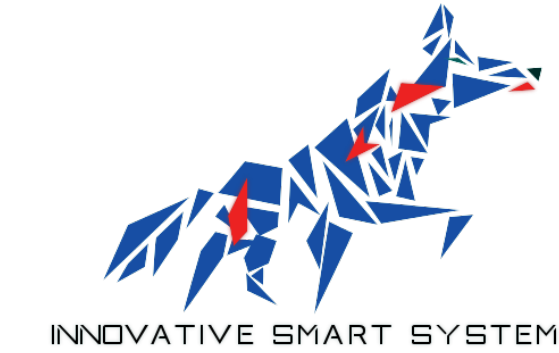
5-ISS Integrative Projects
Final Paper
IoTracking: GPS tracking for boat regattas

Josué ALVAREZ, Cyril ANAK STELL, Axel CHAUVIN,
Aminata DIOP, Cécile DUTHOIT, Linn MJELSTAD, Clovis OUEDRAOGO

September 2016 - January 2017



INSA
TOULOUSE

Contents

Abstract: The project aim was to deploy an ad-hoc LoRa network from scratch and to develop a user-friendly web application to allow spectators to follow the evolution of boat regattas in real time. This document will briefly explain the different needs and constraints that we faced and the technological choices we made to develop our system.

Keywords: *LoRa, GPS, boat regattas, deployment of an ad-hoc network, hardware, web application*

1 Introduction

This paper aims to explain and justify our technological choices and implementation decisions, as well as analyze our results. We will first present a short high-level summary of the project. A lengthy overview will then be developed, followed by a detailed explanation and a justification of our technological choices. Finally, this paper will conclude by presenting our results.

2 Overview

2.1 The initial idea: Make boat regattas more interesting to increase their audience

One of the projects proposed to the 5ISS students this year was inspired by a discussion between two people working at the LAAS (Laboratory for Analysis and Architecture of Systems), one was Thierry Monteil, a professor at INSA in the Electrical and Computer Sciences department (GEI, Génie Electronique et Informatique), and the other was an individual who is passionate about sailing and is a member of an association that organizes, among other activities, boat regattas. That club, called CVRL[1](Club de Voile des Rives de Léran), would like the public to be more involved during its regattas and would like to give them a user-friendly interface to follow the evolution of the races in real time. CVRL is located along the banks of Lake Montbel in Ariège, France. Because the lake is far from the surrounding cities, there is nearly no network coverage at the site, which makes this project a real challenge.

Figure 1: Montbel Lake, Ariège, France.

The first idea was to implement an application to show the position of each boat on a map of the lake. Because there is no network coverage, the entire architecture of the project needs to be implemented locally, with no need for an Internet connection. This requirement increases significantly the workload. The new network needs to be deployed over the lake, from scratch, to transfer data from the sensors to the server, which would be at the nautical base. These sensors, which are to be mounted on each boat that participates in the regatta, send different types of data, such as the positions, directions, and speeds of the boats. These data are then given to the audience via a user-friendly web application showing the route of the regatta and the position of each boat on a map of the lake. The association also requested video streaming of the starting line, which implied an additional Wi-Fi network in parallel with more bandwidth.

This project was extremely challenging because it combined skills in electronics, networking, web design, software architecture, and programming.

2.2 Our team

Our team is composed of seven individuals, including four network and telecommunication experts, one computer science expert, one automatic control and electronics expert, and one business expert.

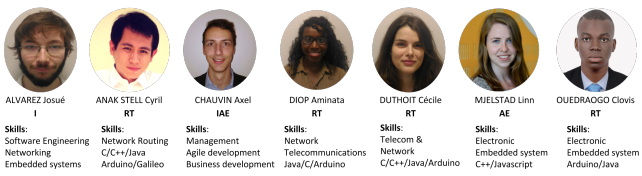


Figure 2: The IoTracking team.

2.3 Personal goals and motivation

We chose this project for two main reasons. First, we were particularly enthusiastic about the idea of locating boats in real time. Second, the fact that this project contained various domains ranging from electronics to web development was highly motivating.

We considered this project to be a stepping stone to improve our skills.

Indeed, in addition to our intention to make a relevant and functional prototype, we also hoped to achieve the following goals during this project:

- Consolidate our skills;
- Develop our ability to work in groups;
- Develop our creativity and engineering skills to solve practical problems; and
- Learn how to carry out a project starting with the specification and continuing through to the realization via prototyping.

2.4 Technical choices

To bring the project to life, we needed to make careful technical choices for the different parts of the project: the network infrastructure and the devices.

First, concerning the collection of the location data from the devices on the boats, we needed a network infrastructure compliant with the following constraints:

- Low cost: the network infrastructure need to be really cheap;
- Low power: the devices connected to the network need to consume a very low amount of energy;
- Long range: the network needs to cover the entire lake (approximately 9 km^2);
- Data rate: each device must be able to send 3 bytes every 10 s, and the network must be able to simultaneously handle 100 devices;
- Mobile objects: the network must be able to correctly capture the data sent from devices moving at a maximum speed of 30 km/h; and
- Ease of deployment: the network infrastructure must be easily and quickly deployable because we have a limited amount of time to build this project.

Most of the wireless technologies did not comply with these constraints:

- Wi-Fi: its range is too short and it consumes too much energy;
- Bluetooth and Zigbee: these technologies are used to build low-range networks; therefore, they are not appropriate for our situation;
- 2G: we have no coverage at the lake, and the deployment of a private 2G network would be too complex;
- Sigfox[2]: the data rate provided by the Sigfox network is too low (a few bytes every 2 min); and
- 4G LTE-M: currently not available.

We needed our devices to be able to perform geolocalization every 10 s. There are two ways to achieve this.

- Use the different LoRa gateways to perform triangulation on the signals they receive. This solution is the most cost effective and energy-efficient method. However, the precision of the triangulation might be very low.
- Use a GPS module on the devices to collect the GPS data to send through the LoRa network. This solution is more expensive and causes the devices to consume more energy. However, we can achieve localization with good precision (approximately 5–10 m).

Taking all of the constraints of the final product into account, we opted for a solution that is minimalistic and is based on cutting-edge technology. Our solution includes domains such as the Internet of Things, LoRa, Angular JS, and Express. This may ensure that our solution will stand the test of time.

2.5 Implementation

We decided to place a small device containing a GPS module on the boats that need to be tracked. At uniform intervals, this device transmits information through a low-energy consumption network, LoRa. The information is received by a LoRa gateway, which is strategically placed to avoid or at least minimize the Doppler effect caused by the variation in the speed of the emitting devices. Then, these gateways transmit all the data received to the LoRaServer located near the base. To avoid using long cables, we decided to use Wi-Fi as the means of communication between the gateways and the LoRaServer. At this stage, our LoRaServer analyzes and interprets all the data so it can be readily used by our front-end application. Figure 3 presents a simple overview of the project architecture.

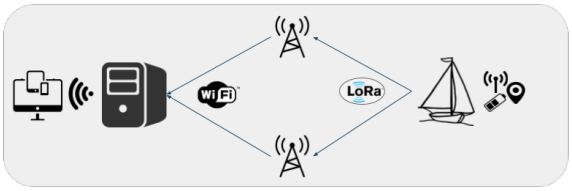


Figure 3: Schema of our global implementation.

This solution was thought to be the best due to its relative simplicity and adaptability. The association should be able to use our system anywhere and only needs to choose the proper locations for the gateways. Because many devices will eventually be required (up to 100), each low powered device has an autonomy of approximately 62 h. We would have preferred to dispense with the GPS module to decrease the energy consumption; however, geolocation via triangulation with LoRa is very inaccurate. There is a link between the accuracy of our tracking and the global cost of the devices. Our solution ensures reasonable accuracy while being affordable in the long term.

2.6 Project management

We divided the project into three main parts: the electronic device, the network, and the server/web application. Based on our domains of expertise and the skills that we wanted to develop by working on this project, we each chose which part of the project we wanted to work on. In addition to the following division, Axel Chauvin was in charge of managing the project.

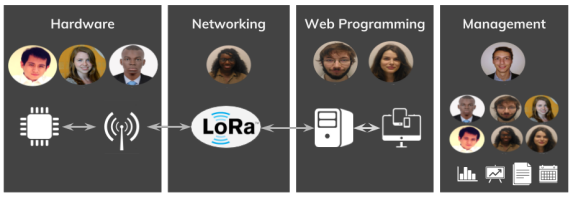


Figure 4: The subgroups and task allocations in our project.

Even though we had different areas of responsibility, all three parts of the project were highly dependent on each other because they needed to be able to communicate. Therefore, we chose to carry out most of the planning of the project with all the members of the group present.

We used the Agile framework Scrum to plan and manage our project. According to Scrum, the time frame of the project was divided into several sprints that we executed one by one. We decided that the duration of a sprint for our project would be one week, and therefore this enforced a weekly meeting to plan the upcoming sprint.

2.7 One solution among many

This project is one solution of many that already exist for the same needs. The examples below are some of the most well-known solutions.

Dimension Data is a company working with various technologies and digitalization. For the 103rd edition of the Tour de France, they designed and developed a connected object to allow fans to follow the entire race. At the departure for the first day of the three-week event, every bike in the peloton was fitted with a device.

The GPS telemetry device has a carbon fiber casing, holds a chipset and a battery, weighs just 60 g, and fits onto the underside of the riders' saddles. It sends its coordinates every second using a radio frequency. These data are used to plot each rider's position as the race evolves and to convert the information into speed, distance travelled, and the gradient of the slope they're on. From the data, other information, such as live weather conditions from the position of the riders, can be derived. Therefore, if a rider breaks away from the peloton, it is possible to show the specific weather conditions for that particular coordinate in real time. Dimension Data made this information available through their own Tour de France website but also passed it on to broadcasters, who then had the ability to provide it to their viewers via on-screen graphics. The system was very useful to the commentators.

The TracTrac [3]company has also found a solution to this type of problem. They give visibility to racing sports, engage the spectator, and increase the fan base and sponsor value by revealing and visualizing the key moments of hard-to-follow-sports in real time.

Finally, Sigfox[2] has its own device for race competitions: a 50-g beacon that transmits the GPS position of the user via the Internet on a dedicated interface. It does not require synchronization or a Bluetooth-Wi-Fi-sim card. The users only need to have it on them and everything is handled by Sigfox, thanks to its low-rate and long-range technology.

2.8 Business Model

2.8.1 Market analysis

In France, sailing represents a market with high potential. According to the French Sailing Federation[4], in 2013, there were more than 290,000 licensees in 1074 clubs throughout the country. Each year, they welcome 1,200,000 persons and their overall revenues are estimated to be €131,000,000. These figures may seem impressive and show that sailing is a popular sport; however, 21% of sailing clubs have a budget lower than €25,000. Therefore, we can segment the market into two categories: approximately 225 clubs with low budgets and approximately 850 with more substantial resources.

2.8.2 Strategic analysis

To assess the market, we will use a SWOT matrix. First, the strengths of our business are clear: we provide a low-cost and simple solution using cutting-edge technologies. The budget is a major constraint for our project; therefore, we need to optimize the cost of the material. For this purpose, and to ensure that our product is of high quality, we sought the best technical solutions to make the solution forward-looking and easy to use. Moreover, our team is composed of very talented people with diverse skills and knowledge.

Regarding our weaknesses, we do not have specific knowledge or experience in the sport of sailing. Yet to make our product a success, we need to precisely meet our customer's needs and to adapt to the constraints of the sport.

We want to establish our company in a market with a large number of potential clients.

One of the main difficulties arises from the technologies we use. These are newer technologies; therefore, they lack maturity and the standards battle could make our solution obsolete.

In the light of the market and our solution, we want to first target limited-budget sailing clubs. This segment represents a substantial number of clients, and therefore it represents a real opportunity to enter the market.

Other competitors offer different solutions in this market. The solution provided by GeoRacing[5] is particularly impressive. It adjusts to numerous activities, such as sailing, gliders, and car rallies, and possesses extremely complete services. For example, it is used during the "VendéeGlobe" race. The solution provided by TracTrac is less comprehensive; however, it corresponds to our product. What distinguishes us is the price. The sailing clubs that we target do not have the financial resources to buy the type of service provided by TracTrac. Therefore, we can offer a low-cost solution that is still fitted to their needs and is easy to use for the organizers and the public.

2.8.3 Operational marketing

This section provides details concerning our value proposition and its deployment through our marketing mix.

We developed a comprehensive solution composed of different technical parts and services to most precisely meet the needs of our potential clients. This includes the network infrastructures, the electronics casings meant to be mounted on the sailboats, and the Internet platform corresponding to the user interface. In addition, we offer to deploy the network and to provide the documentation to reproduce our prototype. We want to provide a minimum valuable product: our product has basic functionalities to ensure minimum value creation but provide sufficient value to our clients. We hope that our system will be enhanced by a future group of students taking into account our customer's feedback to develop new functionalities.

Due to the customer segment that we would like to target as a start-up, our solution needs to be the cheapest on the market. Our closest competitor being TracTrac, we need to establish the price of our product and its services accordingly. For the unit price of the casings, a price of €100 would be 33% cheaper than the competitor's solution. We would use economies of scale, component optimization, and negotiation with our providers to reach an appropriate manufacturing cost.

Concerning the distribution, we would be directly connected to the sailing clubs. This direct marketing without intermediates would ensure that we capture the maximum value and have a privileged relationship with our clients. The integrated distribution would therefore have both economic and relational benefits. This would allow us to have more financial stability and to develop a product close to our clients' needs.

Promotion would be a critical criterion for the success of our project because potential clients would need to be aware of the existence of our solution and its advantages. During the introduction and growing phases, we need to have a very focused communication targeting sailing clubs in our market segment. Even though this solution is time-consuming, it would allow us to have a real impact on prospects establishing a direct contact. We would also participate in technological fairs to publicize our product.

3 Technical choices and implementation

IoTracking is a project mixing several domains, including electronics, networking, and programming. This section will discuss the global architecture of the technological aspects and each part shown below will then be discussed more precisely one by one.

- The device section discusses the choices of electronics components, their assembly, their code, and how they communicate with the LoRa network and the specific tests.

- The network section discusses the choice of the open-source LoRaServer and how to make the network communicate with the devices and the application server.
- The software section discusses implementing the server and the application with the database and how they communicate with each other and the network.

3.1 Hardware

3.1.1 Devices

Our project required a device that could communicate with a server and that could locate itself on the lake. We chose to use LoRa for communicating and GPS for locating, and therefore we needed a LoRa module and a GPS module. Being set in a humid environment, the casing of the device had to be waterproof according to the IP66 standard.

After researching the different components, we found an electronic card from the Dutch company Sodaq. The card, called LoRaOne, came embedded with everything we needed. In addition, the price of the card was not more expensive than if we had bought all the components separately. Using this card also saved us assembly time.

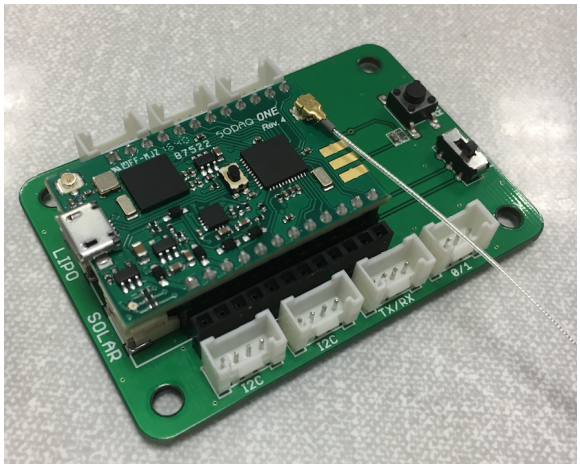


Figure 5: LoRaOne from Sodaq.

Embedded on the LoRaOne card are the following components:

- A GPS module, uBlox EVA 7M;
- ATSAMD21G18, an Arduino compatible microcontroller;
- A LoRa module, Microchip RN2483 Module; and
- An accelerometer/magnetometer LM303.

After conducting several tests using LoRa and triangulation to locate the devices, we realized that this technique did not have sufficient precision for our application. Therefore, we chose to use GPS. With the GPS module embedded on the LoRaOne, we have a precision of 5 m for our location. The GPS module embedded on the LoRaOne has low-power consumption, and therefore it is ideal for our application.

We use the LoRa module to communicate with the LoRa gateways, and via these gateways we can access the server that runs the system.

The accelerometer is a component that we originally did not think we would need for our device; however, we found a great use for it. To eliminate the need for an on/off switch mounted on the surface of the waterproof casing, we chose to use the accelerometer to wake the device up on an interrupt. Using a sleep mode instead of actually turning the device off will have a minimal impact on the power consumption of the device because it consumes very little energy when the CPU goes into sleep mode. The device returns to sleep mode at the end of each cycle.

We configured the accelerometer so that it will generate an interrupt if it detects a movement in x- or y-axis. We set the sleep mode of the CPU to be in STANDBY mode. The CPU can then be woken by asynchronous interrupts, i.e., the external interrupt from the accelerometer.

Because the microcontroller is Arduino compatible, we were able to use the Arduino IDE and all the libraries that already exist for the different Arduino boards.

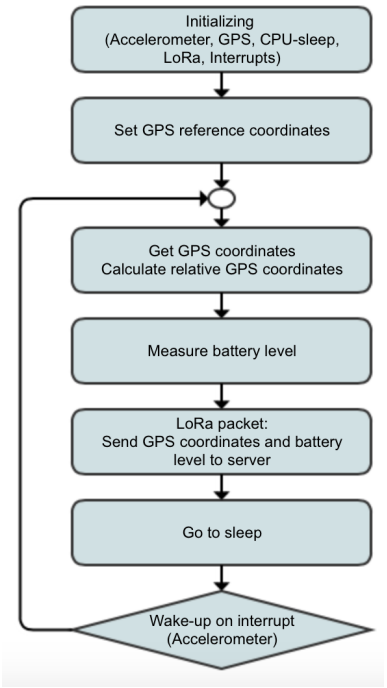


Figure 6: Algorithm for the device.

Due to the limited number of bits we can send in each LoRa packet, we decided to decrease the size of the GPS coordinates sent from the device. To do this without losing precision, we use relative coordinates instead of absolute coordinates. On wakeup, the server will therefore send four GPS coordinates to the device so that it can create an area of reference. Before sending the GPS coordinates that the microcontroller obtains from the GPS module, we map the GPS coordinates to Cartesian coordinates using an approximation that is valid for short distances, as shown in Figure 7.

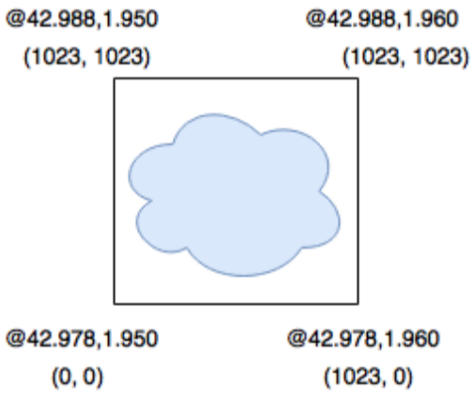


Figure 7: Absolute GPS coordinates for Montbel Lake.

Currently, the absolute GPS coordinates of Montbel Lake are hard coded into the device prototype. This can be changed in a final version of the product, where the coordinates could be sent to the devices from the server in the case of a change of location. The primarily reason for this simplified solution is that the LoRa communication at present is uni-directional, and therefore we cannot currently send messages to the devices.

Result: x, y : 10 bit integers
south, north, east, west : 32-bit float;
long, lat : 32-bit float;
$x \leftarrow (long - west)/(east - west)$;
$y \leftarrow (lat - south)/(north - south)$;

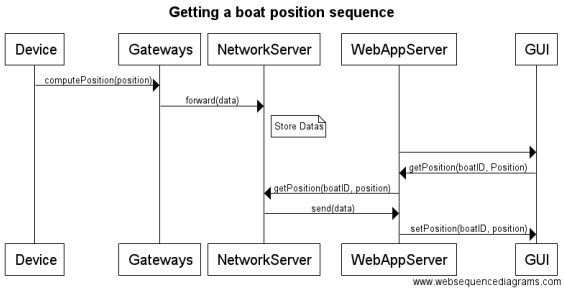


Figure 8: Sequence diagram of the operation for obtaining the boat positions.

The device will also send its current battery level to the server. This way the user can monitor whether or not the device has to be charged before a race.

The device sends one LoRa packet to the server every 10th second, that contains the GPS coordinates and the battery level. Each packet contains 3 bytes of useful information excluding the headers. The 3 bytes contains 10 bits for the relative longitude, 10 bits for the relative latitude and 4 bits for the battery level.

We did an estimate of the power consumption for our device.

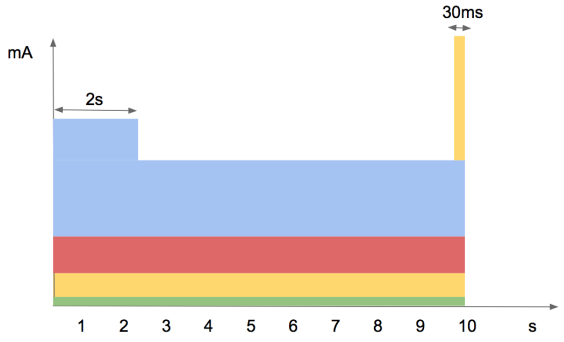


Figure 9: Estimate of the power consumption

- The green part corresponds to the accelerometer that consumes 300 uA when in use.

- The yellow part corresponds to the LoRa module that consumes 2,8 mA, except for when it is transmitting and it rises to 38,9 mA. This value is extra high because we chose the highest TX power setting possible. One transmission takes approximately 30 ms.
- The red part corresponds to the microcontroller that consumes 2,8 mA.
- The blue part correspond to the GPS module that consumes 16,5 mA when in tracking mode and 21 mA when in acquisition mode. The time of acquisition may vary in function of the signal strength (often low when inside of buildings), but we have estimated an average of 2 s.

$$p = \frac{1}{T}\sum p_i t_i$$

$$\frac{1}{10}(4,5mA \times 2s + 36,1mA \times 0,03s +$$
$$(16,5 + 2,8 + 3,64 + 0,3)mA \times 10s)$$
$$= \frac{1}{10} \times (9 + 1,083 + 232,4)$$
$$= \frac{1}{10} \times 242,483 = 24,2483mAh$$

We found that the total power consumption for the device would be 24,2483 mAh. We bought a Lithium-Ion battery with a capacity of 3Ah, which gives our device an autonomy of approximately 124 hours. This will give the associations the possibility to use the same device for several regattas without having to charge it all the time.

3.1.2 Gateways

For the gateways, we decided to use Chistera Pi[6] and Raspberry Pi to build a single channel LoRaWAN gateway[7]. This is a low-cost solution for a LoRa gateway and costs €72.

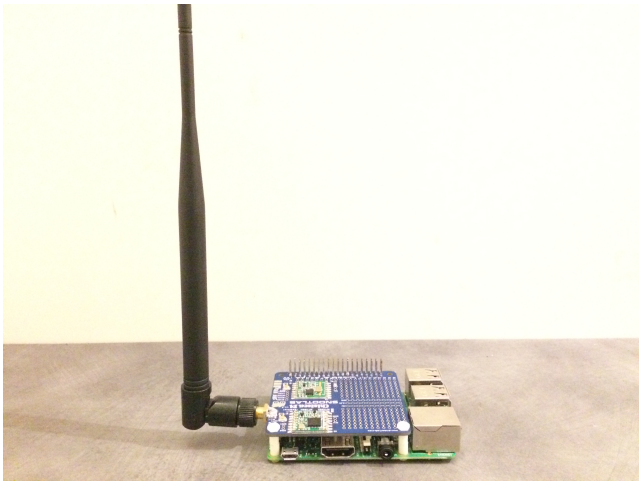


Figure 10: System schematic.

Chistera Pi is designed by Snootlab and is based on a Semtech SX1276 Transceiver: the HopeRFM95W radio module.

The RF transceiver module RFM95W features the LoRa TM long-range modem, which provides ultra-long range spread spectrum communication and high interference immunity while minimizing current consumption. Using HopeRF's patented LoRa^{TM} modulation technique, RFM95W can achieve a sensitivity of over -148 dBm using a low cost crystal and various components. The high sensitivity combined with the integrated +20 dBm power amplifier yields an industry leading link budget, making it optimal for any application requiring range or robustness. LoRa^{TM} also provides significant advantages in both blocking and selectivity over conventional modulation techniques, solving the traditional design compromise between range, interference immunity, and energy consumption.

To manage the Chistera Pi from the Raspberry Pi, we use the original SX1276 library supplied by Semtech.

The gateways are one of the most essential parts of the project. The devices can only communicate with the gateways because they both use the same technology and transmit and receive in the same frequency spectrum. The gateways then relay the data to the LoRaServer.

- The protocol[7] between the gateway and the server is purposefully very basic to be used for demonstration purposes or on private and reliable networks. There is no authentication of the gateway or the server, and the acknowledgement are only used for network quality assessments, not to correct the UDP datagrams losses (no retries).

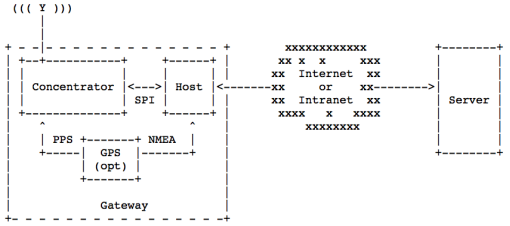


Figure 11: System schematic.

- Upstream protocol

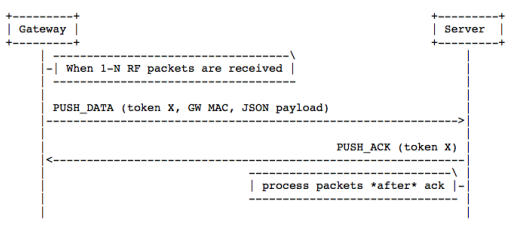


Figure 12: Upstream Sequence diagram.

- Downstream protocol

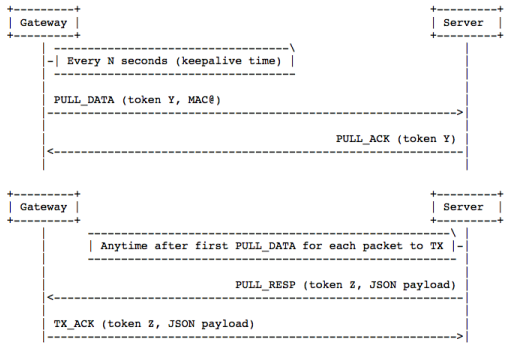


Figure 13: Downstream Sequence diagram.

- Positions
  The position of the gateways over the areas to be covered is a very important choice. If we place the gateways without taking into account the trajectory of the boats, there will be a significant loss of data. Therefore, we placed our gateways perpendicular to the direction of the movement of the boats to minimize the Doppler effect.

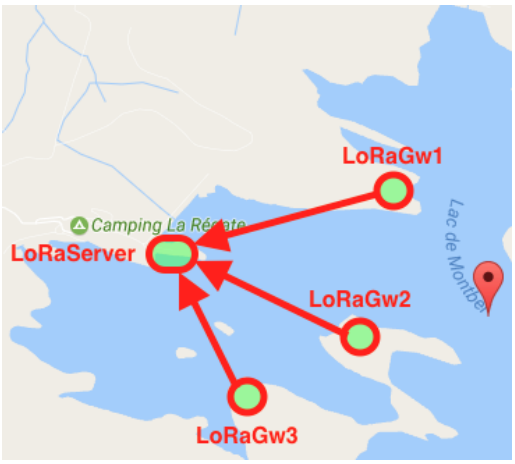


Figure 14: Gateways on the maps.

3.2 Networking

3.2.1 LoRaServer

To make the correct choice regarding the network server, we had to take several constraints into account. The network needs to cover the entire Montbel Lake. Our devices also need to have low-power consumption. In addition, our solution is intended for people who are not network experts. Therefore, we need to provide an easy to deploy and scalable network infrastructure. After studying many solutions, we decided on LoRa because it has the characteristics that are required for our project. Realistically, we do not have enough time to implement an entire server that is capable of receiving data from the LoRa gateways ourselves; therefore, we are using an open source LoRaWAN network-server that is compatible with our LoRa devices and was developed by Broocar. It is available at [8].

The LoRa network server is composed of three main elements that communicate with each other.

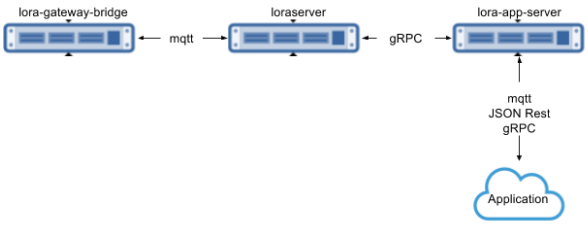


Figure 15: LoRaServer components.

- First, the LoRa-Gateway-Bridge allows our devices to communicate with the main LoRaServer. It converts the UDP communication protocol used by the LoRa gateways to JSON over MQTT. This enables us to use MQTT to receive data from and send data to our gateway via publish/subscribe messaging transport. To run the LoRa-Gateway-Bridge properly, we need an MQTT server. We are using Mosquitto, an open source MQTT server.

- Second, the LoRaServer is responsible for handling uplink data received by the gateways and scheduling downlink data transmissions. It stores all session-related and non-persistent data into a Redis data store. Therefore, we needed to install a Redis server.

- Finally, the LoRa-App-Server will provide a RESTful API to communicate with our web application. It is responsible for the node "inventory" part of the LoRaWAN infrastructure and handling received application payloads and the downlink application payload queue. It comes with a web-interface and an API (RESTful JSON and gRPC, respectively). The received payloads are published over MQTT and can be queued using MQTT or the API.

These three components use gRPC (an RPC framework) for inter-component communication and allow the web application to use the data provided by the devices.

3.2.2 Wi-Fi local network

Wi-Fi technology is used in our project to ensure communication between the backend server and the gateways. We decided to use Wi-Fi because it is a technology that we are familiar with and that is easily to configure. In addition, Wi-Fi already has a frequency spectrum dedicated to it by ARCEP and we will not be breaking any regulations. Further, Wi-Fi allows the coverage of a larger area compared to other technologies such as Bluetooth or ZigBee. This will ensure the portability of our solution if we move to a different location. Furthermore, it is easier to find hardware related to Wi-Fi, such as routers and modems. Because Wi-Fi is widely used, most computers, smartphones, and devices should already be equipped with the technology. We did not choose to interconnect our equipment with cables because we would need very long waterproof cables, which is not economically viable compared to using Wi-Fi technology.

Figure 16: The antennas and modem used.

Initially, we wanted to order and buy antennas specific to our needs; however, Mr. Monteil gave us three antennas to use for free. These antennas can cover a large area because they are able to emit at the required power level. Two of the antennas are directional while one is omnidirectional. We were also given a modem to connect the antennas to our gateway or server. If we were to buy our own antennas, they would have cost at least €70-100 each. Because we are only trying to prototype our solution, the antennas supplied by Mr. Monteil are sufficient. Once we are sure that our solution works, we will buy antennas that have more suitable characteristics for our needs.

The idea is to strategically place Wi-Fi antennas at various points around the lake to ensure good coverage. The challenge is that the required coverage area is very large. Therefore, we need to conduct radio planning to identify strategic placement points. We will install the omnidirectional Wi-Fi antenna at the base to serve the servers and the users. There will then be a directional antenna next to each gateway to allow it to relay its data to the LoRaServer at the nautical base. We need to configure the antennas so as to not interfere with the signals of the other antennas.

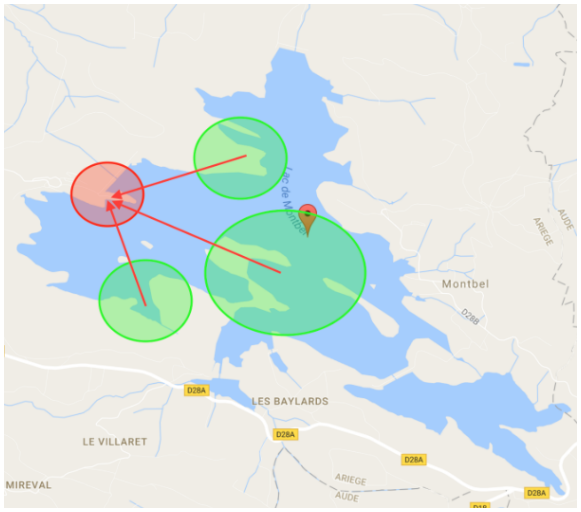


Figure 17: Where to place the antennas around the lake.

Figure 17 shows how we will place the antennas. The red circle represents the nautical base where the omnidirectional antenna will be situated. The green circles represent approximately where the gateways and the directional antennas will be situated. The red arrows represent their directions.

First, we will ensure that the communication between all the components works correctly using an Ethernet cable. Once we are sure that the communication is stable, we will try to reproduce the same results using Wi-Fi. Then, we will need to measure the signal strength of each antenna at the nautical base. Depending on the signal strength, we will need to readjust their positions and directions. Ideally, there should not be any communication errors caused by the Wi-Fi links.

3.3 Software

3.3.1 Server

Architecture We want to build our web server as a set of web services that can be used by any type of client. This is called Service Oriented Architecture (SOA). The reason behind this choice is that we want to make it possible to expand or create new applications without having to make changes on the server.

There are two main types of web service architecture styles (SOAP and REST) and two main types of data exchange formats (JSON and XML).

SOAP/XML is a very robust protocol, and, thanks to XML schemas, is very good when there is a need to automatically ensure the validity of the data exchanged. However, it is less practical to use on the client side and requires more development in the server, as well as writing XML schemas.

REST/JSON is a less robust protocol; however, web services can be quickly developed using these technologies. REST is easy to handle on the client side as well as on the server side, and JSON can be manipulated quickly on web clients.

The biggest constraint we had when making this choice was the development time. We had a very limited amount of time to build a prototype and we wanted to be as efficient as possible. In addition, we wanted this prototype to be easy and quick to interface with our web client (which will work with Javascript).

Database backend There are several types of database technologies. Among these are relational (often SQL databases such as MySQL, the Oracle Database) and document-oriented (called NoSQL, such as MongoDB). Both database types suit our simple needs. Figure 18 shows the data structures we used to represent our data.

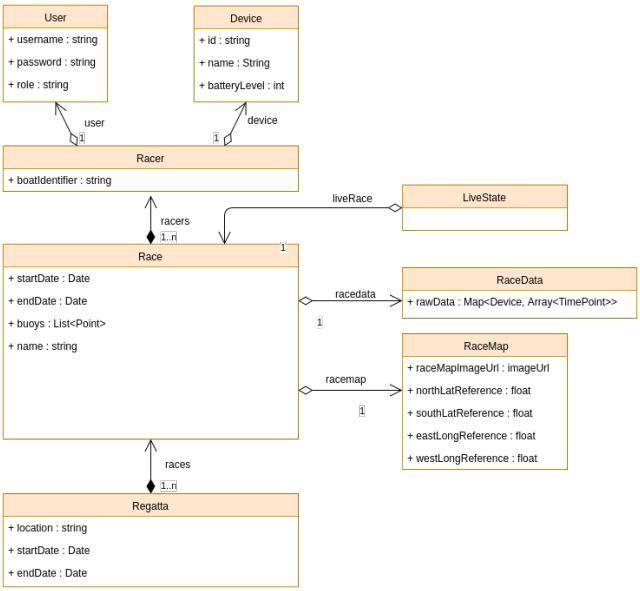


Figure 18: Server class diagram.

SQL databases are very good when it comes to ensuring data integrity and atomic transactions. NoSQL databases are known to be faster than SQL databases (if the data structures are not normalized), are scalable, and provide more flexibility.

During our academic projects, we have only used SQL databases. In this project, both database types suited our needs. Therefore, we chose to use MongoDB, which is a NoSQL database. It might not have been the best option; however, we learned how to use it and what to expect from this technology.

Available technologies There are a handful of server technologies supporting SOA in the IT world, and each of them has their own philosophy and capabilities. The technologies we considered using were:

- C# / ASP.NET;
- Java EE;
- Ruby on Rails;
- Python / Django; and
- NodeJS / Express[9].

Final choices Considering all of the options above, we chose to use a REST/JSON-based web service API, which is the fastest to build and the easiest to interact with on the client side. In addition, we chose NodeJS/Express[9] as the server technology, because it is the best choice to quickly build REST/JSON-based services.

The practical outcome of these choices is that we can share our data types between the client and the server very easily.

The web server will run on a virtual machine hosted on a physical machine managed by the CVRL association. This physical machine has a network (but not necessarily Internet) capabilities and can communicate with both the LoRaServer and the web clients.

The web server will host the REST API and the server will host the HTML client files.

3.3.2 Application - Web Client

We chose to use a new and popular technology to implement our web client: Angular2, which is a Javascript framework for building client applications developed by Google.

The web application will be compatible with any type of screen (e.g., mobile, desktop, or tablet) thanks to Bootstrap, a free and open-source front-end web framework for designing websites and web applications.

The application is divided into different groups of components depending on the type of service:

- Display upcoming regattas;
- Follow a regatta (if there is one) in real time;
- See a member's trajectory for a particular regatta;
- Edit regattas;
- Edit devices; and
- Edit maps.

We added a JWS-based authentication service to allow/restrict access to a service depending on the role of the user. We defined three types of user roles:

- Anonymous: people in the audience – only has access to upcoming regattas and real-time regattas;
- Member: member of the association - can also access previous data; and
- Staff: staff of the association - can also proceed to any edit option.

As we progressed in the implementation of our application (in parallel with the server), we tested all our views and checked the results by verifying that the elements we edited had changed as planned. We proceeded with these tests on each component after being sure that the server had already been successfully tested to ensure that if any error occurred, it would come from the application and not from the server.

4 Results

4.1 Hardware

4.1.1 Device

We managed to develop an autonomous device that can:

- Retrieve its GPS coordinates;
- Calculate relative coordinates according to a set of reference coordinates;
- Communicate with a server via a LoRa network
- Measure its battery level;
- Go to sleep; and
- Wake-up on an interrupt from the accelerometer.

Seeing that we have not been able to do tests at the actual location of the lake, we decided not to prioritize the task of finding a suitable waterproof casing for the device.

We have not been able to do exhaustive test on the autonomy of the device, and for the time being, it has only been tested with a powerbank as power source. Because of a miss match in connectors between the Li-ion battery and the LoRaOne shield, the battery can yet not be connected without being soldered directly on to the board.

We did for a long time experience problems with the RN2483 LoRa module that seemed to be defect, which is why some of the easier tasks have been put aside in order to prioritize to overcome the bigger barrier.

Nevertheless, we managed to retrieve actual GPS coordinates and send a packet every 10s containing the relative coordinates and the battery level to the gateway. We also implemented a mechanism that makes the device go into sleep mode when it is not in movement. Overall, the device is able to do what is supposed to do. The only setback would be the transmission range of the LoRa module which is not as powerful as we thought. This means that the rate of loss of packets is the greater as the distance between the gateway and the device gets bigger.

4.1.2 Gateways

We configured a single channel gateway to receive the data sent from the devices and forward it through the network by the LoRaServer. The gateway contains a proof-of-concept implementation of a single channel LoRaWAN gateway. It is able to receive on a configurable frequency and spreading factor[10] SF7 to SF12 (6 kbit/s to 300 bit/s), to send status updates and to forward data to two LoRaServers. Our results show that the gateway we have chosen for this project, more than being single channel, has non negligible losses: one package out of three is lost. To find the origin of the problem, we connected our device to the INSA LoRa network and we realized that one packet over ten was lost. We deduced that this gateway is not adapted to our architecture.

4.2 Networking

4.2.1 LoRaServer

We succeeded in establishing the entire LoRaServer with its three intercommunicating components that also communicate with our gateway and the web application server. Using the MQTT server that enables subscription to the gateway and lora-app-server topics, we received payloads sent by a device to the gateway-bridge and forwarded by the lorserver to the lora-app-server. Actually, we first authenticated our device to the lora-app-server so that when the join request sent by the device arrived, it was automatically accepted by the LoRaserver. The node could then send GPS information. At the end, data transmission between the device and our web application went well.

4.2.2 Local Wi-Fi network

We have not yet been able to test the actual architecture of our local Wi-Fi network solution at the lake. We have not yet been to the location to configure the antennas or to conduct real tests.

However, we still needed to test if it was possible for the different parts of our solution to communicate with each other wirelessly via Wi-Fi. Therefore, the first step we took was to make sure that the components were able to communicate via a wired connection where we used Ethernet cables. This test was successful and we are confident that the different parts of the solution can communicate correctly and fulfill their functions. Next, we set up a hotspot around the LoRaServer and we connected the gateway to the hotspot via Wi-Fi. We activated the Wi-Fi module on the gateway and then channeled all the data that it received from the devices to the Wi-Fi output instead of the Ethernet output. This worked without any complications indicating that the components should also be able to communicate when we use the actual antennas because the antennas are nothing but a tool to extend the Wi-Fi's range. The last condition for the actual test to succeed is for the antennas to be placed correctly around the lake to ensure functional coverage.

4.3 Software

4.3.1 Server

In this section, we will describe the server functionalities we have implemented.

REST API We successfully created a REST API that can be used to retrieve, alter, add, or delete data from our database. This API was auto-generated from our data model schemas, and the data sent through POST or PUT requests was checked against these schemas.

Authentication We used the JSON Web Token (JWT) method to authenticate our users. This is a popular method that is designed to work well with REST APIs.

LoRa data retrieval As mentioned in Section 4.2, the LoRaServer hosts an MQTT server. We used an MQTT client to connect to this server and to subscribe to the gateways' topics. Accordingly, we successfully received messages coming from the LoRaServer and were able to decode and interpret them.

4.3.2 Web Application

We developed a user-friendly application that implements the previously described services. The web application communicates with the server via its REST API. In this section, we will present the views and services provided to the different users.

Authentication The authentication is managed using a login page, where the user is asked to enter their credentials. Users are also asked to login whenever they try to access a page that requires authentication or a specific role. The token returned by the authentication system is stored in a cookie on the user's browser.

Anonymous and association members The visitor first opens the homepage, which lists the upcoming regattas, see Figure 19. By default, the user is invited to follow the live regatta (if there is a live regatta) by clicking "Live" on the top navigation bar.

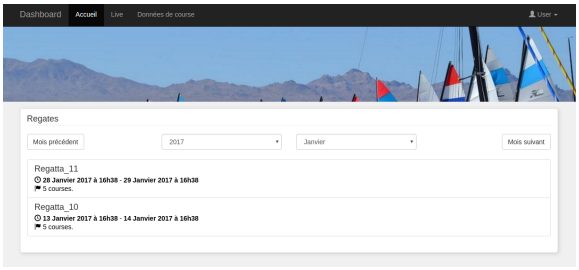


Figure 19: View of the homepage with the upcoming regattas.

By clicking "Live", the user will be able to follow in real time any current regatta. Users can choose which boats to see and if they want the boat's trajectory to be displayed on the map.

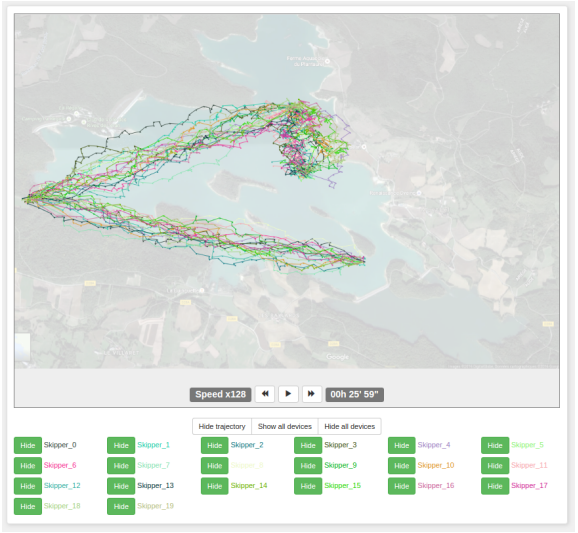


Figure 20: View of a live race.

Dashboard - for Staff members By clicking "Dashboard" on the navigation bar and then "Régates", the user will see all existing regattas and proceed to deleting, editing, or creating new regattas, as shown in Figure 21. Each regatta may contain several races, and each race may contain several skippers that participated in that race. Figure 22 presents the views of the regatta and race editing.

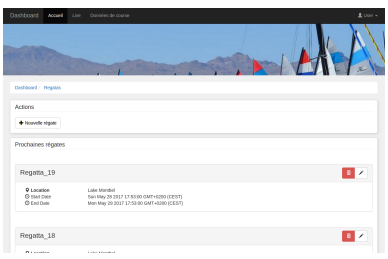


Figure 21: View of all existing regattas.

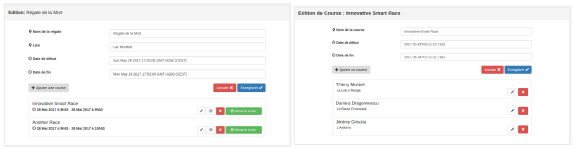


Figure 22: Views of regatta and race editing.

From the dashboard, users can also edit the devices, as shown in Figure 23. They can add, modify, and delete a device. For each device they select, the battery level will be displayed to warn the users if they are about to pick a low-battery device for a race. A container with all low-battery devices is displayed to alert users and encourage them to recharge the relevant devices. Another container informs users of all the devices that are currently active, meaning that they have been activated for a race.

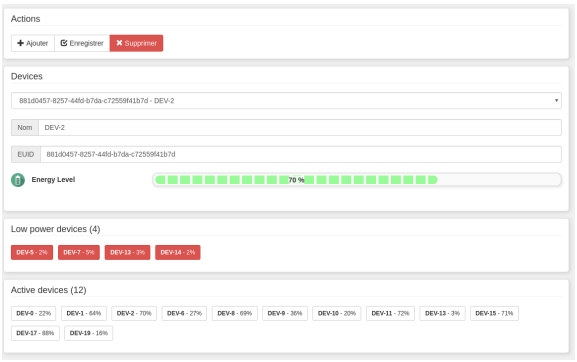


Figure 23: Views of devices editing.

5 Future Work

We accomplished a lot in a span of just three months; however, there are still many things that can be improved. In this section, we discuss different aspects of our project that can be improved in the future.

First, the use of a dual channel gateway instead of a single channel gateway should be examined. This would allow the gateways to send data to the devices. This will be useful if we need to change or reconfigure something on the device. A downlink transmission to the devices will then suffice instead of reconfiguring each device individually. The dual channel gateway could be used to send new reference GPS coordinates. Currently, the GPS coordinates are programmed into the devices and cannot be changed unless the program on each device is modified. This makes it difficult to implement our solution at different locations. Therefore, a dual channel gateway solution will ensure that our solution is mobile and reusable for different applications.

Second, an in-depth study of the realistic transmission range of LoRa as a function of its spreading factor should be conducted. This will help determine a number of things, such as the maximum number of devices possible and the size of the area needed to cover a given number of devices, and it will eventually help determine if LoRa is technically suitable in the long term. If the result of the study is negative, a different technology may need to be used.

Third, additional tests of our solution to analyze the collision rate as a function of the channel load need to be conducted. This will require having more devices. It would be interesting to understand the evolution of the collision rate as the number of device varies. The density of the devices in an area may also affect the collision rate. Therefore, this type of test will help determine the technical suitability of LoRa for this particular application. It is possible that LoRa may not be suitable for this particular application but it will be perfect for another application such as tracking trail runners.

Fourth, a load balancing system for the LoRa network could be implemented. This means that there would be an intelligent component in the system determining which channel a device or a group of devices should transmit on based on the channel load and the detected collision rate. This would allow more devices to transmit simultaneously on the same network. The signal strength and the error rate of data reception could be used as indicators to help determine the best channel to use when developing the load balancing system.

Lastly, tests on the autonomy of the device in real-life situations should be conducted. This is because we were not able to make an accurate estimation of a device's rate of usage during a regatta. We assumed that it would be active throughout the regatta; however, that may not actually be the case. Variations in the temperature are another factor that could increase or decrease the autonomy of a device. This is dependent on the location where our solution would be deployed. The autonomy of our device at this particular lake may be very different than it's autonomy elsewhere. Having a better idea of the rate of usage of the device and the variation in the temperatures it may have to endure will help determine the size of the battery that is needed for the level of autonomy required.

6 Conclusions

The goal for our project was to make regattas more interesting for the spectators that come to follow the races. We chose to use GPS to track the sailboats on the lake. We created a web application that can display the position of all the sailboats in real-time. We had to deploy two networks to enable communication between the devices and the server: a LoRa network to gather all the data from the devices mounted on the sailboats and a local Wi-Fi network to collect the data from the gateways and forward it to the server. We developed a LoRaServer and configured the LoRa gateways. All the devices require a LoRa module for communication.

We also wanted to create a video broadcast of the starting and finishing line and to store video from the jury boats to be evaluated after the race. Due to a shortage of time and the simplicity of the task, we chose not to prioritize this feature for the prototype of the product. This is something that could easily be implemented in the future.

In the end, we were not able to communicate with the device via the LoRa module; therefore, we chose to test our system with one gateway operating as a device, using a serial communication with the defective device to obtain real positions from its GPS module. This allowed us to check that the system worked. Future work would include finding another communication module for the device, so that the we can use it with all its implemented functions.

References

[1] CVRL. Presentation of the club. [Online]. Available: www.cvrl.fr

[2] Radio-Electronics^{TM}. Sigfox for m2m iot. [Online]. Available: www.radio-electronics.com/info/wireless/sigfox/basics-tutorial.php

[3] TracTrac. How does it work. [Online]. Available: www.tractrac.com/web/wp-content/uploads/2015/06/gettractracClub-sailing.pdf

[4] Ffvoile.fr. Statistiques annuelles. [Online]. Available: www.ffvoile.fr/ffv/Statistiques/V1.4/BilanAnnuel.asp

[5] Georacing. Georacing, a gps tracking and visualization system for outdoor sport events. [Online]. Available: www.georacing.com

[6] Snootlab. [tutoriel] chistera-pi et lora. [Online]. Available: forum.snootlab.com/viewtopic.php?f=59t=1512

[7] Semtech-Cycleo. Lorawan network server demonstration: Gateway to server interface definition. [Online]. Available: www.thethingsnetwork.org

[8] L. Server. Lora server documentation. [Online]. Available: docs.loraserver.io/loraserver

[9] ExpressJS. 4.x api documentation. [Online]. Available: http://expressjs.com/en/4x/api.html

[10] S. Corporation. Recommended sx1276 settings for eu868 lorawan network operation. [Online]. Available: www.semtech.com/images/datasheet/an1200.24.pdf