
SOCIAL NETWORK FOR POLLUTION

Project Report – ISS – 2019/2020



Tutor

Thierry Monteil

Students

Franck Bourzat – Krishna Sujan Inuganti

Claire Gageot – Yuheng Jin

Ismail Touzani – Théo Zanchi

We want to thank our tutor, Thierry Monteil for his help and availability during the whole project and also François Aissaoui, Guillaume Garzonne and Nicolas Verstaëvel for their help and answers to our questions.

SOCIAL NETWORK FOR POLLUTION

THÉO ZANCHI, OCTOBER 2019 – JANUARY 2020

KEY-WORDS: POLLUTION, SOCIAL NETWORK, LORA, OM2M, TTN,
MONITORING, PARAMETERS

Monitor pollution is growing and will become a major concern along with the need to provide quantified information about pollution to people. We want to provide a transparent way to user to be a part of this challenge and be at the centre of this monitoring. Our project aims to give people the opportunity to access and know information about the quality of their surrounding environment.

This goal can be achieved by building low-cost platforms, able to monitor pollution parameters and at a specific location (district, city...). The platform being low-cost, we can distribute it to a large amount of person and create a kind of social network of the pollution monitoring and display the air quality data collected as a map.

The solution is composed by four main parts. First, the platform which is interconnected with GPS receiver, some gas sensors: CO, O₃, SO₂, NO₂ and more common sensors like temperature, humidity and pressure. This platform will send the measures via the LoRa protocol to a worldwide LoRaWAN shared network. The third part is composed with an OM2M server running along with a NodeJs server. We then develop a mobile application to fetch data from the NodeJs server and displays them to the user as a map.

At the end of the project, we produced a functional prototype of the platform. Our IOT product is able to monitor and send measures about pollution to our TTN application. Our OM2M server receives this data and store it. Users are able to access this data via a React-Native mobile application communicating with a NodeJs server to access OM2M data and present them in the form of a map.

Toulouse City Council wants us to present them the project. Further work is needed to complete the project. Indeed some part about security, user database and server optimization and work on the hardware part could be required.

Table of content

Introduction.....	7
Context	8
1. Overview.....	9
1.1. Client.....	9
1.2. Specifications.....	9
1.2.1. Low-cost	9
1.2.2. Scalable.....	9
1.2.3. Easy to use.....	9
1.3. General architecture	10
2. Connected platform	11
2.1. Study of pollution parameters	11
2.2. Platform data acquisition	11
2.2.1. Introduction.....	12
2.2.2. Requirement analysis	12
2.2.3. List of sensors	12
2.2.4. Data acquisition architecture	13
2.2.5. Data acquisition implementation.....	14
2.2.6. Conclusion	16
2.3. Networking part	17
2.3.1. Choice of the wireless technology.....	17
2.3.2. Hardware choice.....	18
2.3.3. Programming the Hardware.....	18
2.3.4. Over the air activation and data sending	19
3. Server Architecture	22
3.1. Infrastructure part.....	22
3.1.1. Setting up of OM2M server online.....	22
3.1.2. Setting up the Node.js server	23
3.2. The Thing Network	24
3.2.1. A shared LoRaWAN network	24

3.2.2.	Choice of TTN	25
3.2.3.	Applications and devices	25
3.2.4.	Data pre-processing.....	26
3.3.	OM2M	26
3.3.1.	oneM2M standard and OM2M platform	26
3.3.2.	Choice of OM2M	27
3.3.3.	OM2M architecture.....	28
3.3.4.	IPE for TTN	30
3.4.	Node.js back-end server.....	32
3.4.1.	Choice of Node.js.....	32
3.4.2.	REST API.....	33
3.5.	User Database	34
3.5.1.	Choice of MongoDB.....	35
3.5.2.	Data structure.....	35
3.6.	Limits and future implementations.....	36
4.	User application.....	37
4.1.	Choice of Mobile application.....	37
4.2.	Choice of React-Native	37
4.3.	Applications Screens.....	38
4.3.1.	Home Screen	38
4.3.2.	Map Screen.....	38
4.3.3.	Login Screen	38
4.3.4.	Registration Screen.....	38
4.4.	Google Map API.....	39
4.5.	User and platform management.....	40
4.5.1.	User registering process	40
4.5.2.	Platform registration process.....	40
4.6.	Future implementations.....	40
5.	Business plan	41
5.1.	Introduction.....	41
5.2.2.	The market	42
5.2.4.	Positioning.....	43

5.3.	Pricing model.....	44
5.3.1.	Business to Client (B2C).....	44
5.3.2.	Business to Business (B2B)	45
5.4.	Acceptability	46
5.5.	Funding.....	46
6.	Project management.....	48
6.1.	Project work breakdown structure	48
6.2.	Team.....	49
6.3.	Project management method	49
6.4.	Project management tools	50
7.	Results	51
8.	Conclusion	52
	References.....	53
	Annex 1: Franck's abstract	54
	Annex 2: Ismail's abstract.....	55
	Annex 3: Krishna's abstract	56
	Annex 4: Mobile application screens	57

Introduction

We count more than 593 million of connected objects in the world. As we are beginning this new 2020 years, this number will probably double by the end of the year. We can observe this rapid growth very easily. In the health, wearables, monitoring domains, the number of connected objects sold is exploding. This phenomenon is raising some questions about privacy, security, reliability which can brake on the IOT development. Despite all these questions and issues, we cannot denial that this domain is in constant growth.

Along this development of the IOT and connected objects, people get more and more aware of concerns related to the environment, the impact of our actions on the planet, the pollution levels and the way in which their impact our health. As a matter of fact, there are about 250 million litres going through our lungs during a lifetime. Furthermore, estimations say that the bad air quality is responsible for 7 million of deaths every year. This impact of the pollution on our health bring people to be more aware and concerned about these topics. More and more articles and information are talking about all these concerns. Since pollution and bad air quality have a really bad effect on people health, a big economical is raised due to the care of diseases and injuries implied by that. Estimations say that this cost for the European Union in 2016 was about 66,7 billion euros specially due to the fine particles and other pollutant emits by transports and industry. This is why today, we observe that ecology, pollution reduction and monitoring of air quality take a larger space in all the decisions taken.

Monitor pollution and air quality is thus growing and will become a major concern along with the need to provide quantified information about pollution to people. We want to provide a transparent way to the user to be a part of this challenge and be at the centre of this monitoring. Our project aims to fulfil that need of transparency and thus, give people the opportunity to access and know information about the quality of their surrounding environment.

Social Network for Pollution is a project which include itself in this race of monitoring. The idea is to imply people into these concerns of pollution reduction and ecology by giving them the tools to quantify, observe and analyse the impact of human activities, especially in cities on the air quality. The main objective of our project is not to directly reduce pollution and have an action on people's behaviour. Gather people around common concerns and use social and group influence may be better and more efficient, to get people to change their habits and behaviours.

Context

The Social Network for Pollution project is mainly destined to urban zones, cities and municipalities. It includes itself into the current development of smart cities. This notion of smart and numeric cities comes with the current growth of urbanization. As a matter of fact, we now that a large amount of population lives in cities but estimations say that this urbanization could reach 70% of the population by 2040. This massive migration of people is motivated by several reasons. However, it also brings some challenges relative to many domains: health, education, food, water, energy consumption, security and ecology. The smart city as it is thought right now as to give answers and solutions to respond to these challenges.

Several characteristics may be respected by a city to be considered as a smart city, some of them are the durability, the environmental-friendly and the participative ones. The Social Network for Pollution is designed to have an impact on these three points. The essence of the project is to help the pollution reduction and air quality improvement by giving people quantified measures and information and thus impact the environmental-friendly aspect. As the mean of the project to get quantified air quality information and create maps of these data is to gather people around a kind of social network, it also impacts the participative aspect. Finally, in this perspective to provide to cities and municipalities the opportunity to gather the citizens around these concerns of air quality, we could imagine that decisions, solutions and projects could be led and decided based on measures performed with the Social Network for Pollution solution.

To give to this project an even more precise context, we can take as an example the city of Toulouse. Since several years Toulouse tends to position itself into a durable green city development logic, by taking decisions aiming to reduce the impact of pollution. Toulouse count around 840 hectares of green space, around 153 000 trees and there are almost 4000 new trees planted each year.

Our project include itself into this context where cities tends to choose a durable and green decisions and where people want to be involved and have information about topics like pollution and air quality. The Social Network for Pollution project is about providing citizens low-cost air quality monitoring platforms and tools like mobile applications to visualize the data collected.

The platform has to monitor the most relevant air quality parameters and send them via radio telecommunication. This device is meant to be distributed to the largest amount of people to collect the most diversified data and be able to produce maps of these data with the most coverage as possible.

1. Overview

1.1. Client

The Social Network for Pollution project was initiated and proposed by M.Monteil. During the whole project development, M.Monteil was both the Client and Tutor of the project.

1.2. Specifications

The project came along with some requirements. These specifications mainly deal with the acceptability of the project by the users but with the future implementations that will probably be performed to improve and complete the project.

1.2.1. Low-cost

In order to be distributed to the largest amount of people, the air quality monitoring platforms have to be the cheapest possible. All the hardware parts used to design the platforms have to be chosen to reduce the final cost.

1.2.2. Scalable

In the future, this project will probably be improved and enhanced by other features or the integration of other devices. As the number of devices growth, the architecture of the project has to be scalable to handle the growing number of devices and interoperable to handle the difference of the devices and protocols.

1.2.3. Easy to use

As long as the platforms have to be low-cost, the Social Network for Pollution project has also to be easy to use. This is an important feature to gather the large amount of users. The ideal would be a plug and play platforms which can be deployed directly by the user, with nearly any configuration steps.

1.3. General architecture

The architecture defined for the Social Network for Pollution project is represented on the following figure:

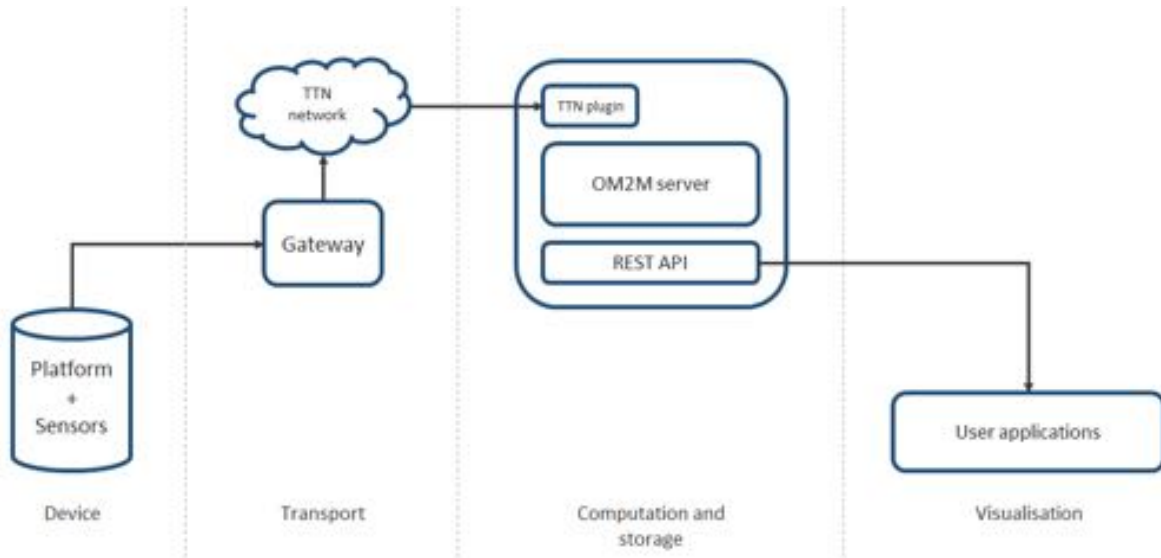


Figure 1 : General architecture of the project

The project is divided into several parts which are in interaction to be able to monitor, transmit, store and provide air quality measurements.

The first part is the connected platform. This part is composed of the LoPy4 board along with all the sensors. His part is to send air quality measurements via LoRa communication.

The second part is the transport part. This part is represented by the LoRa gateways receiving the data from a connected platform but also the network server provided by The Thing Network and is responsible for the data transport.

The third part is the server part composed by the OM2M server which implements the TTN IPE and the Node.js REST API. This part is responsible for the data storage, computation and providing

The final part is the visualization one, composed by the mobile user application responsible of the displaying of the data measured by the platforms.

2. Connected platform

2.1. Study of pollution parameters

To monitor pollution, we have decided to measure the concentration of physical parameters involved in pollution. We have defined the main ones to qualify air quality:

Fine particles (PM10 and PM2.5): These are less than 10 or 2.5 micrometres sized particles. Thanks to their extra-lightness, these particles are not submitted to the gravity action. That means they are constantly in suspension if there is not rain or aggregation to put them down. Due to their size, they can enter the lungs and get to the pulmonary alveoli which is quite dangerous and can cause several diseases and complications. Also they can be used as condensation core by other pollutant particles like sulphur dioxide and plumb.

Carbon monoxide (CO): Carbon monoxide is a toxic gas created during forest fire or combustion for example. This gas is odourless, colourless but highly inflammable. More than 35 ppm of this gas is toxic for human beings because it disturbs the process of oxygen carrying in the body.

Ozone (O₃): Not emitted directly into the air but resulting of the reaction between oxides of nitrogen and volatile organic compounds (VOC).

Nitrogen dioxide (NO₂): This Chemical compound is mostly used to produce fertilizers and is a red-brown toxic gas. Its toxicity comes from his ability to enter the epithelial lining fluid (ELF) and cause potential inflammation, immune system dysfunctions and also affect the heart.

Sulphur dioxide (SO₂): SO₂ is a toxic chemical compound which can easily react with other compound like sulphuric acid and create harmful components. We can recognise sulphur dioxide by its smell that looks like match smell. Inhale SO₂ can cause breathing complications like nose, throat, lungs irritations, shortness of breath...

Temperature and pressure: These parameters are not directly used to evaluate AQI. Sensors used to monitor AQI, especially the cheapest ones are highly affected by these three factors. So it could be interesting to get this information to in complement to the ones directly related to AQI.

2.2. Platform data acquisition

This section describes the collection of relevant data to our application in terms of gas, temperature and localisation. It's divided into six essential parts: Introduction, Requirement analysis, List of sensors, Data acquisition architecture, Data acquisition implementation, Conclusion.

2.2.1. Introduction

Depending on the study made in order to list the main responsible parameters for air pollution (as shown in the previous section), we used several sensors to get the correct data (gas sensors, temperature/pressure sensors, GPS). The data collection will be relevant for the next parts of the project.



2.2.2. Requirement analysis

The choice of the sensors was made after analysing the list of requirements below:

- Req_1: The sensor must be available.
- Req_2: The sensor must be able to communicate with the PyCom board.
- Req_3: The sensor must return the correct measurement of the corresponding unit.
- Req_4: The sensor should consume less power.
- Req_5: The sensor can detect more than one unit.

2.2.3. List of sensors

After analysing the list of requirements above, we were able to make a list of sensors depending on their availability. The figure below shows a list of sensors that we'll be using in order to collect data:

Sensor Reference	Description	Image
ULPSM-SO2 968-006	Ultra-Low Power Analog Sensor Module for Sulphur Dioxide.	
ULPSM-O3 968-046	Ultra-Low Power Analog Sensor Module for Ozone.	
MQ-9 GAS SENSOR	Sensor module for Carbon Monoxide.	

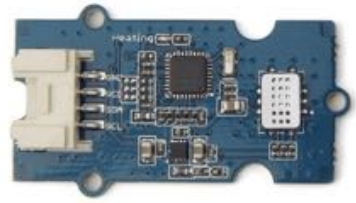
Grove Multichannel gas sensor	- gas Sensor used to detect Nitrogen Dioxide No2	
BMP180 sensor	Barometric Pressure/Temperature/Altitude Sensor.	
NEO-6m	GPS Module	

Figure 2 : Sensors

2.2.4. Data acquisition architecture

During this part, we'll expose both the high level and low level design of data acquisition. The High level design part describes the main architecture of platform data acquisition. The Low level design part describes the different modules of the MicroPython script for platform data acquisition.

a) High level design

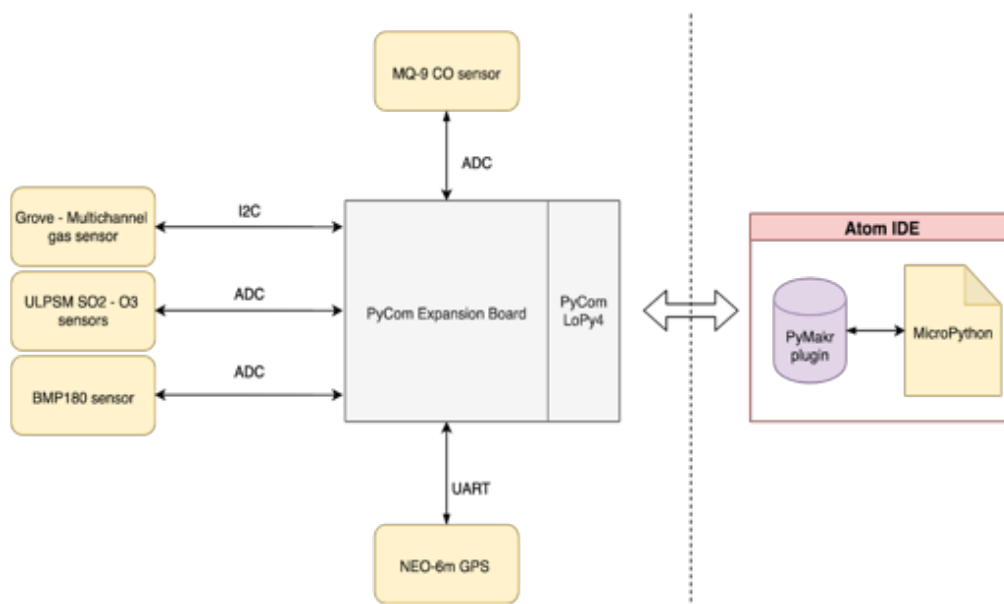


Figure 3 : High level design

As shown on the figure above, sensors are communicating with the PyCom expansion board using serial communication protocols such as I2C, ADC and UART. For the programming part, we use PyMakr plugin that is based on MicroPython and available for Atom IDE.

b) Low level design

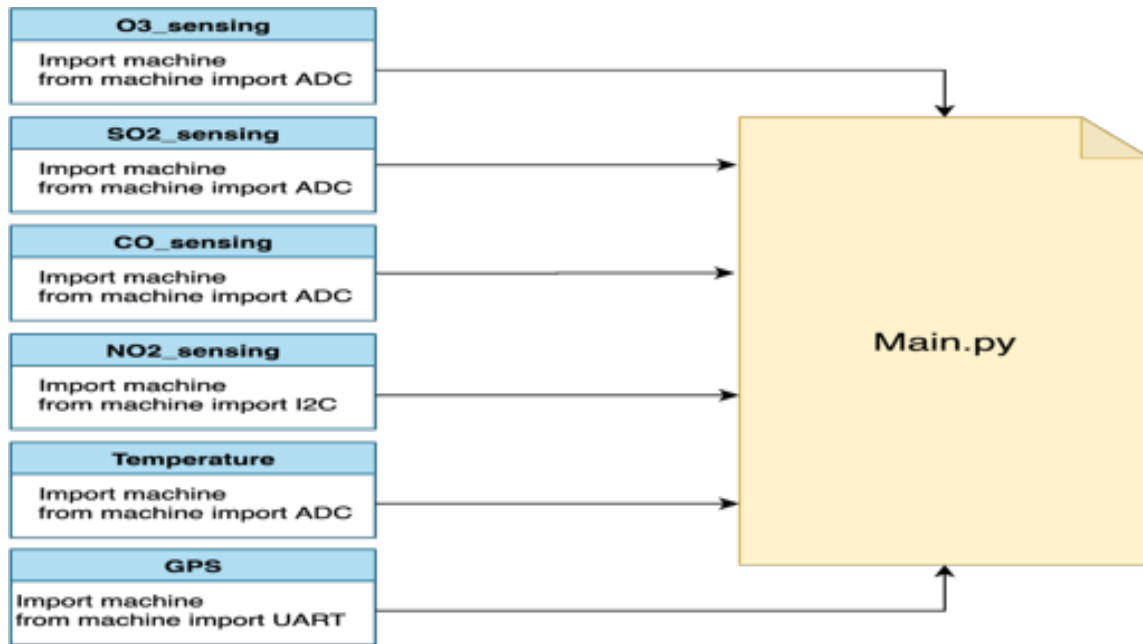


Figure 4 : Low level design

PyMakr plugin offers a module called machine where we can import different classes that will help us set up the serial communication with the hardware (I2C, ADC, UART). We've created a module for each sensor where it will be imported in the main script.

2.2.5. Data acquisition implementation

The implementation of collecting data from sensors requires a knowledge in some protocols and operations (e.g. GPS frame, ppm computing for gas sensors).

In order to compute a gas sensor ppm (parts per million), we have to do some operations depending on the measurement we had from the sensors. These operations are defined in the sensors datasheet and are specific for each sensor.

For the SO2 & O3 sensors, the ppm computing of the ppm is done using the following equation:

$$Cx = \frac{1}{M} \cdot (V_{gas} - V_{gas_0}),$$

On the above equation, V_{gas} is the voltage output signal (V), C_x is the concentration (ppm) and V_{gas0} is the voltage output gas signal in a clean-air environment. and M is the sensor calibration factor (V/ppm) and can be calculated using:

$$M (V/ppm) = \text{Sensitivity Code} (nA/ppm) \times TIA \text{ Gain} (kV/A) \times 10^{-9} (A/nA) \times 10^3 (V/kV),$$

where the Sensitivity Code is provided on the sensor label and the TIA Gain is the gain of the trans-impedance amplifier (TIA) stage of the ULPSM circuit.

For the CO sensor, the figure below shows how gas concentration in PPM after calculating the ratio R_s/R_0 using the equation:

$$R_0 = R_{S_air}/9.9$$

$$\text{ratio} = ((5.0 - \text{sensor_volt}) / \text{sensor_volt}) / R_0$$

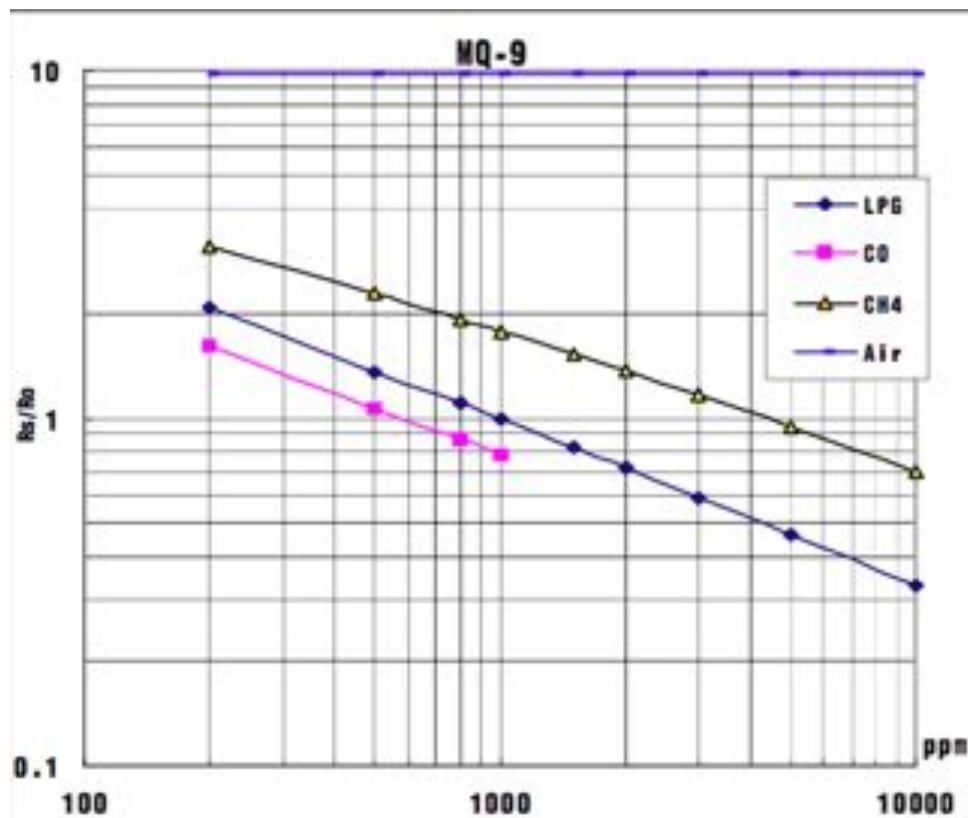


Figure 5 : MQ-9 gas concentration

Also, for the GPS, we'll have to analyze its frame in order to get the correct information. Using the NMEA protocol, we are interested in the frame **\$GPGGA** that represents data acquisition for the fix GPS.

A \$GPGGA frame looks like:

\$GPGGA,064036.289,4836.5375,N,00740.9373,E,1,04,3.2,200.2,M,,,0000*0E

\$GPGGA	:	Frame type
064036.289	:	Frame sent at 06 h 40 min 36 s 289 (UTC)
4836.5375,N	:	Latitude 48,608958° North = 48° 36' 32.25" North
00740.9373,E	:	Longitude 7,682288° Est = 7° 40' 56.238" Est
1	:	Position type (1 for GPS)
04	:	Number of satellites used to calculate coordinates
3.2	:	HDOP (Horizontal dilution of precision)
200.2,M	:	Altitude 200,2, in meters
,,,0000	:	Other information
*0E	:	Parity checksum, XOR operation of the characters between

\$ et *

2.2.6. Conclusion

This section summarizes the platform data acquisition part of the project. In the following sections, we will go further on the project and expose its networking part as well as its server architecture and the mobile application.

2.3. Networking part

Secondly, we have been focused on the transmission of the data once acquired. To be able to read them on the final application, we need to send them to a server to make the computation and storage of it.

2.3.1. Choice of the wireless technology

The figure below provided by Pycom presents the different technologies which can be used to be able to communicate with the server:

Module	WiFi	Bluetooth	LoRa	Sigfox	LTE CAT-M1NB-IoT
WiPy 3.0	✓	✓			
SiPy	✓	✓		✓	
GPi	✓	✓			✓
LoPy	✓	✓	✓		
LoPy4	✓	✓	✓	✓	
FiPy	✓	✓	✓	✓	✓
Antennas	External WiFi/BT Antenna Kit	External WiFi/BT Antenna Kit	LoRa & Sigfox Antenna Kit	LoRa & Sigfox Antenna Kit	LTE-M Antenna Kit

Figure 6 : Wireless technology choice

First, we have to define which wireless technology we want to use to communicate.

It is clear that Wi-Fi and Bluetooth protocols are not suitable for our kind of application as it delivers short range communication and Wi-Fi is very energy consuming.

LoRa and Sigfox are both long power wide area network protocols so they offer long coverage and low power consumption which is appropriate for our prototype.

If we compare briefly those two technologies, we can say that the topology of the network is the same (star topology), the urban and rural range are little different but what can be inconvenient for us is that Sigfox technology has a one hundred and forty limitation messages per day. That means we cannot send more than one message every ten minutes. On the other hand, Sigfox is a network operator so we would have to pay a subscription fee in exchange for their services.

INSA has also a LoRa gateway on the roof of the computing and electrical department so LoRa technology is easier to use for us. Therefore, that leads us to choose the LoRa wireless technology to process the networking.

2.3.2. Hardware choice

Qualitative Data is acquired using the Pycom expansion board below by connecting pins from sensors to the expansion board via a breadboard.



Figure 7 : Pycom expansion board

Next, if we compare now the module to put on the expansion board between the LoPy, LoPy4 or FiPy we can retain that the LoPy4 and FiPy are a more evolved version of LoPy. LoPy4 offers four wireless technologies and FiPy up to five (with LTE-CAT M1/NB1 in addition). We finally opted for the LoPy4 because INSA had some.



Figure 8 : Pycom LoPy4

2.3.3. Programming the Hardware

To be able to program the Pycom LoPy4 for the networking part and for the data to be sent we have used the fast micropython programming Pymakr IDE plugin with the Atom editor. The programming environment settle can be summarized with the following figure:

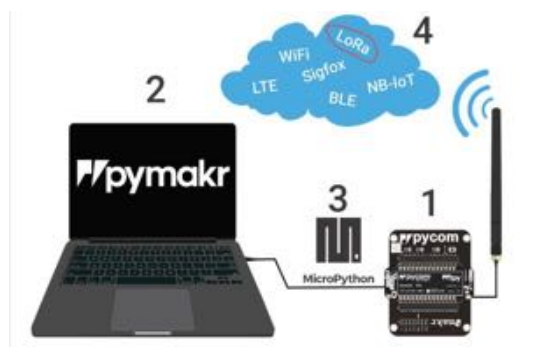


Figure 9 : Programming environment of the LoPy4

2.3.4. Over the air activation and data sending

There are two ways to activate an end device to the LoRa network. The first is over the air activation (OTAA), the second is activation by personalization (ABP). We have chosen OTAA because it provides a more secure and easier way to connect our device to the server.

Before explaining OTAA method we introduce some definitions. We define EUI as Extended Unique identifier which is a sixty-four bits long used for the identification of network components. DevEUI identifies an end device (similar to a mac address) and AppEUI identifies the application server (similar to a port number). AppKey is an AES 128-bit symmetric key to generate the message integrity code (MIC).

Before activation, end devices have to know the DevEUI and AppEUI and the AppKey. The server must know only the AppKey as shown on the following figure:

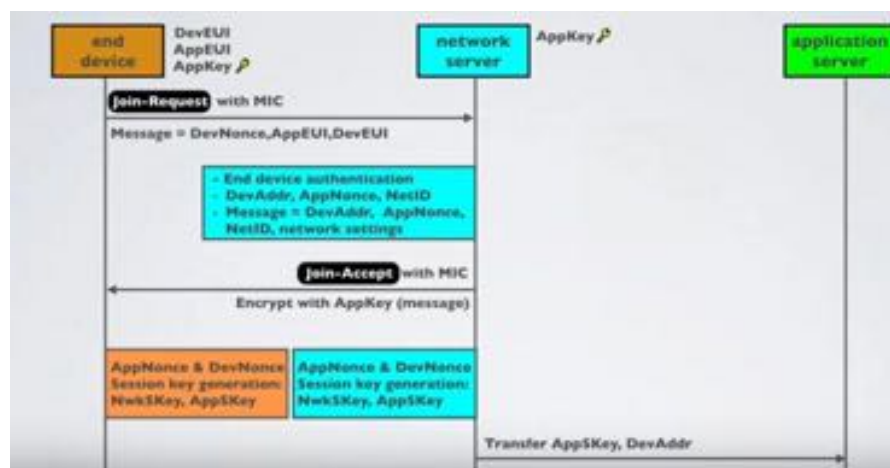


Figure 10 : Over the air activation method principle

The end device defines the DevNonce which is a randomly generated number to prevent rogue devices from replaying the join request. The MIC is generated by the AppKey.

The encrypted message sent by the end device contains the DevNonce, AppEUI and DevEUI. Once the join request is received by the network server, it checks if the DevNonce has not been used previously. If it is the case it calculates next the MIC using its AppKey (the same key as the end device) and compare it to verify the integrity of the join request message. If it is the same the end device is then authenticated.

Once authenticated, the network server send a join accept response containing networks settings such as data rate to be used for receiving, receive delay, channel frequency list and MIC still generated by the Appkey.

So, end device and network server both share AppNonce (generated number by the network server) and DevNonce and use it to generate two sessions key: the network session key (NwkSKey) and the application session key (AppSKey).

Next, the network server transfers the AppSKey and the device address to the application server as shown below:

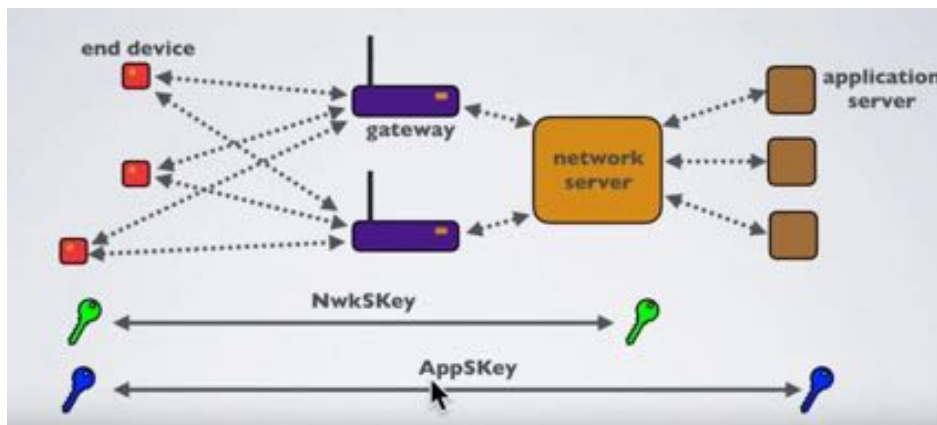


Figure 11 : Keys used for OTAA

The NwkSKey is used to calculate and verify the MIC for all data messages to ensure the integrity of it and encrypt and decrypt the payloads. The AppSKey is used to secure end to end communication and also encrypt and decrypt the payloads. Those two methods are explained in detail in the LoRaWan 1.0.2 specification.

2.3.5. LoRa parameters

The following figure presents the parameters we have chosen for our module:

```

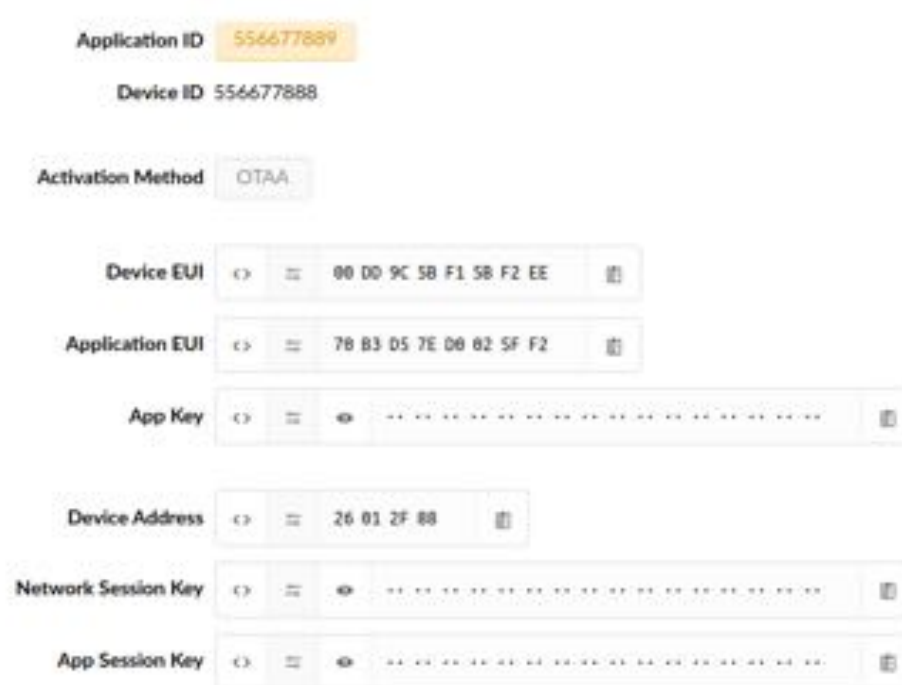
Metadata
{
  "time": "2020-02-13T14:58:06.152859191Z",
  "frequency": 868.1,
  "modulation": "LoRa",
  "data_rate": "SF7BW125",
  "coding_rate": "4/5",
  "gateways": [
    {
      "gtx_id": "eui-5827ebffec479df",
      "timestamp": 3364876013,
      "time": "",
      "channel": 0,
      "rssi": -81,
      "snr": 9.5,
      "rf_chain": 1
    }
  ]
}
    
```

Figure 12 : LoRa parameters chosen

About the metadata, the first LoRa parameters (frequency, spreading factor, bandwidth, coding rate) on this image are the parameters defined by the gateway for communication throughout the sending of the join response by the network server to the end device. The others like the received signal strength indication (RSSI) and the signal to noise ratio (SNR) qualify the quality of the signal between those two entities.

2.3.6. End device ID, EUI and keys

These parameters can be found on the thing network platform (TTN). We will introduce TTN in the server architecture part.



The image shows a configuration page for an end device on the TTN platform. The parameters are as follows:

Parameter	Value
Application ID	556677889
Device ID	556677888
Activation Method	OTAA
Device EUI	00 D0 9C 5B F1 5B F2 EE
Application EUI	70 B3 D5 7E D0 02 5F F2
App Key	[32 hex digits]
Device Address	26 01 2F 08
Network Session Key	[32 hex digits]
App Session Key	[32 hex digits]

Figure 13 : End device parameters

3. Server Architecture

3.1. Infrastructure part

The OM2M and Node.js server plays an important role in connecting TTN and the user interface which is a mobile application. During the planning phase, the idea was to have the application servers running online rather than running them in the localhosts like desktops or laptops. Also, keeping in mind various other factors like scalability, accessibility and security of the application it was planned to use Amazon Web Service (AWS) to host the application servers.

We created an AWS account and launched the instances (VMs). AWS's quick start AMIs were used to launch the VMs. In the VMs one of the Linux flavours Amazon Linux was used as the OS was available for free without any additional charge. After launching the instances using web-ui we logged into the VMs as shown in the image below by using the key pair for authentication.

```
$ ssh -i Downloads/docker-iot.pem ec2-user@18.221.245.118
```

Figure 14 : SSH command to access the VM

3.1.1. Setting up of OM2M server online

Once we logged into the VM, we installed the required packages like git, java, maven using the yum command which helps in installing the above packages by downloading them from the Amazon Linux yum repo. After installing, the bitbucket repository was cloned into the VM as can be seen below:

```
ec2-user@ip-172-31-20-3 ~]$ sudo -i
[root@ip-172-31-20-3 ~]#
[root@ip-172-31-20-3 ~]#
[root@ip-172-31-20-3 ~]#
[root@ip-172-31-20-3 ~]#
[root@ip-172-31-20-3 ~]#
[root@ip-172-31-20-3 ~]#
[root@ip-172-31-20-3 ~]# git clone https://krish2904@bitbucket.org/issnnp/snp-server.git
Cloning into 'snp-server'...
Password for 'https://krish2904@bitbucket.org':
remote: Counting objects: 35, done.
remote: Compressing objects: 100% (33/33), done.
remote: Total 35 (delta 12), reused 0 (delta 0)
Unpacking objects: 100% (35/35), done.
[root@ip-172-31-20-3 ~]#
```

Figure 15 : OM2M project cloning

Once the repo was cloned the code was compiled to build the IN-CSE and MN-CSE using the maven compiler. Below command was used to compile the code:

```
[root@ip-172-31-20-3 om2m-build]# mvn clean install
[INFO] Scanning for projects...
[WARNING] No explicit target runtime environment configuration. Build is platform dependent.
[WARNING] No explicit target runtime environment configuration. Build is platform dependent.
[WARNING] No explicit target runtime environment configuration. Build is platform dependent.
[WARNING] No explicit target runtime environment configuration. Build is platform dependent.
[WARNING] No explicit target runtime environment configuration. Build is platform dependent.
[WARNING] No explicit target runtime environment configuration. Build is platform dependent.
[WARNING] No explicit target runtime environment configuration. Build is platform dependent.
```

Figure 16 : Maven compilation command

Later, the TTN config file was copied to the IN-CSE folder and IN-CSE was started:

```
[root@ip-172-31-20-3 x86_64]#
[root@ip-172-31-20-3 x86_64]# sh start.sh
[INFO] - org.eclipse.om2m.datamapping.jaxb.Activator
Starting Data Mapper
[INFO] - org.eclipse.om2m.datamapping.jaxb.Activator
Registering XML Mapper Service..
[INFO] - org.eclipse.om2m.datamapping.jaxb.Activator
XML Mapper service registered.
[INFO] - org.eclipse.om2m.datamapping.jaxb.Activator
Registering JSON Mapper Service..
[INFO] - org.eclipse.om2m.datamapping.jaxb.Activator
JSON Mapper service registered.
[INFO] - org.eclipse.om2m.binding.coap.Activator
Register CoAP RestClientService..
[INFO] - org.eclipse.om2m.binding.coap.Activator
Starting CoAP server
Jan 12, 2020 6:11:18 PM ch.ethz.inf.vs.californium.network.config.NetworkConfig createStandardWithFile
INFO: Create standard properties with file Californium.properties
```

Figure 17 : OM2M launching

3.1.2. Setting up the Node.js server

To set up this server, node.js was installed using the yum commands and finally installed the npm using the command `npm install`. Before starting the `app.js` file, the `om2minterface.js` file's endpoint was modified so that the node server communicates with the OM2M server via private ip so that the latency can be reduced to maximum extent. Later the `app.js` file was started and was made to run in the background using `nohup` feature of linux:

```
[root@ip-172-31-41-254 snp-server]# nohup node app.js &
[1] 16578
[root@ip-172-31-41-254 snp-server]# nohup: ignoring input and appending output to 'nohup.out'

[root@ip-172-31-41-254 snp-server]#
[root@ip-172-31-41-254 snp-server]#
[root@ip-172-31-41-254 snp-server]# ps -ef | grep node
root      16578 16412  2 18:37 pts/0    00:00:00 node app.js
root      16586 16412  0 18:37 pts/0    00:00:00 grep --color=auto node
[root@ip-172-31-41-254 snp-server]#
```

Figure 18 : Node.js server launching

3.2. The Thing Network

The following part will present the use of The Thing Network in the Social Network for Pollution project to receive and forward device measurements to the OM2M server. This choice, its implementation in the project will be exposed.

3.2.1. A shared LoRaWAN network

Along with the fast development of IOT that we are currently attending came the need to find low energy protocols to allow the battery-powered devices to communicate via wireless communications.

LoRaWAN is a MAC protocol using a LoRa modulation. It allows long range wireless communication for battery powered devices. As LoRaWAN covers both the second and third layer of the OSI model, it allows to easily create networks using several devices. A typical architecture of LoRaWAN network can be found on the following figure which represent the entities and data flow:

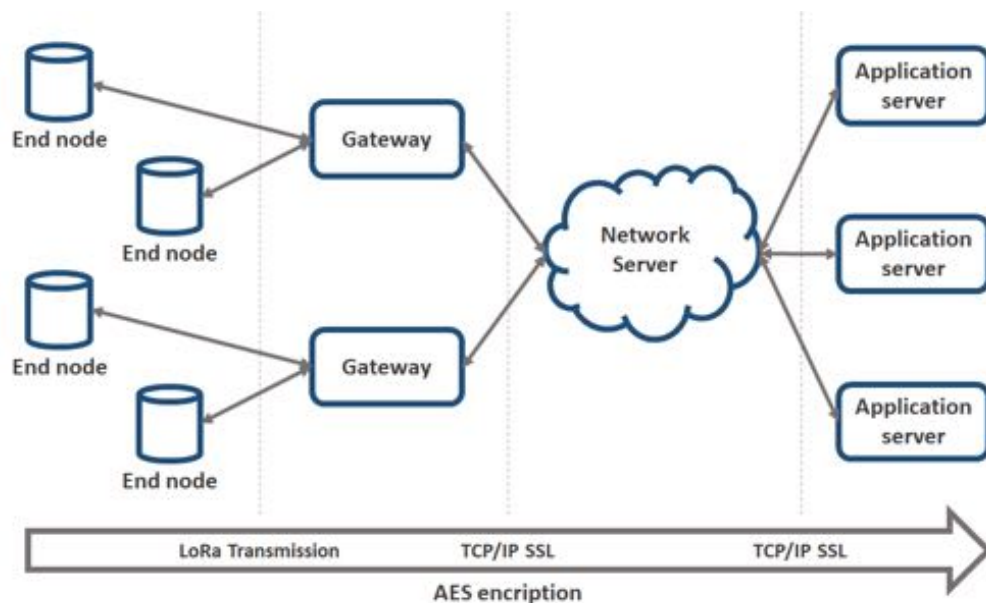


Figure 19 : LoRaWAN architecture

On the architecture, we found several types of entities. A first layer is composed by the end nodes, which represents the sensors collecting and sending data via the LoRa modulation. The end devices can monitor several parameters and send them to a gateway. The gateway is rarely battery-powered as it has to act as a translator between the LoRa packets received and the Network layer using TCP/IP protocols like http, https or MQTT. The network server is the part responsible for the data storage, processing and distribution. It could be for example a web server using a REST API to allow applications interact with its data. At the end, all the data collected and stored inside the network server can be accessed by other applications which can be of several types like web or mobile applications or other

web services. This simple architecture combined with the low cost of the LoRa hardware, gives LoRaWAN a great strength as everyone can build his own network.

The Thing Network (TTN) is the biggest open shared LoRaWAN network. It is counting present in numerous countries and gather almost 100000 users around 10000 gateways. Referring to the figure 17 above, TTN is taking in charge the network server part. Joining TTN is a simple process, aiming to bring together the largest amount of people. The end nodes and gateways are indeed added directly by the users. This from this collaboration that TTN is highly relevant. Everyone can participate to the network by adding a gateway or just using the ones nearby to build their personal network.

3.2.2. Choice of TTN

For the Social Network for Pollution, the choice of LoRa communication was made regarding the ability of this protocol to preserve the end nodes battery, and the easy process of network creation. TTN was a great solution to implement the air quality monitoring platforms. This choice was highly motivated by the client and tutor M. Monteil. As a favoring factor, previous students did a project which consisted into the deployment on the campus of a TTN gateways involving a Raspberry pi and a LoRa module.

However, this choice of using the TTN network server was really suitable because of the time gained. Future updates and development may need to the replacement of the TTN network by a network server proper to the SNP project. This replacement will lead to an increased control on the data transit from the connect platforms to the OM2M server. Along with that, it will lead to the responsibility of the security. Indeed, the project manipulates location data of the users. These data may be sensible taking responsibility and control on the securitisation of these data instead of letting this responsibility to TTN can be advantageous.

3.2.3. Applications and devices

TTN allows user to create applications and manage devices for these applications. The application purpose is to gather all devices relative to the same project or application. Every application

manages a different number of devices that you can register directly from the online dashboard. Create a device will assign it an ID which will be transmitted as metadata when an application server will get the information from the TTN application. For the SNP project, only one device was added to the application as the ID provided is not required. We use directly hardcoded ID in the monitoring platforms.

3.2.4. Data pre-processing

Once the data is sent from the end nodes and received by the TTN application, a processing can be done on these data to format them and make some computing if it is required. The SNP project uses these features to delegate to TTN the task of data pre-processing. During transmission, raw data in bytes are sent. The role of this feature called “Payload formats” is to translate these data in a JSON format. This pre-processing function is written in JavaScript. The following figure presents the result of the pre-processing on a simulated data sent by a platform:



Figure 20 : Result of data pre-processing

3.3. OM2M

This part deals with the OM2M server used to store and organize data in the Social Network for Pollution project. The choice of this technology will be presented along with its implementation in the project and its limitations.

3.3.1. oneM2M standard and OM2M platform

The oneM2M standard is the result of a consortium between several standardization bodies such as ATIS, ETIS, TTA. The initial goal of oneM2M is to enable machine-to-machine communication, regardless of the hardware and software used.

The biggest asset of the standard is that it takes a transversal position that gives it a high degree of interoperability, adaptability and scalability with discovery and integration mechanisms. This makes it possible to create applications with many disparate technologies from several different vendors. This is why this standard can be found in many fields such as: intelligent transport, health, industry and automation, home automation and connected houses...

OM2M is the Eclipse implementation of the oneM2M standard. This java implementation is proposed by the research laboratory LAAS-CNRS of Toulouse. It is an open source software used to implement oneM2M architectures.

AE (application entity) are applications that will register to our architecture. These entities are named applications in a general way but can also be objects connected to our system such as data visualization applications (client applications) for example.

The CSE (common services entity) are the network management entities. These entities implement various functions allowing them to interact with the AEs. The mechanisms implemented by the CSE can be for example mechanisms of subscriptions and notifications, discovery, registration...

At the transport level, the oneM2M can communicate with HTTP, COAP, MQTT protocols.

3.3.2. Choice of OM2M

For the Social Network for Pollution project, the use of OM2M was mandatory and decided by M.Monteil, client and tutor of the project. This constraint comes from the need to use an architecture matching with the project specifications.

As said previously, the pollution monitoring platforms may not be the only devices used in the social network project. Future implementations could possibly add more diversity of devices and applications that can bring other information or fulfil other purposes. That need of scalability highlight in a certain way the coherence of the OM2M choice. We can imagine create a complex network of devices coming from different constructors. all these devices can be integrated to our OM2M network.

Along with the scalability, OM2M bring a lot of flexibility. Based on a plugin framework, and being open source, this platform eases the creation, deletion and modification of specific functionalities. For instance, the interconnection between the TTN server and the OM2M server was done using a plug in providing an IPE for TTN.

3.3.3. OM2M architecture

Several resources come with OM2M platform. Define a clear architecture to store data in and with the correct resources is a very important aspect. The following figure present the data flow along with the network architecture of the SNP project:

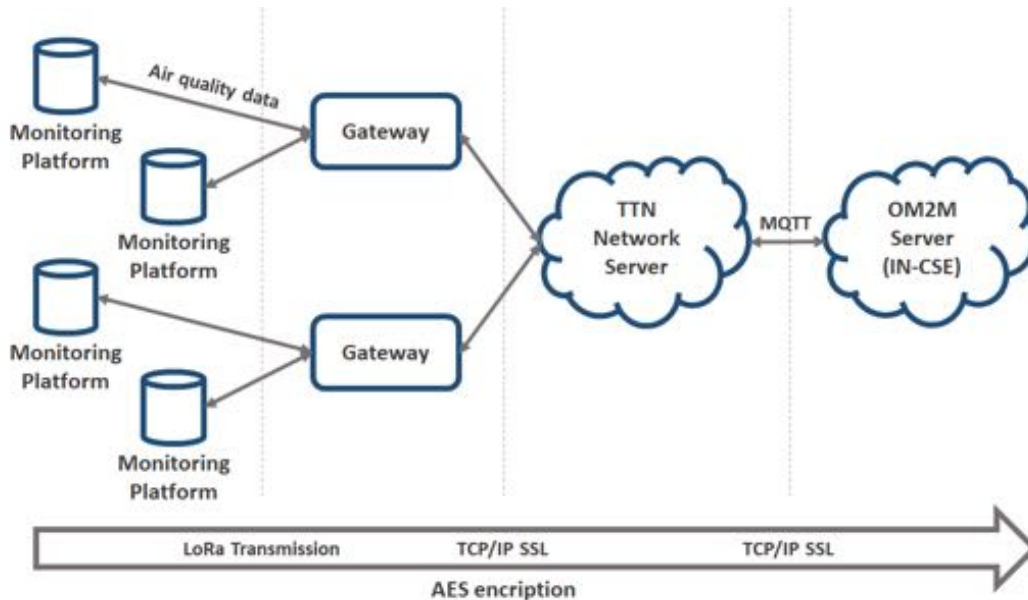


Figure 21 : Social Network for Pollution server architecture

The previous schematics was the starting point of the OM2M architecture definition. As shown, as there was no need to slice the network into subnetworks or making subgroups of platforms, only one CSE is required. The architecture inside this CSE is organized divided into several parts. The following figure represents the OM2M architecture from the dashboard available from the http://om2m_ip_address:8080/webpage address:

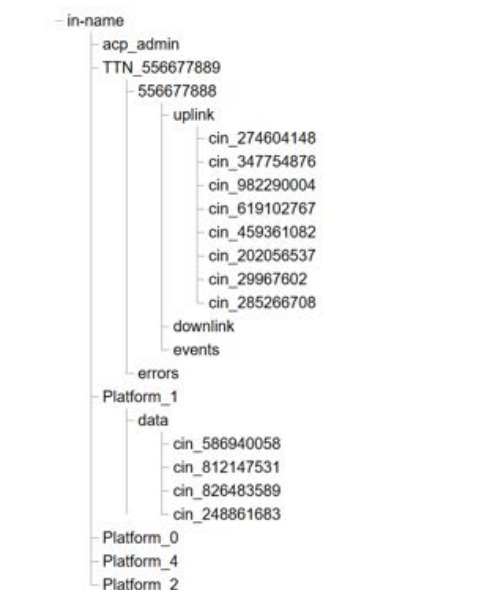


Figure 22 : OM2M architecture

a) OM2M server

The OM2M server is implemented as an IN-CSE. This CSE is responsible for the storage of all the data and has to maintain the connection with the TTN application via mqtt. The mqtt link is managed by an IPE included in a TTN plugin which will be presented further.

b) TTN Application

The TTN application is implemented as an AE under the IN-CSE. This AE is not representing strictly speaking the TTN application but it is more feedback on the state of the connection between the OM2M server and the TTN mqtt broker. Every error is pushed under an error CONTAINER. Finally, there are three other CONTAINERS representing the last messages received by the platforms, but also the events on the broker and the downlink messages which are not used in this case.

c) Platform

The air quality monitoring platforms are stored as AEs. Under every AE, a data CONTAINER is present to organize data for future possible implementations. The AE name begins with "Platform_" followed by the ID of the platform. To add some semantic content and ease the retrieve requests, labels are also defined for every platform:

- "device/platform" is a label used to make a retrieve on a given type of device, here the air pollution monitoring platform. This label includes itself in the logic of future implementations. We can imagine to have other devices from other constructors with other purposes.
- "id/XXX" is a label used to retrieve a specific platform depending on its ID. This request can be made for instance to display to a user all the information about his platform on his mobile application dashboard.

d) Air quality measure

Every measure made by the platform is stored under the data CONTAINER of the associated platform AE using the CONTENT INSTANCE resource. This resource just stores the JSON data given by the platform.

3.3.4. IPE for TTN

As said earlier, OM2M is based on a java plugin framework. Combined with the fact that the platform is open source, it is meant for every developer to add their own functionalities and participate to the platform development.

IPEs which stands for Interworking Proxy Entity, is a program which plays the role of an adapter and a translator. One of the strengths of OM2M is its ability to integrate and interconnect itself with numerous different entities. IPE can be used for example to integrate several sensors using various protocols or constructor protocols.

For the interconnection between OM2M and the TTN server a TTN IPE was used. This plugin was written by Nicolas Verstaevael and contributed by Francois Aissaoui and is accessible as an open project at: <https://github.com/Eldev/om2m-ttn-ipe>.

a) TtnApplication

The TtnApplication class represents a TTN application that we can create on the TTN dashboard. It is composed of several parameters:

- name: the name of the TTN application
- key: the access key of the TTN application (which is available on the application dashboard)

maxMessages: the maximum number of content instances which will be saved in the AE representing the application.

b) TtnBroker

The TtnBroker is a java class representing a mqtt broker. Several parameters:

- host: represent the host of the mqtt broker
- port: the port used by the mqtt broker
- uplinkTopic: a string which represents the topic used to subscribe to the messages coming from the end nodes devices
- downlinkTopic: a string which represents the topic used to subscribe to the messages coming from the network server to the end node devices
- eventsTopic: a string which represents the topic used to subscribe to the event messages

c) RequestSender

The RequestSender class groups a set of methods used to easily create OM2M resources. These methods are relying on the common resources given by the OM2M platform like the AE, Container or ContentInstance classes.

d) Controller

This class is used as an InterworkingService for the plugin.

e) Activator

The activator is used to start and stop the TTN IPE. It also creates all the resources and entities which will be used by the IPE.

f) TtnMqttClient

The TtnMqttClient is the class which represents a link between the IPE and a specific TTN application. Using the well-known `org.eclipse.paho.client.mqttv3` library, when an mqtt message relative to the subscribed topics are received, the TtnMqttClient is responsible for treating it and create or update the OM2M architecture.

g) Monitor

The Monitor class is responsible of the TtnMqttClient configuration. By using a configuration file, it will be able to create, start and stop all the TtnMqttClient. A typical `ttn_config` file can be found on the figure below:

```
{
  "broker": {
    "host": "eu.thethings.network",
    "port": 1883,
    "uplink": "+/devices/+/up",
    "downlink": "+/devices/down"
  },
  "applications": [
    {
      "name": "ttn application name",
      "key": "ttn application key",
      "max_message": 9999
    }
  ]
}
```

Figure 23 : TTN IPE config file

As shown on the figure above, there is a single mqtt broker specified but several TTN applications can be configured. In this case, every application will lead to a TtnMqttClient creation in order to manage the link and messages regarding this application. The functioning of the IPE can be described by the following sequence diagram:

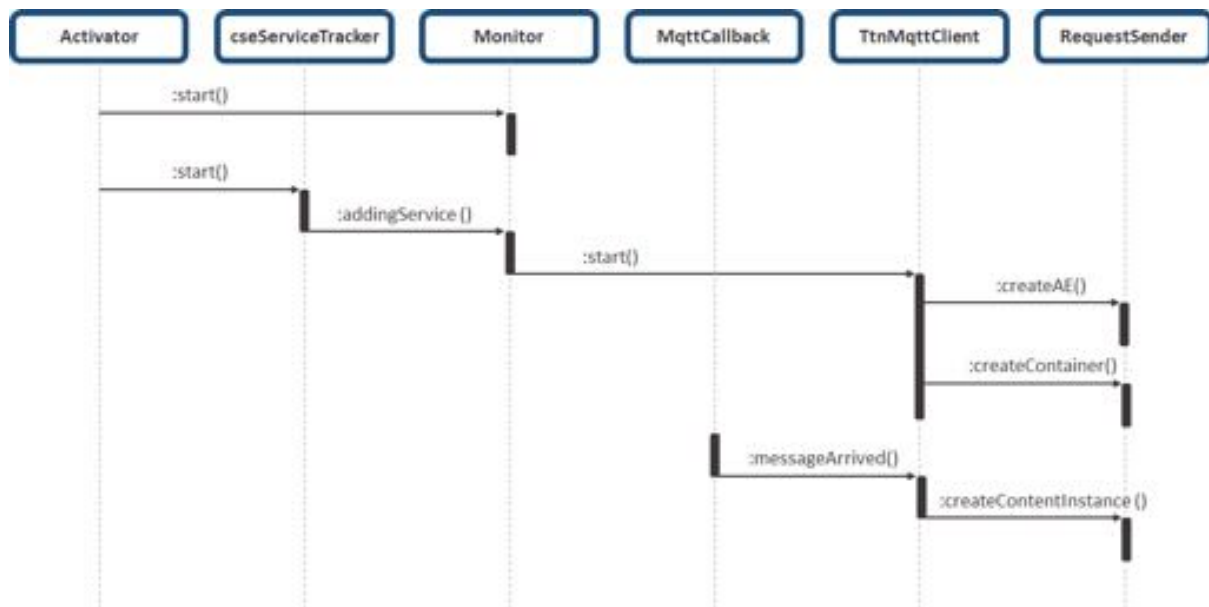


Figure 24 : Sequence diagram of the TTN IPE

3.4. Node.js back-end server

This part will present the Node server. This REST API is used to make the transition between the OM2M server and the user applications. To get the required data in order to display to the user their information or create the map representing the data, several requests have to be made. The purpose of this server is to handle these requests on the Node server layer and allow the user applications to get the data in only one request.

3.4.1. Choice of Node.js

Node.js is a software platform based on JavaScript. Its purpose is to allow the development of server oriented applications. This is thus the platform used to create web services, web applications. Express is a framework used to build Node.js applications. This is thus the default framework for the development of Node.js services.

Many frameworks are available in other languages to develop web services and RESTful API:

- Java: Spring Boot, Spark Framework, Ninja Web...
- Python: Django, Flask REST...

- Ruby: Sinatra, Roda, Padrino, Ruby on Rails
- C#: ASP.NET
- PHP: Laravel, Lumen, Silex
- Node.js: Express, Sails, Meteor, Loopback

These frameworks propose different characteristics. Some of them can be more quick to treat requests, others can handle large projects, the speed of development is also an important parameter.

The technological choice to use the JavaScript language and the Express framework was led due to the rapidity of development. A REST API developed with Express can be created easily and rapidly. Furthermore, the framework allows to create robust applications and dispense several libraries to do SSL and error handling.

Along with the above reasons, the Express framework was chosen by convenience. The back-end developer of the Social Network for Pollution project was familiar with the JavaScript language and with the Node.js platform.

3.4.2. REST API

The REST server can either be accessed with http and https protocol thanks to a self-signed SSL certificate. The Node.js server initial purpose is to ease the data retrieval from the user applications. Several requests have to be sent to the OM2M server to get information. Along with that, other tasks were delegated to this API due to his simplicity of use. A sequence diagram of the interactions between the OM2M server, the Node.js API and the user application can be found below:

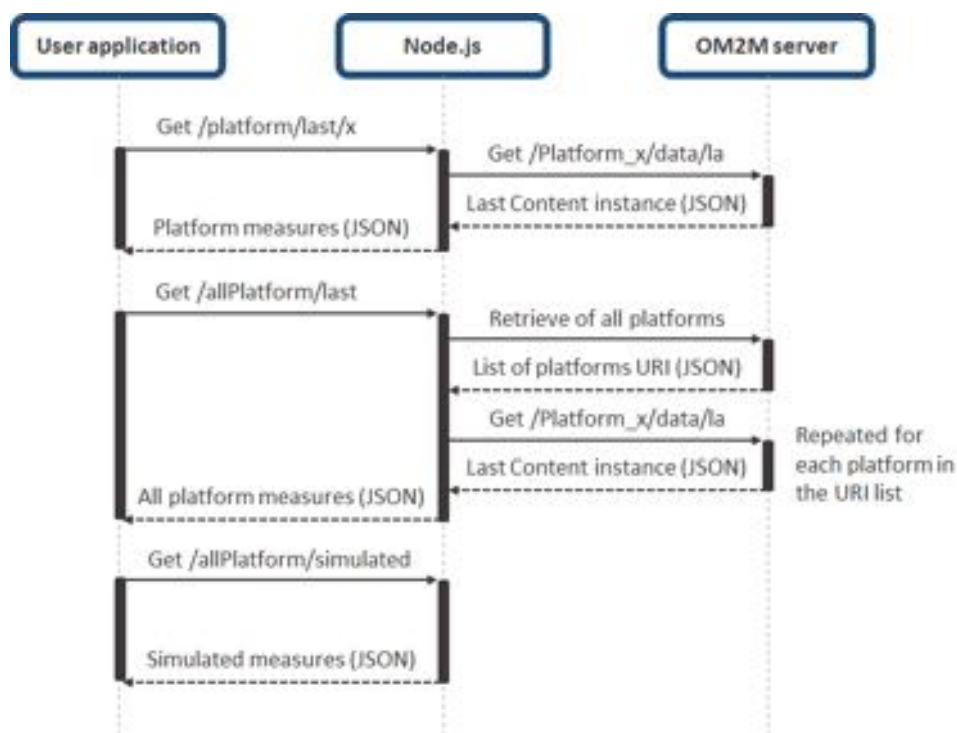


Figure 25 : Sequence diagram of the Node.js server

a) /platform/last/:id

This resource of the API returns the last CONTENT INSTANCE of a platform specified by his ID. This resource can be used by the user application to display some information to a user about his platform.

b) /allPlatforms/last

This resource return to the user application the last values of all the platform registered on the OM2M server. Its role is to give to the user application all the data necessary to be able to display a map of the air quality to a user. This resource involves several requests to give a result:

- The Node.js server make a request to the OM2M server. This request is a retrieve on all the resources which have a “device/platform” label. The response of OM2M is a list of URIs representing all the platforms.
- For every URI present in the list, the Node.js server performs a request on the given URI to retrieve the last CONTENT INSTANCE which is representing the last measurement performed by the platform.
- The Node.js server format all the responses and reply to the initial user application request.

c) /allPlatforms/simulated

This resource purpose is to allow some data simulation. Data representing several platforms are generated with the same format as the real one. These data are then sent to the user application and are displayed as a map. This resource meant to be used by the mobile application during the development.

d) /users/create

This resource allows to create a new user to the mongoDB database.

e) /users/find

This resource allows to find a user in the database in order to log him into the application.

f) /users/add/platform

This resource allows to add a platform to a user. This process will be described later on the report.

3.5. User Database

This part will present the implementation of the user database. The technical choices along with the implementation will be presented.

3.5.1. Choice of MongoDB

For the implementation of the user database, the noSQL MongoDB database was chose. The choice was made regarding the lightweight of MongoDB, its simplicity and its scalability.

3.5.2. Data structure

The new structure of the SNP project looks like the following figure:

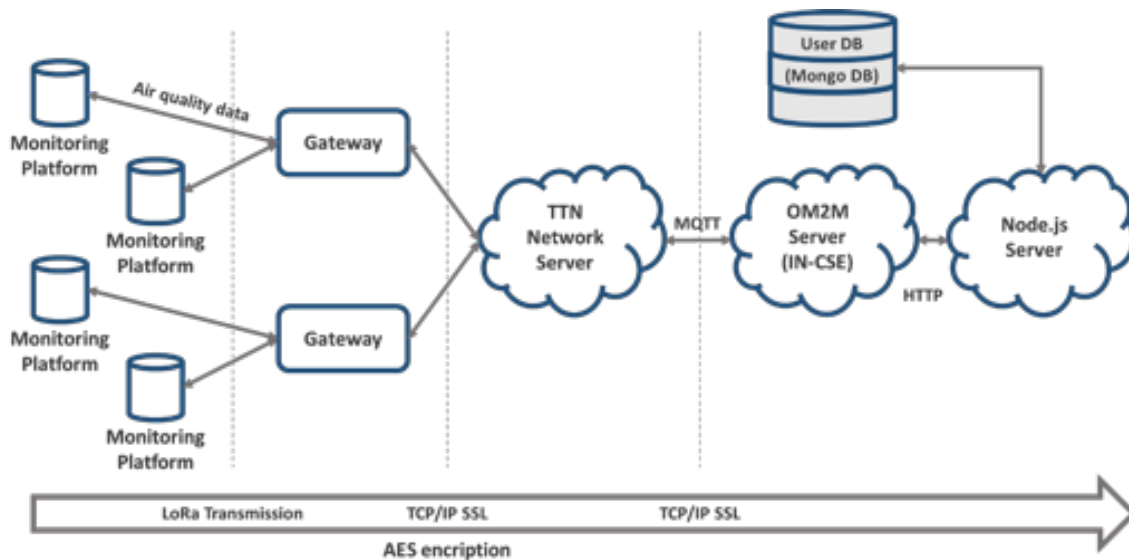


Figure 26 : Schema of the SNP project with the user's DB

We can observe that the Node.js server acts as an orchestrator, being in interaction between the user's database and the OM2M server.

The user's data structure is defined by the following structure:

- Id: which is the same as the username to ensure that two users can't have the same username
- Username
- Password
- Platforms: lists of the user's connected platforms which consists in an array of platform id

3.6. Limits and future implementations

There are several limits to the current architecture. The use of OM2M was mandatory, and the choice of this technology is coherent with the Social Network for Pollution needs and constraints. However, despite its ability to handle the scalability and interoperability with numerous devices and applications, some technological problems directly due to the OM2M platforms. Previously in the report was presented the Node.js server and the resources it can provide to make the interaction between the OM2M server and the user applications. To map the air quality data, the user application needs to get the last measurement from all the platforms. The best way to do that would be to do a request to the OM2M server to get the last CONTENT INSTANCE from all the resources which have a “device/platform” label. However, instead of getting one response with all the CONTENT INSTANCES contents, list of URIs is returned. That lead to several requests in addition to get one by one all the last measurements of all the platforms. At the end, to provide to fulfil the request of one user, $N+1$ requests have to be made to the OM2M server, N being the number of platforms. This number can rapidly increase with the number of platforms or with the number of users.

This issue is really important and was not anticipated. It can lead to the server jamming and shut down. Several options are practicable:

- Upgrade the retrieve functionality of OM2M to be able to handle this case.
- Keep the exact same architecture but with another platform than OM2M. This could be done using a MongoDB for example.

Also, a lot of implementations can be done to improve the Server part of the Social Network for Pollution project:

- Add the location handling. Now the user can only get the last measurement of all the platforms. As the project growth, we may be wanting to just have the measurement for a given location, for example Toulouse.
- Reinforce the OM2M server security with https protocol.
- Host the OM2M server and Node.js server on the same VM.
- Use signed certificate instead of a self-signed certificate for the Node.js server.

4. User application

This part will present the mobile application of the Social Network for Pollution project. The technological choices along with the application itself will be exposed. This application's purpose is to give users the ability to access a map of the data monitored by the connected platforms. It also has to provide to the user information about his own platform.

4.1. Choice of Mobile application

For the user application, the choice of a mobile application was made at the start of the project. This choice was motivated by the fact that a mobile application can gather a much larger amount of people.

4.2. Choice of React-Native

React-Native is a framework based on the React framework developed by Facebook. Its purpose is to allow the development of real, natively rendering mobile applications for iOS and Android. The functioning principle is nearly the same as the ReactJS framework but targeting the mobile platforms. One of the strengths of React-Native is that it is cross-platform. That means you can develop the same application for both Android and iOS.

The React-Native philosophy is the same as ReactJS. The language is based on JavaScript and JSX markup language which is an XML markup language. The JSX allows to create components which can be reused in other places of the project. One of the main principles of ReactJS and React-Native is to develop little components with a high reusability. The JavaScript part of React-Native is used to interact with the life cycle of the components. Also several API are implemented to give access to the phone hardware components like the accelerometer, the camera, the bluetooth in JavaScript.

Unlike other mobile frameworks like Ionic, React-Native provides the closest to native application performances. React-Native did not display components in WebViews which pull down the performances but use a set of components translated into React components. That way, developers can have the flexibility of React-Native while also having great performances.

The choice of React-Native over other frameworks like Ionic or over native development with Android studio or XCode comes from this ability to provide a nearly native application with the flexibility of web and React development. Along with that, the front-end developer of the project had a ReactJS developer background along with a mobile developer one. The React-Native was the best choice regarding its performances, its numerous number of libraries and API and its ability to produce a cross-platform application.

4.3. Applications Screens

The application is sliced into several screens. These screens can be found in annexes and their purposes will be described on the following parts.

4.3.1. Home Screen

The home screen is a basic screen that is just visible at the application start. It is composed of only one button at the moment. In future implementations, this screen may redirect screens with information about the application, redirect to other functionalities or a possible website.

4.3.2. Map Screen

The map screen is the heart of the application. Based on a library called react-native-maps, this screen displays a Google Maps based map with a heatmap representing the data collected from the platforms.

4.3.3. Login Screen

This screen is responsible for the login of a user inside the application to gain access to his dashboard. This screen will talk with the Node.js server to verify if the information filled by the user are present in the user's database.

4.3.4. Registration Screen

The registration screen allows a user to create an account.

4.3.5. User dashboard Screen

It is accessible only by users which have created an account. These users can add a platform to their platform lists and also can see the history and measurement of their personal platform. A user has to identify himself to gain access to his dashboard. On this dashboard, several information on his platform

can be displayed like the evolution of measures, the health of the platform (if any dysfunction has been detected) and other information about his account.

4.4. Google Map API

The map screen described in the above section uses a library called react-native-maps. This library dispense a React-Native component based on the Google Maps native or the Apple Plan ones. Using this library require to create a Google Cloud Platform account. This account gives the possibility to create a project and assign to it several Google API. For the Social Network for Pollution project, only the Google Map API was required. By adding an API to a project a key is given. This key is used to track the requests made by a specific project to perform some monitoring (number of requests per day, number of errors...). The whole activity of the project can be found on a dashboard represented by the following figure:

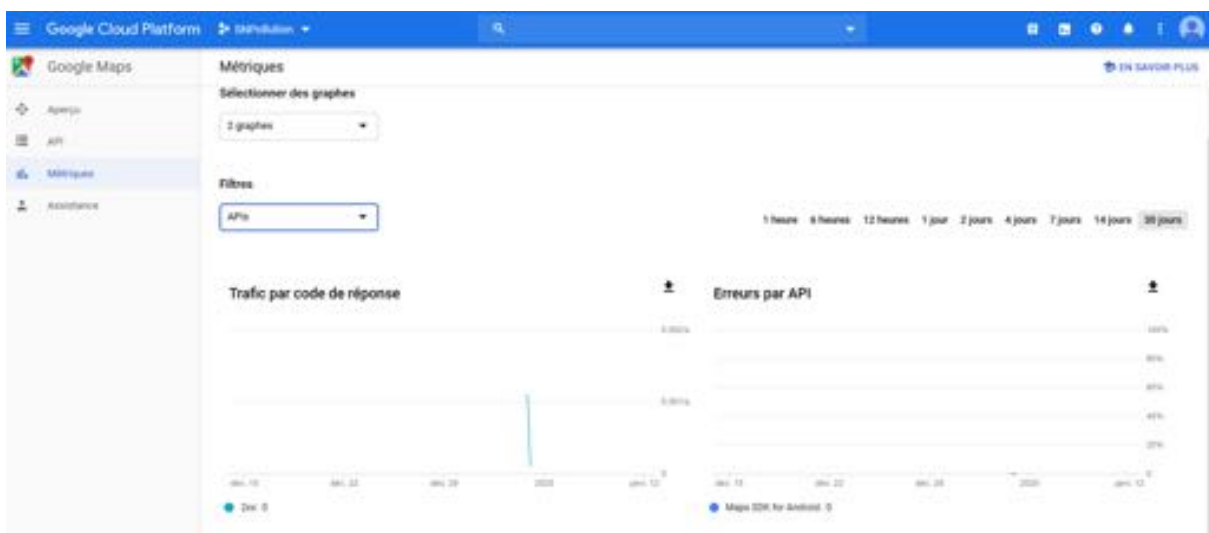


Figure 27 : Google Cloud Dashboard

The API used by the application is free to use, even with a high number of requests. However, if future implementations require other API, that may involve a payment depending on the number of daily requests.

4.5. User and platform management

4.5.1. User registering process

Create a new user is quite simple. It can be done directly from the registering screen from the application. The only requirement is that the username may be unique so that two users can't have the same username.

4.5.2. Platform registration process

To fulfil the easy-to-use requirement, a simple way to add a platform to the project was defined. When a platform is powered for the first time, it begins to transmit data to OM2M with a hardcoded ID. OM2M thanks to the TTN IPE will create an AE as showed previously for the platform. However, any Access Control Privacy has been defined so the measurement done by the platforms are not stored as CONTENT INSTANCES under the platform's AE. By this mechanism, a platform is only able to create measurements under its own AE. When a user is logged in, he can add a platform to his account. This action will add the platform id under his platform property in the mongo DB and also create an ACP linked to the platform's AE. This ACP give full access to different originators:

- The admin originator
- The user originator (username:password)
- The platform originator (platformID:platformID) which enables the platform to register measurements under its AE.

From the moment where the platform is registered to a user, it can register its data under its AE.

4.6. Future implementations

The application as the other parts of the project is willing to be improved. For now, two of the three screens are not yet implemented and some functionalities relative to the map are also not present. Some improvements are listed below:

- Handle the location of the user to display localised data.
- Add a gps functionality, which can be used to find the less polluted circuit to go to a point A to a point B.
- Optimise the app performances.
- Deploy it on Google Play and Apple Store.
- Bring some graphic redesign.

5. Business plan

5.1. Introduction

In this part of our report, we will take a deeper look at our market and the pricing model of our solution. We realised a business model canvas in order to illustrate our business model in a straightforward and structured way. In this canvas you will find many insights about our customers, our value propositions and the channels through which we offer our solution. You will also have a better understanding at our revenue streams. You can find this Business Model Canvas in the annex, at the end of our report. At the end of this section, you will also find information regarding the acceptability of our solution and potential funding.

5.2. Market study

5.2.1. The client

Fortunately for our project, which try to monitor the pollution and provide a transparent way to the user to let them know about their surrounding environment, in short, this is an application that benefits the public, enhances residents' awareness of environment, therefore we can not only sell our product to individual, but also to some institution client like city council, environmental research centre, school and universities even cloud computing service provider which can integrate our data in their platform.



Figure 28 : Social Network for Pollution benefits

5.2.2. The market

a) B2C market (Business-to-Consumer)

Domestic users familiar with IT and resident in big cities, these people care about their quality of life, then the pollution indicator which we provided could be very useful for them, in most cases, they have relatively high consumption levels and they would like to invest in their own health, so we have a great expectation for this specific market valuations.

b) B2B market (Business-to-Consumer)

Partners and corporate users, mainly schools and public institutions.

We did some research in the estimation of the target market, first step we pick up the slice of age from 15-59 from all the population, and select the people who are highly educated, we suppose that these people who would really care about the subject about the pollution, and they are our target client.



Figure 29 : Numbers about the target population

5.2.3. Competition

Today, many solutions exist to monitor the pollution levels. You can find dedicated web apps like: breezometer, waqi, airparif or pollutrack. However, the precision is limited as you do not have your own device and thus cannot access the pollution levels at your exact location.

Having an IoT device allow users to have precise and real-time levels of pollution. Many hardware solutions exist. You can find the AirBeam by Habitat Map. It is a ready to use « air quality instrument » costing 289\$. Smart citizen company also offers its own hardware solution. The smart citizen starter kit cost 119\$ but will need to add the shipping cost...

In recent years, many initiatives for citizen measures of air quality have emerged, such as Ambassad’Air in Rennes, Smart Citizen, Air Citizen, Citizensense, Mobicitair or Luftdaten. You can have a deeper look at those different initiatives on the links below:

- **Breezometer:** <https://breezometer.com/air-quality-map/>
- **Waqi:** <https://waqi.info/>
- **Airparif:** <https://www.airparif.asso.fr/indices/horair> <https://www.paris.fr/pages/etat-des-lieux-de-la-qualite-de-l-air-a-paris-7101>
- **Aircitizen:** <https://aircitizen.org/>
- **Mobicitair:** <http://www.mobicitair.fr/>
- **Citizen Sense:** <https://citizensense.net/kits/airkit/>
- **Luftdaten:** <https://luftdaten.info/>
- **SmartCitizen:** <https://smartcitizen.me/>
- **Aircasting (Habitatmap):** <https://www.habitatmap.org/aircasting>
- **Airbeam (Habitatmap):** <https://www.habitatmap.org/airbeam>
- **Ambassad’Air:** <http://www.wiki-rennes.fr/Ambassad'Air>

5.2.4. Positioning

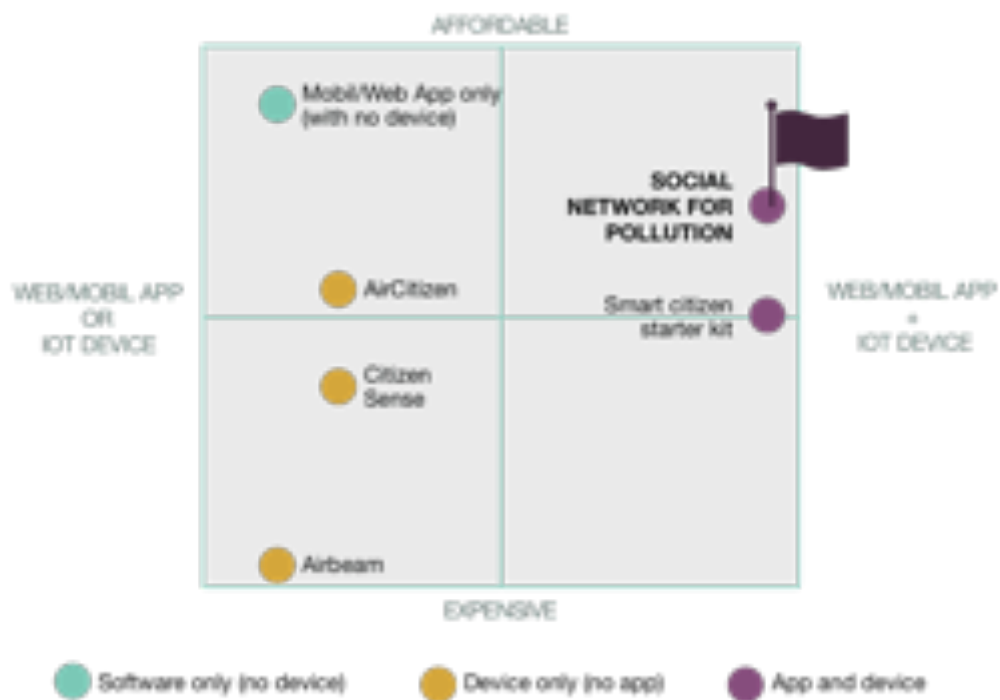


Figure 30 : Social Network for Pollution's market position

5.2.5. Competitive advantages

Our solution has various advantages. First of all, we offer a complete solution with both an IoT device and a mobile app to visualize the data. Our solution is also quite affordable when we consider the price of the Airbeam for example. And finally, we are the only solution of that kind in our area (South of France). Toulouse is a dynamic city and one of the French leaders regarding IoT. The population is also quite friendly technologically speaking. Toulouse is a great place to launch such a solution.

5.3. Pricing model

5.3.1. Business to Client (B2C)

a) Device pricing

The current production price of our device is about 200€ each. And the price of our device will be set between 100€ and 150€. By increasing the number of produced devices we will be able to lower significantly this price. We estimated that with a production of 50 devices the price of each device could go down to 100€.

We saw just above that our worldwide market size is really wide. And if we narrow down our search to Toulouse, we still obtain a lot of potential buyers.

a) Mobile app pricing

In order to monetize our mobile app, we decided to go for the freemium model (In-App purchases). It is a very popular way to monetize mobile apps. In this model, we distribute our mobile app for free and generate revenue from users that are willing to enhance their experience by purchasing or subscribing to new features. Usually, in a freemium app, 0.5 to 2 percent of users are willing to pay.

Since few users will monetize, it is important that our mobile app reaches a wide audience.

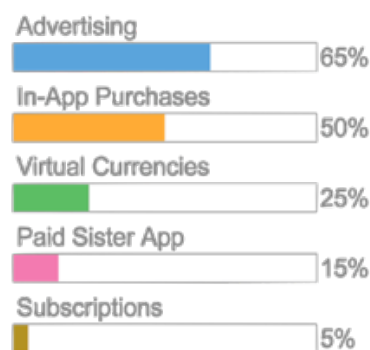


Figure 31 : Commons methods of mobile app monetization



Figure 32 : Different content provided

We will offer 2 options to domestic users, the free access to the mobile app or the premium access. The premium price will be set to 6€ for an unlimited access to all the features.

b) Sales channel

A sales channel is a way of bringing products or services to market so that they can be purchased by consumers. For our devices, our main sales channel will be direct sales and maybe online e-commerce. And we will use app-stores to distribute our mobile application.

5.3.2. Business to Business (B2B)

a) Pricing

For institutions, schools, private and public partners, the sales of devices and premium access of our mobile app will be done through a partnership. The price and conditions of the partnership will highly depend on the needs of the client and the number of potential users.

Example: For a university of 10 000 students that would buy a set of sensors to equip its campus (let say 20 sensors) and give premium access to all its students, it would cost a total of 4000€ (2000€ for the devices and 2000€ for the premium access)

The mobile app, sensors and data retrieved could also be used to organize practical works for the students.

b) Sales channel

For businesses, the contracting process and sales will be done through direct marketing. This means we will directly reach out to prospective customers by telephone or digital communication tools such as email or LinkedIn.

5.4. Acceptability

Acceptability of a project depends on a variety of criteria, including performance requirements and conditions, these standards must be met before the projects are accepted. For instance, on our project, one of the acceptance criteria that we worked on was user-friendly. Even though we have already implemented the central part of this project which is the communication and storage of core data, but if we want the client or the market accept our product, we still need a lot of things to improve or to implement, just like the application interface. Secondly all the unit tests need to be completed successfully, we will try our best to meet the requirements of all customers and perform rigorous testing before delivery, at the same time some backup and restore testing are necessary.

In the beginning of the project, we had several meetings just to gather a clear set of requirement and acceptance of this project, as all the development part are based on that. As a team, we persisted in our efforts to gather a clear set of requirements and the acceptance criteria and were able to get them approved. This helped us a lot in reducing the time required for coding and development and avoid other conflicts which would have eventually ensued had this not been done.

5.5. Funding

For a start-up company, the funds are basically in short supply. Without the funds, the start-up business like us cannot make progress easily. Many small businesses fail because they don't have the funds to continue. Firstly, the equipment, for example, higher-precision and larger number of detectors, and a server with a larger physical storage space, secondly daily necessities, all of this as business operating costs may hinder the development of enterprises.

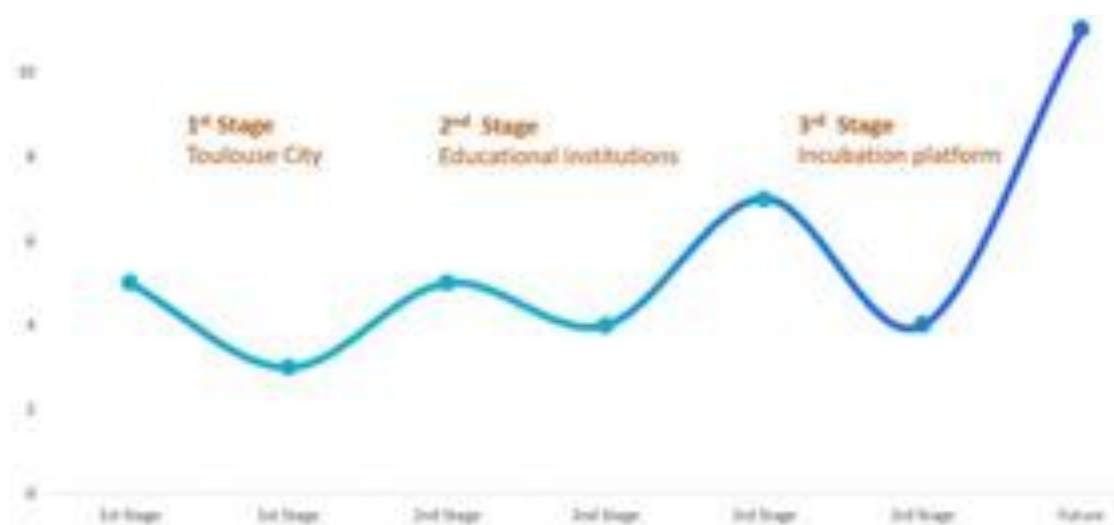


Figure 33 : Project development stages

Our project has a great guiding role in the promotion of environmental protection policies and environmental supervision for the municipal government, hence we try to submit our business plan to Toulouse City Council or the Toulouse Environmental Protection Agency in cooperation or financing, and then we will develop the next version according to their needs. If we can get successful in Toulouse, it is suggested to expand in other city in France even Europe.

We can try to cooperate with other educational institutions, using our real-time data platform to provide data support services for energy departments in some universities.

The business incubation platform aims to help high-tech achievements and entrepreneurial enterprises to promote cooperation and provide financial support, aim to make enterprises "bigger", especially in the early stage of a start-up. As a local innovation project, we will seek to join some local platforms to use their resources and funding.

When we have enough start-up funding, in the subsequent financing process, we must also consider the optimization of financial structure and the exit of old investors. At the same time, we still have to work and improve our product, as all the investment are based on the functionality of our platform.

6. Project management

6.1. Project work breakdown structure

The Work Breakdown structure (WBS) is a project management method which aims to slice the project into several parts. This division help to better organize the project by having defined sections. This method of division was used to define the project parts and assign them to every member of the group as presented on the figure below:

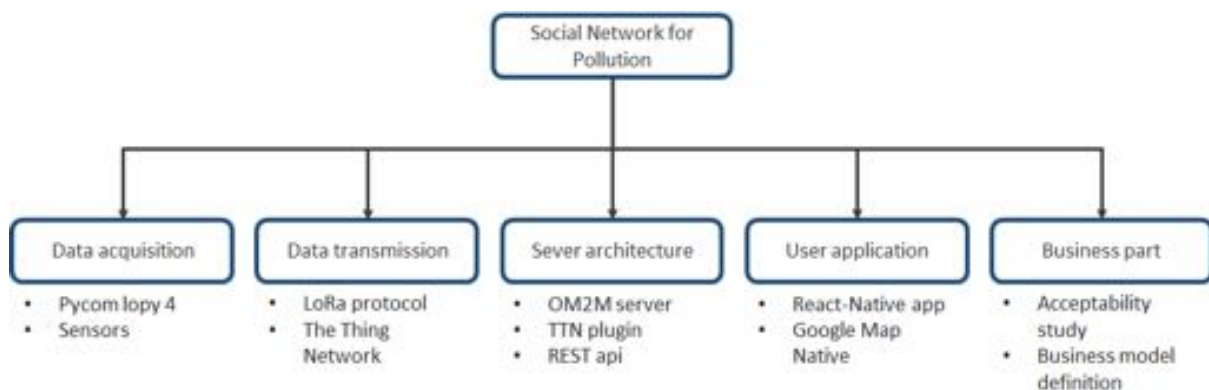


Figure 34 : Social Network for Pollution WBS

The Project has been sliced into five parts:

- Data acquisition, which is the platform development with the sensors and data acquisition from the sensors.
- Data transmission, which is the transmission of the sensor data over LoRa protocol.
- Server architecture, which is the interconnection between TTN, OM2M and the Node.js REST API.
- The user application, which is the user application development.
- Business part, which is the part responsible for the business model definition, study of the market and study of the acceptability.

6.2. Team



Ismael



Yuheng



Théo



Claire



Franck



Krishna

The Social Network for Pollution team is composed of six people. All the group members have different backgrounds and skills and have been distributed on different tasks of the project:

- Claire, IOT Valley: Business and acceptability aspects
- Yuheng, TSM: Business and acceptability aspects
- Ismail, MSIoT: Platform data acquisition
- Franck, ISS: Data transmission with LoRa
- Krishna, MSIoT: cloud management
- Théo, ISS: mobile application and server development

6.3. Project management method

The methodology used for the Social Network for Pollution management is the Agile Method

The Agile methodology is based on this simple principle: plan the project throughout its progress and not plan the project entirely before starting it. The Agile method allows you to take into account all the hazards that can occur throughout a project without disrupting its planning. The Agile method recommends setting short-term objectives. The project is therefore divided into several sub-projects. Once the objective has been reached, we move on to the next one until the final objective is reached. This approach is more flexible. Since it is impossible to foresee and anticipate everything, it leaves room for the unexpected and changes.

Another important point is that the Agile method is based on a privileged relationship between the client and the project team. Customer satisfaction being the priority, the total involvement of the team and its reactivity to customer changes and unforeseen events are necessary. The dialogue with the customer is privileged. It is he who validates each stage of the project. Their changing needs are taken into account and adjustments are made in real time to meet their expectations. With the Agile approach, nothing is set in stone. The project team must be able to constantly question itself and continually seek to evolve.

This method was the most suitable regarding the project. Development projects are highly willing to be realized following this type of method. Furthermore, the flexibility that offers the Agile method was

crucial to meet the needs of the client but also to adapt ourselves to our planning and organizational difficulties due to the fact that two members of the group were following other formations.

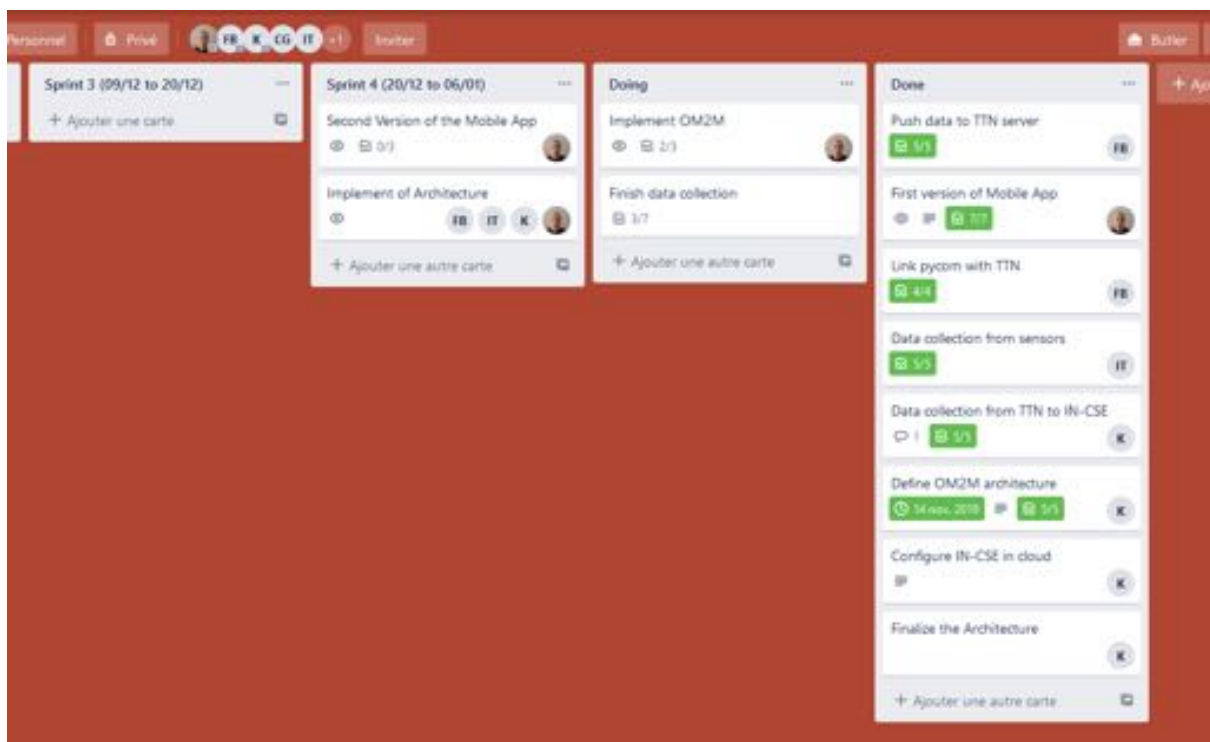
6.4. Project management tools

For the project management, several tools were setup to ease the work and communication between the different group members.

Google drive was used to share documents like reports and presentations. It also gives the opportunity to work at the same time on the documents which is convenient during reports writing.

Slack is a communication platform that is highly convenient for the project management. It proposes to create channel which can help to divide communication by project parts.

Trello was used as the task planner of the project. Sprint were defined here with their tasks. A picture of the Trello dashboard can be found on the figure below:



Bitbucket is the software development management and hosting platform chosen for the Social Network for Pollution project.

7. Results

At the end of the project, the whole transmission chain has been implemented. The platform is able to get the data from the sensors and send them via LoRa radio communication. The data measured are real data representing the air quality present at the platform localisation. The data sent to the LoRa gateway is passing through the TTN network server where a pre-processing is done to adapt the raw data to a JSON format. The data is published on the TTN MQTT broker and is received by the TTN IPE implemented in the OM2M server. The IPE creates the OM2M resources corresponding to the measurement. Finally, users can display these data onto a mobile application which sends requests to a Node.js server. This server is making the interaction between the mobile application and the OM2M server.

The user's database ensures that the integration of new users and platforms is really easy also from a non-technical person.

Furthermore, a complete market study is done including a competition study, a business and pricing model, a founding study and an acceptability study. These parts of the project are really important to have a clear view on the impact the project may have on users and on society.

However, many improvements and features can be brought to the project but at the end of the semester, the whole data transmission chain is set, the architecture is implemented, a business model is defined along with a market and acceptability study. Thanks to that, the future implementations will be greatly facilitated.

8. Conclusion

This project was really interesting. From a technical point of view, this project group a large panel of skills that we developed during the formation. The fact that the project covers the creation of a hardware device to the development of a server to store data and an application to display it is really interesting and representative of the IOT domain.

From a human point of view, working with people coming from different backgrounds and are specialized into different domains was a good experience and a good way to learn.

Finally, even if there are some improvements and enhancements to bring to the project, the result is quite satisfying.

References

- [1] Statista Research Department. Les objets connectés - Faits et chiffres. 9 août 2019.
<https://fr.statista.com/themes/2972/les-objets-connectes/>
- [2] Air pollution: everything you should know about a public health emergency. 5 November 2018.
<https://www.theguardian.com/environment/2018/nov/05/air-pollution-everything-you-should-know-about-a-public-health-emergency>
- [3] Adrien Lelièvre. Le coût de la pollution automobile estimé à 67 milliards d'euros en Europe. 27 novembre 2018. <https://www.lesechos.fr/industrie-services/energie-environnement/sante-le-cout-de-la-pollution-automobile-estime-a-67-milliards-deuros-en-europe-150688>
- [4] Noëlle TSAPAS. Des préoccupations écologiques. 13 February 2019.
<https://www.bulletindescommunes.net/toulouse-preoccupations-ecologiques/>
- [5] Sriganesh K. Rao, Ramjee Prasad. Impact of 5G Technologies on Smart City Implementation. 2018.
<http://iranarze.ir/wp-content/uploads/2018/09/E9300-IranArze.pdf?fbclid=IwAR1egcPQ92r-8ejTya1JOervBW88Ffxwlp1aC72XGEmVYTIAe2b1NKmnfiY>

Annex 1: Franck's abstract

Social Network for Pollution

Abstract – Bourzat Franck

Key Words: Pollution, Internet of Things, Connected objects, Mobile application

Nowadays, there are more than 593 million of connected objects in the world. The development of the Internet of Thing (IoT) and connected objects in general can be an asset to be aware of climate change and act in favour of future generation. Our project aims to monitor pollution and air quality by providing a mobile application and a connected device to users to give them the ability to see in real time levels of pollution measured areas of connected objects. This project also takes part in social context where a user can register his own device and have access to the data from other users. To do this, we are selling a connected device which measures gas concentration (ozone, nitrogen dioxide, sulphur dioxide, carbon monoxide) involved in pollution and other general parameters (temperature, pressure...). Those data are sends to a server going through a LoRa gateway. Finally, a mobile application for users collects this information and displays a map of the devices and measured parameters. The prototype developed and related results are quite conclusive. We are able to send data correctly and see in real time thanks to the application the levels of concentration of each gas.

For further work, we will have to think about the scalability of our product to be distributed at a larger scale.

Annex 2: Ismail's abstract

Abstract

This project took place in order to let people be aware of the polluted zones in their region where there isn't enough information about air quality provided by their government.

The main objective of this project is to implement an air pollution monitoring platform that will help users track the most polluted areas in their region so they can avoid them.

In order to achieve this objective, the implemented methods consist of using specific sensors to collect data (e.g. gas, localisation, temperature) that will be transmitted over a LoRa network and finally recovered by a suitable mobile application.

As a result of this project, we were able to build an initial prototype, where most of the functionalities are working, that is ready for testing in the INSA campus.

Furthermore, the next steps will be to first test the platform in the city of Toulouse, then find collaboration partners (e.g. City Hall, Companies interested by the project) in order to expand the air pollution detection range.

Keywords: air Pollution, air quality, gas sensor, mobile app, LoRA, TTN.

Annex 3: Krishna's abstract



SOCIAL NETWORK FOR POLLUTION

Air pollution kills an estimated seven million people worldwide every year. WHO data shows that 9 out of 10 people breathe air containing high levels of pollutants. More than 80% of people living in urban areas that monitor air pollution are exposed to air quality levels that exceed WHO guideline limits that include both indoors and outdoors.

The objective of the project is to detect the presence of air pollution and other parameters in the atmosphere and inform the public about the conditions prevailing locally and re-route the traffic through least polluted routes.

To achieve the above objective, a prototype consisting of the CO, NO₂, SO₂ and O₃ sensors along with humidity, pressure and temperature sensors are connected to a pycom device which then sends the metrics over LoRa wireless network to the TTN. From TTN Cloud, via TTN plugin, data is sent to OM2M server hosted in cloud. The data is then made available to the users through a mobile application which is sent through SNP-server.

The end goal is to create a prototype at a low cost and deploy it in the INSA campus and demonstrate live and near real time pollution metrics through heat maps in mobile application. Further, we also intend to partner with Toulouse Municipality for deployment is large scale for air pollution monitoring in the city through public engagement.

Annex 4: Mobile application screens

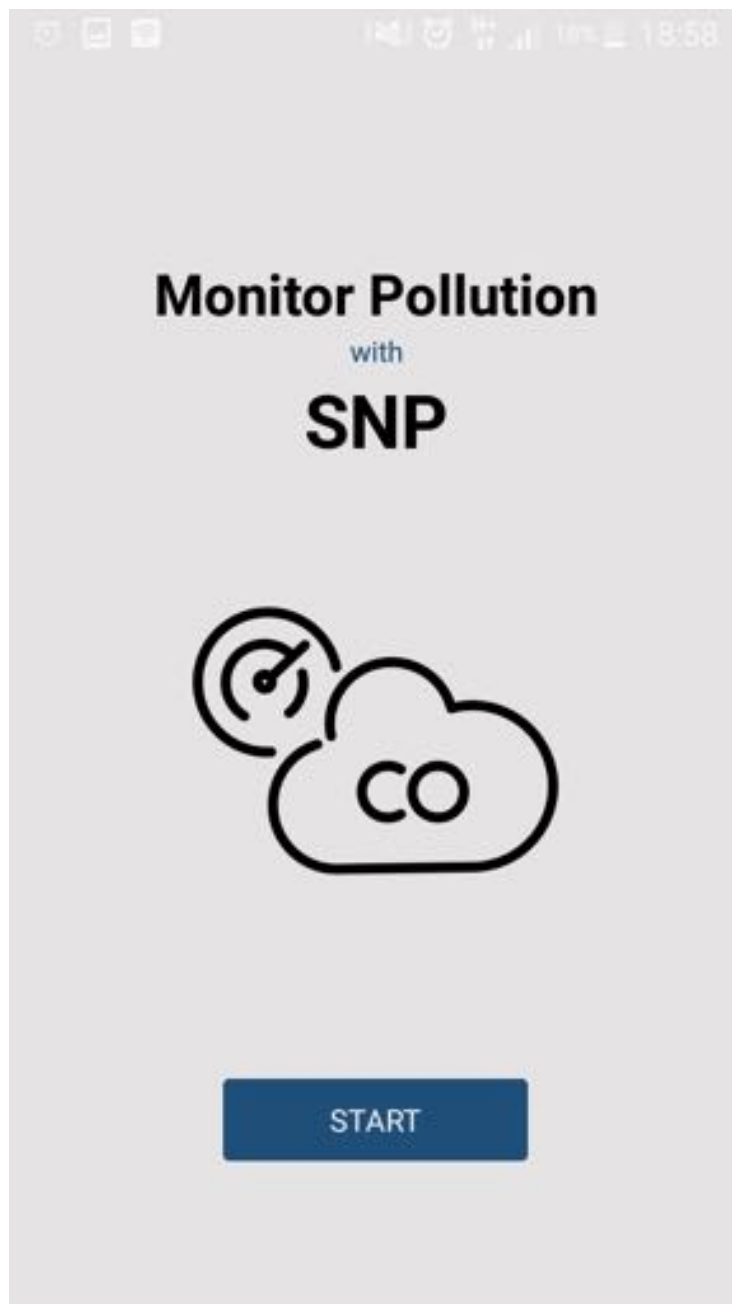


Figure 35 : Home Screen

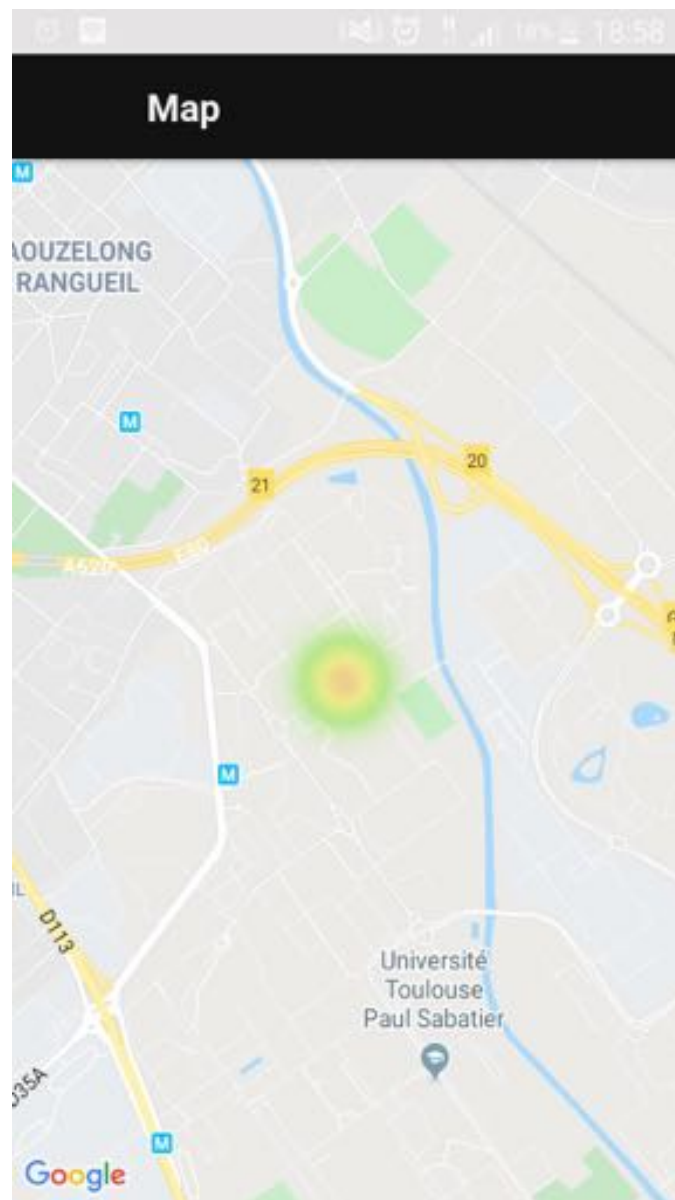


Figure 36 : Map Screen