor

The test for the DHT22 consist in connecting the sensor in the microcontroller and compile the part of code which makes the DHT22 work. Then, the collected data is shown by the Serial monitor.

4.6.1.2. Pressure sensor

The test in the pressure sensor consist in the same as the previous one, but with the detail that the BMP280 is using I2C connection and two sensors are using I2C. So different addresses has been set in each sensor. Then, again, the collected data is shown by the Serial monitor.

4.6.1.3. Accelerometer

The test in the accelerometer is the same as in the pressure sensor, but this time, logically, with the part of code that corresponds to the accelerometer. Then, the Serial monitor shows the acquired data.

4.6.1.4. Libelium and bridge in the node

In this part the libelium card with the board has been tested using the libraries and tests already made by the seller and the manufacturer [9]. During this test has been found some issues, and it has lead to some hardware adaptations that are explained in the Acceptance test document[6].

4.6.1.5. Libelium and bridge in the gateway

As in the previous test, it has been used libraries and tests from the manufacturer. During this test there has also found some troubles, but they had been solved without any hardware affectation.

4.6.2. 5.6.2 Implementation test

4.6.2.1. Fully functioning node

The implementation of the full node has been done using the whole code that involves all the sensors and the LoRa card, and by the results displayed on the Serial Monitor, check if each part of the nodes working.

4.6.2.2. LoRa communication test

The LoRa communication test has been made with the tests of the manufacturer[9], as the Unit tests of the LoRa cards. So, the Raspberry Pi is configured as receiver and the Arduino is configured as transmitter and the Serial monitor displays what is going on.

4.6.2.3. Database test

The database test has been made with saving a false JSON object and uploading after with the mongoDB.py script. The screenshots about the logs of mongo database are at the Acceptance test document [6].

4.6.2.4. Veroboard

The test of the veroboard has consist in connecting the alligator clips in every wiring made, and these alligator clips, at the same time, are connected to the multimeter. Then, by the values showed in the multimeter's screen, it can be seen if it is a short circuit or open circuit [6].

4.6.3. Acceptance test

The procedure followed to do this test has been analysing step by step every use case of the requirements specification.

Use case number	Systems involved	Result
1	Collect temperature and humidity	OK
2	Collect pressure	OK
3	Collect relative position	NO
4	Gather data to be send	OK
5	Storing data in the database	OK

Table 11: Acceptance test

OK: The test is successful
NO: The test is not successful

For further details the Acceptance test document can be checked [6]

5. Results and discussion

The acceptance test shows which of the use cases completed its function and if the system is working how it's supposed to. The system is divided in two subsystems. One part is the End Node is composed with three sensors that are going to provide the data to process. The other part is the Gateway what will get the data to store it in a database.

The test results are made to express if every use case and every part in the use cases is working as it was expected and explained in the requirement specifications. This end node use cases involve the temperature and humidity sensor, the pressure sensor, the accelerometer, timers and obviously wireless LoRa communication. Most parts of the end node are working correctly as it was desired. By the other hand, gateway is complying with the expectations that we had. So, all use cases fulfilled expect of one.

The only part of one use case what is not fulfilling what it was expected is the accelerometer. The accelerometer is working by itself when it is not implemented with the rest of the components in the same sketch in the end node. In this research of the error in the code, we figured out that is failing in one function, anyway one possibility could be that it is failing in the library. The other possibility could be that is failing because somehow the value of the variable what have to be to enter in this function is not getting it. That's the mistake that is not solved yet, despite of this, we have completed most of the points of the project description, being a new technology as LoRa and the challenge that means.

However, in the project description was stated the possibility of a GPS signal, that wasn't a mandatory feature as explained in the requirements specifications[3], so it is a future work for this project. Also some requirements, in this case, mandatories, regarding to statistic information in every package like frequency and code rate were not added.

6. FUTURE work

With more time, do some improvements of this system could be possible. Some of these improvements are going to mentioned below:

- **Adding more end nodes:** LoRa can get an Star communication between various devices, implement this improvement could help for taking measurements from different points inside the same zone. (More information at [14])
- **Use a gps in the end node:** The user could get more information about where are the different end nodes.
- **Resolve problems with the accelerometer.**
- **Implements a control light panel:** The user could get basic static information about the correct functionality of the system without the need of connect a computer at full system.
- **Improve the power consumptions in the node:** some of the boards in the node can work in low power mode, so that, the power consumptions can be reduced.
- **Read and store the statics connection.** Like frequency, code rate, BW, SF and packet-RSSI, and payload information.
- **Automatic root for all system.** Linking the execution of the gateway at root system of Raspberry Pi gives an easy implementation for each kind of users
- **Give an internet connection.** Using internet, the gateway could upload its local database on internet and check the information everywhere.

CONCLUSION

This team has worked during one semester in order to create a smart sensing system in the internet of things environment, with the aim of creating a low-consuming system. This project had some previous objectives and ambitions, which some of them are solved and some of them not.

The main unsuccessful attempt has been that the previous idea with the communication has been LoRaWAN instead of LoRa, but during the development of the project there has been an issue and the lack of time didn't let us to make it possible. During the analysis we selected the RN2483 chip, that could only work as a node but not as a gateway. This missundistanig came due to our lack of previous knowledge wireless communications we confused the communication between by radio P2P instead of LoRaWAN, so it was not a gateway[10].

Then we made some searching for implementing the LoRaWAN as a gateway with the RN2483 and we couldn't find anything, only comments that it can't be used as a gateway and that the correct chip to make a LoRaWAN gateway is the sx1301(which is a baseband chip, that was what we wanted to develop).

Searching in some forums we found that it was suitable as a gateway[11][12] or some comments that the only way is changing the firmware or adapting the stack[13], which obviously was too much of our knowledge.

So that, we decided to give up with the LoRaWAN and try out with LoRa communication, using the SX1272 chip that also fits in our boards.

Another issue, was the PCB printing, in order to make the project smarter it would has been a good point to achieve, but the last changes in the design during the tests[6] didn't allow us to do it on time. However, the project is a prototype and the design is made, so the Veroboard is enough to show the result. Also, it is important to mark that we achieved to design a PCB for first time, that was one important objective reached for this project.

Also, some of the requirements stated in the requirements specifications are not totally accomplished, although most of them are ready to be implemented.

About LoRa/LoRaWAN communication, we have not reached the level of knowledge that we expected. It is a huge world, and it take a lot of time to study for arriving a high level of knowledge.

The most important objective achieved was to increase our skills in the Arduino and Raspberry Pi boards. In the case of Arduino platforms we had previously worked with them, but never with an

Arduino Mega 2560. During the project we had to work with this board, so our comprehension of the microcontroller and its interfaces has increased, and we are very satisfied with the result.

About the Raspberry Pi, it was the first time we worked with it, and it has also been an objective we accomplished the result. This board is really different from the Arduino, as it is not only a microcontroller, but a computer. The understanding of it functioning and its programming has taken some time but it is an objective accomplished successfully, although it is still a small proportion in comparing with the feature it provides. But the result is really satisfactory.

In conclusion, our acknowledgements have increased in a satisfactory way, improving most of the skills we wanted to achieve. About the project itself, there is still work to do in order to improve it.

References

[1]Project description, *Carlos González, Jordi Calvo and Jose Antonio Gil*
[2]Analysis, *Carlos González, Jordi Calvo and Jose Antonio Gil*
[3]Requirements specifications, *Carlos González, Jordi Calvo and Jose Antonio Gil*
[4]Hardware architecture and design, *Carlos González, Jordi Calvo and Jose Antonio Gil*
[5]Software architecture and design, *Carlos González, Jordi Calvo and Jose Antonio Gil*
[6]Acceptance Test, *Carlos González, Jordi Calvo and Jose Antonio Gil*
[6]About LoRa. *Semtch* [online] http://www.semtech.com/wireless-rf/internet-of-things/what-is-lora/
[7]LoRa alliance homepage, *LoRa aliance* [online] https://www.lora-alliance.org
[8]https://learn.adafruit.com/dht-humidity-sensing-on-raspberry-pi-with-gdocs-logging
[9] sx1272 tutorial, *cooking hacks* [online] https://www.cooking-hacks.com/documentation/tutorials/extreme-range-lora-sx1272-module-shield-arduino-raspberry-pi-intel-galileo/
[10] LoRaWAN tutorial, *cooking hacks* [online] https://www.cooking-hacks.com/documentation/tutorials/lorawan-for-arduino-raspberry-pi-waspmote-868-900-915-433-mhz/
[11] Cooking hacks forum, *cooking hacks* [online] https://www.cooking-hacks.com/forum/viewtopic.php?f=37&t=8799
[12] LoRaWAN RN2483 chip forum, *microchip forum* [online] http://www.microchip.com/forums/m917513.aspx
[13] Discussion baout RN2483 chip, *things network forum* [online] https://www.thethingsnetwork.org/forum/t/single-channel-gateway-with-rn2483/1848
[14]Libelium 868 guide, *libelium* [online] http://www.libelium.com/downloads/documentation/waspmote_lora_868mhz_915mhz_sx1272_networking_guide.pdf
[15]Raspberry pi pinout, *Raspberry* [online] https://pinout.xyz/#
[16] Raspberry gpio, *Raspberry* [online] https://www.raspberrypi.org/documentation/usage/gpio-plus-and-raspi2/
[17] Arduino mega2560 pinout, *Arduino* [online] https://www.arduino.cc/en/Hacking/PinMapping2560
[18] Picture Arduino Mega, *robotistan* [online] http://www.robotistan.com/Data/EditorFiles/Arduino/mega/tumblr_mjm0bml7441s5t695o1_1280.png
[19]DHT22 datasheet [online] https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf
[20]BMP280 datasheet [online] https://cdn-shop.adafruit.com/datasheets/BST-BMP280-DS001-11.pdf
[21]MPU6000 datasheet [online] https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
[22]Project schedule, *Carlos González, Jordi Calvo and Jose Antonio Gil*