Open Innovation Platform for IoT-Big data in Sub-Sahara Africa

Grant Agreement Nº 687607

Agriculture MVP

LoRa Weather Station – Technical Guide

Responsible Editor:	Unparallel Innovation, Lda
Document Reference:	LoRa Weather Station - Technical Guide
Distribution:	Public
Version:	2.0
Date:	11 June 2018

TABLE OF CONTENTS

1. Introduction 4
2. Material Details 5
2.1. Adafruit Feather M0 with RFM95 LoRa Radio 5
2.2. Sparkfun Weather Shield 7
2.3. Weather Meters 8
2.4. Adafruit AM2315 Temperature & Humidity sensor 10
2.5. 1.5W Solar Panel 11
3. Main Architecture 13
3.1. Working Model 13
3.2. Data Acquisition 14
3.2.1. Weather Station Dataflow 14
3.2.2. Weather Station Library 16
3.2.3. Weather Station Data Model 17
3.2.4. Particular Specifications 18
3.2.5. Weather Station example code 19
3.3. Data Communication 21
4. Test Results 23

LIST OF FIGURES

Figure 1 – Adafruit Feather M0 with RFM95 LoRa Radio module 5
Figure 2 – Adafruit Feather M0 with LoRa Radio pinout diagram 6
Figure 3 – Sparkfun Weather Shield 7
Figure 4 – Sparkfun Weather Shield schematic 8
Figure 5 – Weather Meters Kit 9
Figure 6 – Adafruit AM2315 Temperature & Humidity sensor 10
Figure 7 – SeeedStudio 1.5W Solar Panel 11
Figure 8 – LoRa Weather Station working model 13
Figure 9 – Agriculture MVP solution workflow 13
Figure 10 – LoRa Weather Station dataflow 14
Figure 11 – OWL data model overview 17
Figure 12 – Weather Station's OWL data model ontology 18
Figure 13 – LoRa Weather Station example code 20
Figure 14 – Weather Station pushing data to Waziup platform 22
Figure 15 – Deployed LoRa Weather Station 23
Figure 16 – Power consumption testing results 24
Figure 17 – Charge Cycle for a Lithium-Ion battery 24

LIST OF TABLES

Table 1 – Adafruit Feather M0 with LoRa Radio specifications 6
Table 2 – Sparkfun Weather Shield measurements specifications 7
Table 3 – ADC arrangement values 9
Table 4 – Adafruit AM2315 sensor specifications 11
Table 5 – SeeedStudio 1.5W Solar Panel specifications 12
Table 6 – WeatherStation library resources 16
Table 7 – Communication message package 21

1. INTRODUCTION

This document describes and detail the technical information carried out in development of the LoRa Weather Station using low-cost hardware that together will communicate with the LoRa Gateway.

With an expert level architecture, the challenge was to develop a solution capable of acquire data from the surrounding area and then make it available to the Waziup Cloud Platform. The data transmission is carried out by the LoRa bi-directional communication, representing a low-cost technologic long-range communication.

This work was done in the context of Waziup Research Project, which has received funding from the European Union's H2020 Programme for research, technological development and demonstration under grant agreement No 687607.

2. MATERIAL DETAILS

This chapter detail the chosen material in order to deploy the LoRa Weather Station. The selected material can be found in the chapter 2 of the "LoRa Weather Station - Assembly Guide" document available in the Waziup GitHub repository[1].

2.1. Adafruit Feather M0 with RFM95 LoRa Radio

Figure 1 shows the “Adafruit Feather M0 with RFM95 LoRa Radio”[2] that is a portable microcontroller developed by Adafruit. This board have the ATSAMD21G18 ARM Cortex M0 processor, clocked at 48MHz and at 3.3V logic and the build-in RFM95 module to allow LoRa communication.

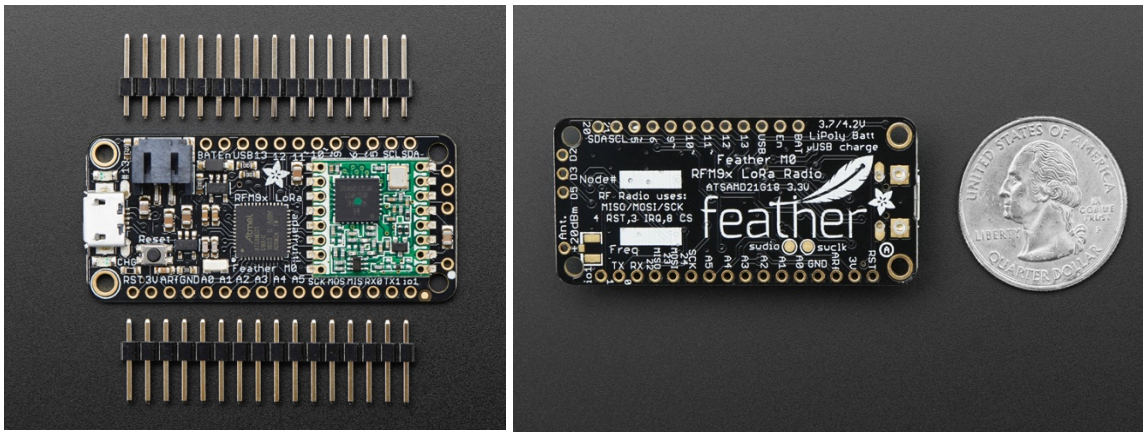

Figure 1 – Adafruit Feather M0 with RFM95 LoRa Radio module

It is necessary to add in the Adafruit Feather M0 a “uFL SMT Antenna Connector[3]” to be able to connect the “RP-SMA to uFL/u.FL/IPX/IPEX RF Adapter Cable[4]”. With these two components is possible to use the “4,5 dBi 868-900 SMAM-RP Antenna[5]”.

It has a JST jack that allow to use the “Lithium-Ion Cylindrical Battery - 3.7v 2200mAh[6]” that can be recharging via USB. When connected via USB the Adafruit Feather M0 will automatically switch to USB power when available. More specific information can be consulted in the official Adafruit product page[7]. Table 1 have the Adafruit Feather M0 specifications and Figure 2 presents the pinout diagram.

[1] https://github.com/Waziup/WAZIUP-WeatherStation/tree/master/extra/documents
[2] https://www.iot-catalogue.com/components/588b6b3f7cf42e571cf27cb3
[3] https://www.iot-catalogue.com/components/59b17a49763cfc066f6d092c
[4] https://www.iot-catalogue.com/components/59b17ab5763cfc066f6d092d
[5] https://www.iot-catalogue.com/components/58b41c348c5da2520e6b335d
[6] https://www.iot-catalogue.com/components/59b17c85763cfc066f6d092f
[7] https://www.adafruit.com/product/3178

Table 1 – Adafruit Feather M0 with LoRa Radio specifications

Measures	51mm x 23mm x 8mm
Weight	5.6g
Microcontrollers	ATSAMD21G18 @ 48MHz with 3.3V logic/power
Supply	3.3V regulator with 500mA peak current output
USB support	Comes with USB bootloader and serial port debugging
GPIO pins	x20
Hardware	Serial support, I2C support, SPI support
PWM pins	x8
Analog inputs/outputs	x10 / x1
Reset button	Yes

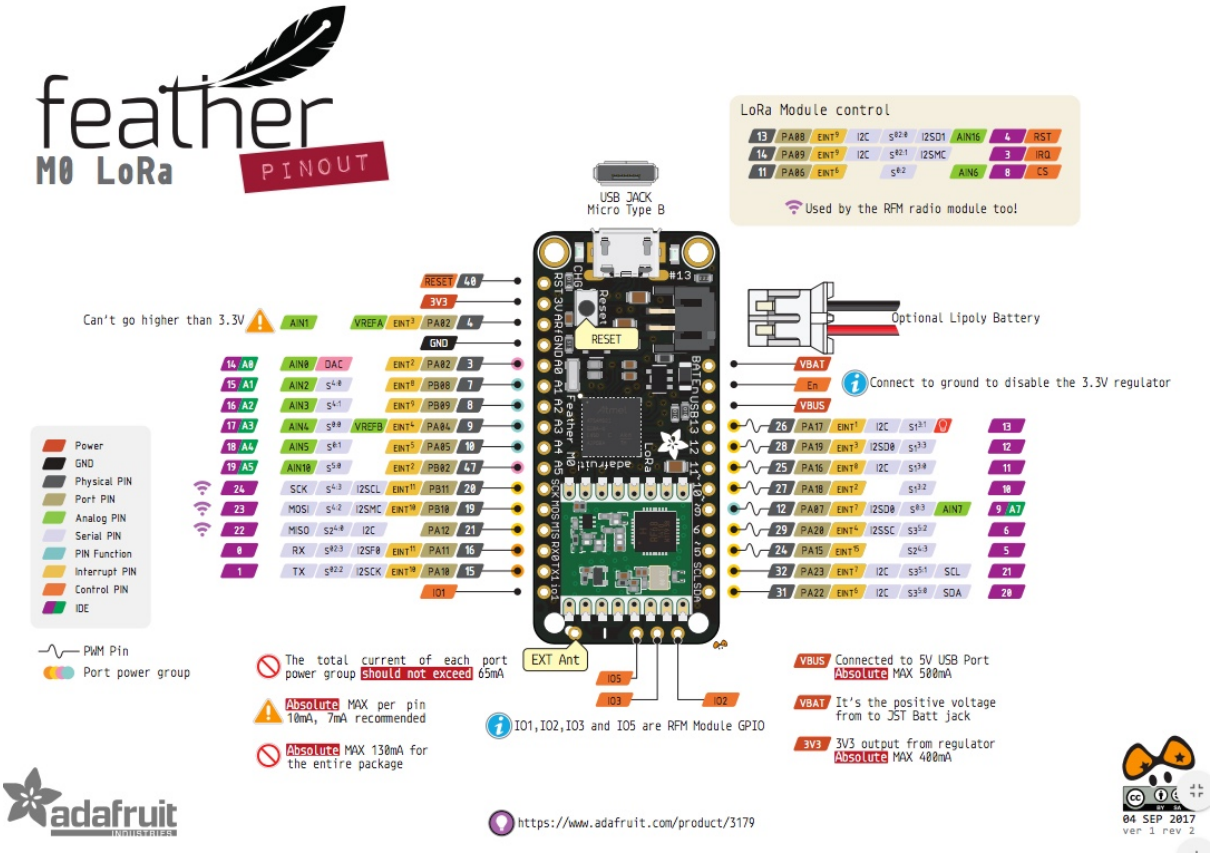

Figure 2 – Adafruit Feather M0 with LoRa Radio pinout diagram[8]

[8] https://cdn-learn.adafruit.com/assets/assets/000/046/204/original/Feather_M0_LoRa_v1.2.pdf?1504806734

2.2. Sparkfun Weather Shield

The “Sparkfun Weather Shield”[9] is an easy to use Arduino shield designed by the SparkFun that allows to acquire data typical of a metrological station such as temperature, atmospheric pressure, altitude and humidity through the internal sensors that are embedded.

Can also acquire information such as wind speed, wind direction and the amount of rain through RJ11 connections that allows the connection of dedicated sensors. For this is necessary to use two “RJ11 6-Pin Connector[10]”.

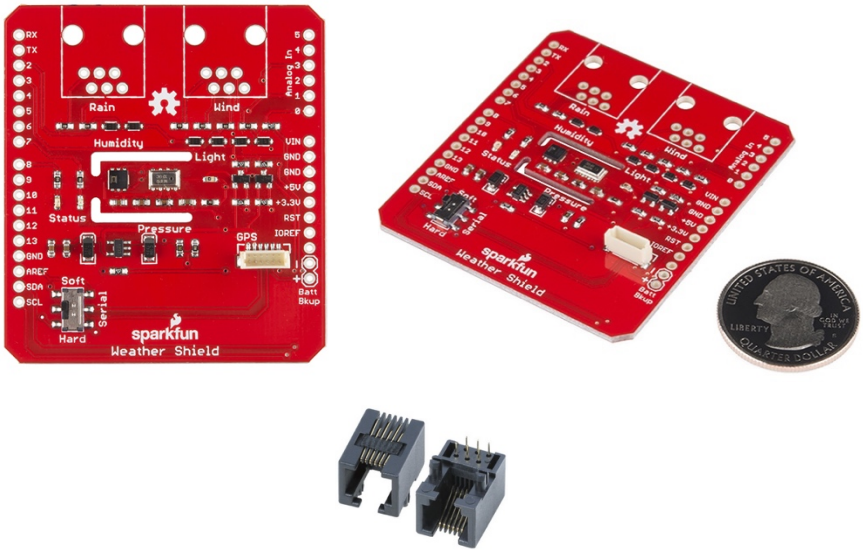

Figure 3 – Sparkfun Weather Shield

This shield can operate from 3.3V up to 16V, have built in voltage regulators, signal translators, have the Si7021 sensor, the MPL3115A2 sensor, the ALS-PT19 sensor and two unpopulated RJ11 connector spaces to connect the rain and wind sensors. To use this weather sensors will be needed the Si7021 and the MPL3115A2 Arduino libraries. More specific information can be consulted in the official Sparkfun product page[11].

Table 2 and Figure 4 respectively presents the Sparkfun Weather Shield measurements specifications and schematic.

Table 2 – Sparkfun Weather Shield measurements specifications

Temperature	Return the temperature in Celsius degrees with maximum error of 0.3ºC.
Humidity	Return the humidity in percentage with maximum error of 2%.

[9] https://www.iot-catalogue.com/components/5889db8b1a51d38d0517c1bb

[10] https://www.iot-catalogue.com/components/59b17b3b763cfc066f6d092e

[11] https://www.sparkfun.com/products/13956

Pressure	Return the atmospheric pressure in pascal with maximum error of 50Pa.

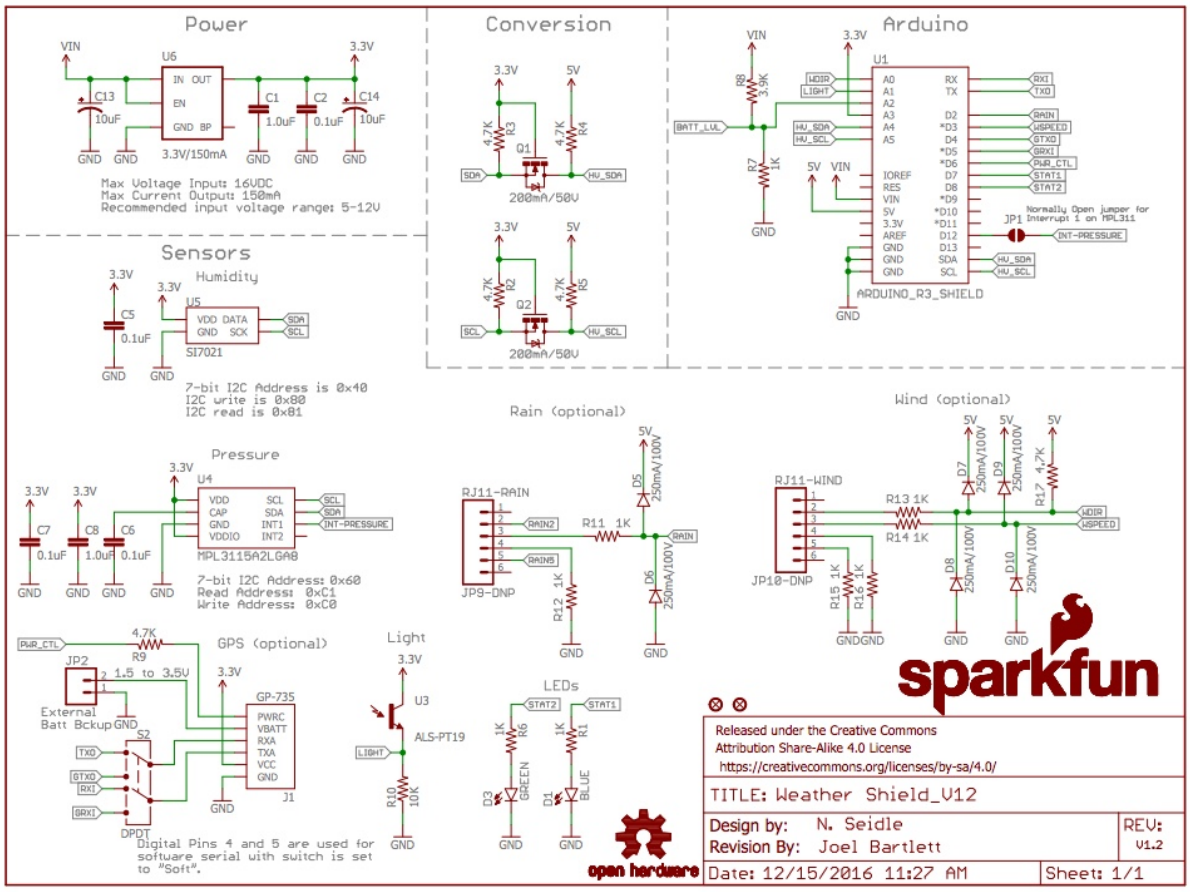

Figure 4 – Sparkfun Weather Shield schematic[12]

2.3. Weather Meters

The "Weather Meters"[13] kit has the necessary components to acquire information about wind speed, wind direction and amount of rainfall.

These sensors not contain active electronics. Instead they use switches and sealed magnetic magnets to allow easier interpret the signals. The rain gauge is a self-emptying bucket-type rain gauge which activates a momentary button closure for each 0.2794mm of rain that are collected.

The anemometer is the wind speed meter and encodes the wind speed by simply closing a switch which each rotation. A wind speed of 2.4 km/h produces a switch closure once per second.

[12] https://cdn.sparkfun.com/datasheets/Dev/Arduino/Shields/Weather%20Shield_V12.pdf

[13] https://www.iot-catalogue.com/components/5889e2851a51d38d0517c1fa

The wind vane reports wind direction as a voltage which is produced by the combination of resistors inside the sensor. The vane's magnet may close two switches at once, allowing up to 16 different positions to be indicated. The ADC arrangement is shown in Table 3. For more specification, consult the official Sparkfun product page[14].

Figure 5 – Weather Meters Kit

Table 3 – ADC arrangement values[15]

	Angle (º)	ADC with 5Volts
North	0	913
	22.5	680
	45	746
	67.5	393
West	90	414
	112.5	380
	135	508

[14] https://www.sparkfun.com/products/8942

[15] https://www.sparkfun.com/datasheets/Sensors/Weather/Weather%20Sensor%20Assembly..pdf

	157.5	456
South	180	615
	202.5	551
	225	833
	247.5	801
East	270	990
	292.5	940
	315	967
	337.5	878

2.4. Adafruit AM2315 Temperature & Humidity sensor

The "Adafruit AM2315 Temperature & Humidity sensor"[16] has the temperature sensor DS18B20 and a capacitive humidity sensor that uses I2C communication to acquire the weather measurements.

Simply connect the red wire to 3.3V or 5V power, black wire to Ground, yellow wire to your I2C data pin, and the white wire to the I2C clock pin. It is not possible to change the I2C address so only one sensor per I2C bus is available.

Figure 6 – Adafruit AM2315 Temperature & Humidity sensor

[16] https://www.iot-catalogue.com/components/58b01e40809a05801d8dd30a

To use this sensor will be needed the AM2315 Arduino library. Table 4 have the Adafruit AM2315 sensor specifications. For more specification, consult the official Adafruit product page[17].

Table 4 – Adafruit AM2315 sensor specifications

Measures	98mm x 16mm
Weight	82.64g
Supply	3.5 to 5.5V power
Wires	x4: (Vin, GND, SDA & SCL)
Temperature	Return the temperature in Celsius degrees with maximum error of 0.1ºC.
Humidity	Return the humidity in percentage with maximum error of 2%.

2.5. 1.5W Solar Panel

This "1.5W Solar Panel"[18], shown in Figure 7, is made of single-crystal material and performs high solar energy transformation efficiency at 16%. It has a fine resin surface and a 2mm JST connecter which allow to be used in several IoT projects.

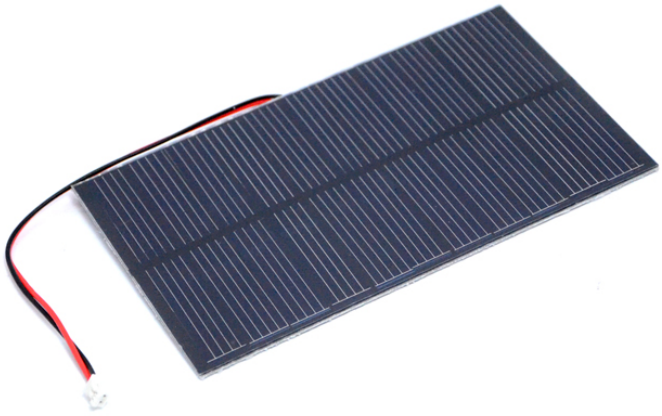

Figure 7 – SeeedStudio 1.5W Solar Panel

[17] https://www.adafruit.com/product/1293

[18] https://www.iot-catalogue.com/components/59084b279c53062850862573

For more detailed information can be consulted the official SeeedStudio product page[19] and the Table 5 presents the respective specifications.

Table 5 – SeeedStudio 1.5W Solar Panel specifications

Measures	137mm x 81mm
Weight	48g
Typical peak power	1.5W
Voltage at peak power	5.5V
Current at peak power	270mA
Efficiency	16%
Connector	2.0mm JST

[19] https://www.seeedstudio.com/1.5W-Solar-Panel-81X137-p-952.html

3. MAIN ARCHITECTURE

This chapter details the characteristics of the "LoRa Weather Station" main architecture such as the working model, data acquisition, the data communication and some other particular specifications.

3.1. Working Model

The working model of the data acquisition and supervision system it is constituted by the previously mentioned hardware, together with software capable of monitoring and supervising the sensory variables and the devices. In Figure 8 is shown the LoRa Weather Station working model.

The working model it is composed by the Adafruit Feather M0 which takes a specialized role in the system where it performs control functions through software, with processing power, enabling the sensory devices to gather data from the environment, using specific libraries.

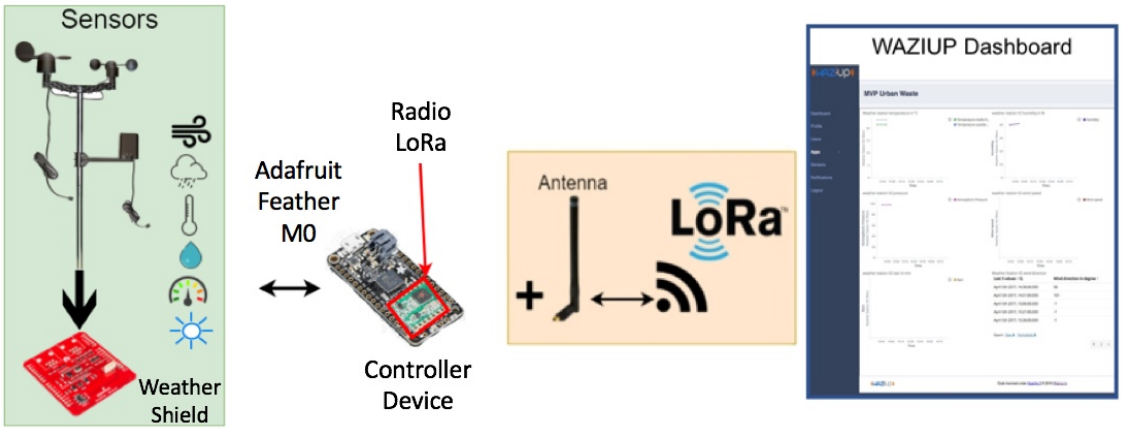

Figure 8 – LoRa Weather Station working model

Was also coupled LoRa communication capabilities in order to send the acquired weather data to the LoRa Gateway which is responsible for seeding to the Waziup Cloud Platform. Once received, all the weather data can be viewed through the Waziup Dashboard Platform. In this way, it is possible to guarantee the Agriculture MVP solution workflow represented in Figure 9.

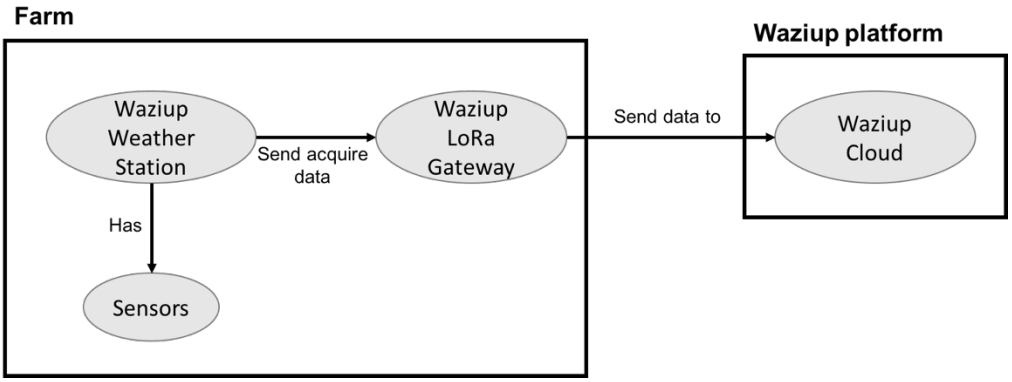

Figure 9 – Agriculture MVP solution workflow

3.2. Data Acquisition

3.2.1. Weather Station Dataflow

In Figure 10 is shown, through flowcharts, all the dataflow that occurs in the LoRa Weather Station.

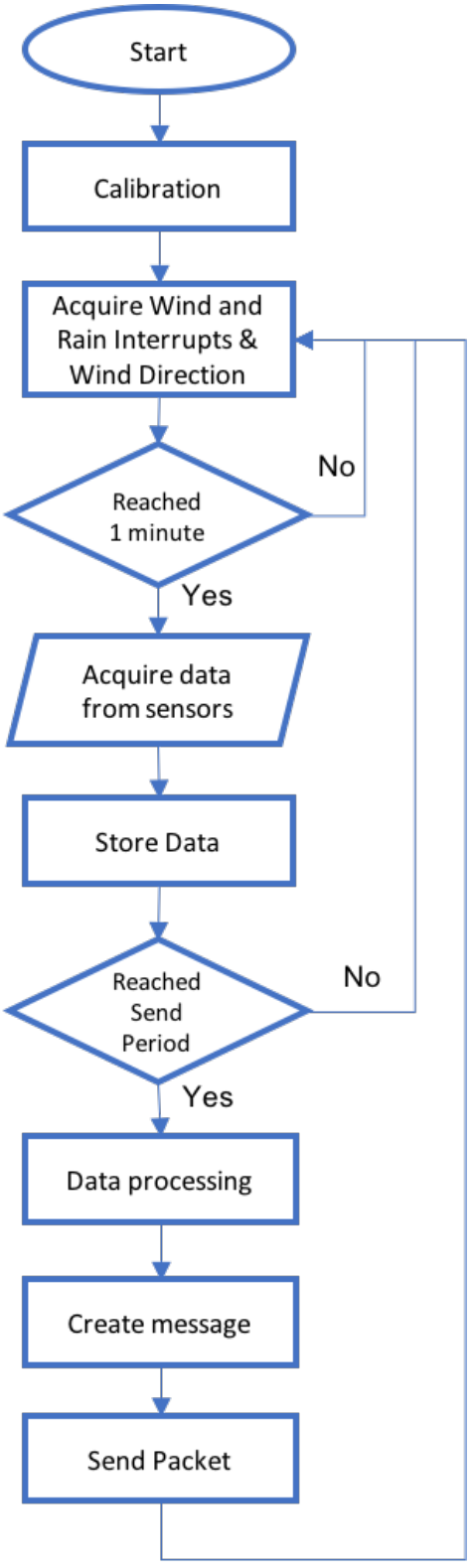

Figure 10 – LoRa Weather Station dataflow

The controller software uses the “WeatherStation” library that is a dedicated library developed for the LoRa Weather Station including several features to be able to call all the functions responsible for activating and reading the sensors signals and comply the established specifications and operate according to the flowchart presented.

When the LoRa Weather Station is connected, the sensors will be calibrated and initialized. After this task, the system will stay in a cycle in order to wait until an internal RTC interrupt occurs. This interrupt acts as an alarm and occurs always when a minute has elapsed. Through this interrupt it is also possible to control the packet to be sent.

While waiting for the internal RTC interrupt, the software is able to record the rain and wind sensors interrupts, when they occur, and every half a second also registers the wind direction to later be possible to calculate the wind tendency.

When the internal RTC interrupt occurs, the system will acquire from the sensors the temperature, humidity, pressure and the battery voltage measures and will store these data for later processing.

If the send period is not reached the mentioned process will be repeated. If the defined send period is reached, the data will be processed. In the data processing, all the measures acquired by the sensors will be processed to be available to send. In this process are performed averages for temperature, humidity, pressure and battery voltage and the respective calculations for wind direction tendency, wind speed and amount of rainfall.

Finally, the message will be created with the processed weather values and a packet with this message will be sent to the LoRa Gateway through LoRa communication. This entire process is performed for each send period defined.

The external “Adafruit AM2315” sensor is used the acquire the temperature and humidity measures. It is recommended that the sensor be positioned as explained in the section 5.2 of the "LoRa Weather Station - Assembly Guide" document available in the Waziup GitHub repository[1].

The “Sparkfun Weather Shield” is used to acquire the pressure measure and allows the interaction with the “Weather Meters” sensors to be able to acquire the rain and wind measurements.

To acquire the battery voltage value, form the Lithium-Ion battery, it has been included the resource supplied by Adafruit[20] that reads the ADC value of the “Adafruit Feather M0” pin A7 and through some calculations acquires the battery voltage. This is possible because was stuck in the “Adafruit Feather M0” a double-100K resistor diver on the BAT pin and connected to the pin A7 (digital pin D9).

Using the mentioned sensors this solution can collect the required weather data. Based on the working model, presented in the section 3.1, and the dataflow from Figure 10, it becomes clear the connection between the controller (Adafruit Feather M0) and the sensors.

This connection is established with the I2C protocol that allow a digital integrated circuit to communicate with one or more masters. It is used this type of protocol because it is only intended short distance communications within a single device and only requires two signals to exchange information.

[20] https://learn.adafruit.com/adafruit-feather-m0-radio-with-lora-radio-module/power-management

3.2.2. Weather Station Library

The "WeatherStation" library is indispensable for the correct operation of the LoRa Weather Station and is available in the Waziup GitHub repository[1]. Table 6 lists the resources and sub-libraries that are required and were developed for the LoRa Weather Station.

Table 6 – WeatherStation library resources

Adafruit_AM2315	This library allows the interaction with the Adafruit AM2315 Temperature & Humidity external sensor.
SparkFun_Si7021	This library allows the interaction with the SparkFun Si7021 Temperature & Humidity sensor.
SparkFunMPL3115A2	This library allows the interaction with the SparkFun MPL3115A2 Altitude & Pressure sensor.
SX1272	This library allows the interaction with the Adafruit Feather M0 RFM95 LoRa Radio module.
RTCInt	This library allows the interaction with the internal RTC (Real-Time-Clock) of the Adafruit Feather M0.
SensorHumidity	This library was developed to allows the interaction with the Humidity sensor. Includes all necessary functions to handle all the Humidity information. Return the Humidity value in percentage with maximum error of 2%.
SensorPressure	This library was developed to allows the interaction with the Pressure sensor. Includes all necessary functions to handle all the Pressure information. Return the atmospheric Pressure value in Pascal (or in Kilo-pascal) with maximum error of 50Pa.
SensorRain	This library was developed to allows the interaction with the Rain sensor. Includes all necessary functions to handle all the Rain information. Resort external interrupts and return the Amount of Rain value in mm of water. The work process and calculus for the Rain information are explained below in this document.
SensorRTC	This library was developed to allows the interaction with RTC sensor. Includes all necessary functions to handle all the information with the internal Adafruit Feather M0 RTC. It should be noted that the LoRa Weather Station is dependent on the internal Feather M0's RTC due to using this sensor to control the time periods. In this case, resort internal interrupts to control the measure acquisition period and the period of time to send the message.
SensorTemperature	This library was developed to allows the interaction with the Temperature sensor. Includes all necessary functions to handle all the Temperature information. Return the Temperature value in Celsius degrees with maximum error of 0.1ºC.

SensorWind	This library was developed to allows the interaction with the Wind sensor. Includes all necessary functions to handle all the Wind information. Resort external interrupts and return the Wind Direction value in degrees, the Wind Speed and Wind Gust values in km/h. The calculus for the Wind information are explained below in this document.
WeatherRecord	This library was developed to save the acquired data. It includes all the functions necessary to receive and save the data to be sent. Can return each weather measure individually.
WeatherCommunication	This library was developed to allows the communication. Includes all necessary functions to handle and configure the Adafruit Feather M0 RFM95 LoRa Radio module. Receive the acquired weather values and allow to build and send the packet with the message.
WeatherStation	This library was developed to allows the Weather Station control. Includes all the necessary functions to allow the interaction with the previously mentioned libraries and control the correct Weather Station work-flow. Allows the calibration as well as the initialization of the sensors and the acquisition of the weather values every minute and in each send period process the data and send the package.

3.2.3. Weather Station Data Model

Figure 11 presents the classes that compose the LoRa Weather Station's OWL data model as well as the relation between them and their properties.

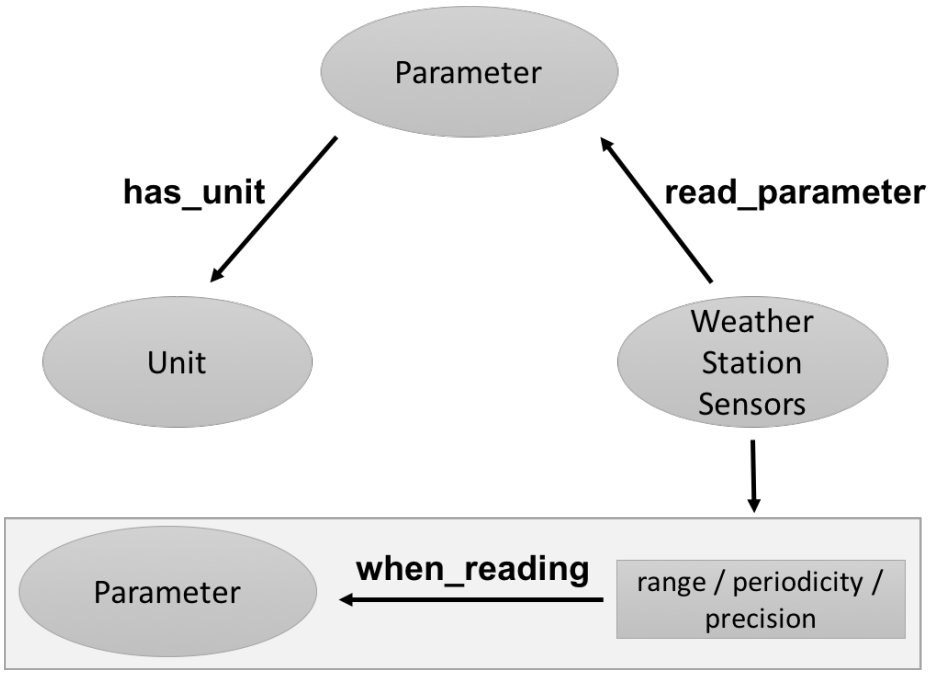

Figure 11 – OWL data model overview

From the figure, it is possible to analyse that are described three important classes. This classes are “Parameter”, “Unit” and “WeatherStationSensors”.

- The “Parameter” class describes the type of weather measurement that is associated with a unit and a sensor through the property “read_parameter” and “has_unit”.
- The “Unit” class represent the measuring unit and can have relations between them (for example “kPa_unit” is equivalent to “factor_0.001_of” “Pa_unit”).
- The “WeatherStationSensors” class represents a sensor and will have properties associated to a measuring unit (for example the Temperature sensor has a 0.1ºC precision of the Cº_unit).

Figure 12 presents the full representation of the OWL data model ontology including all the three classes and their respective properties.

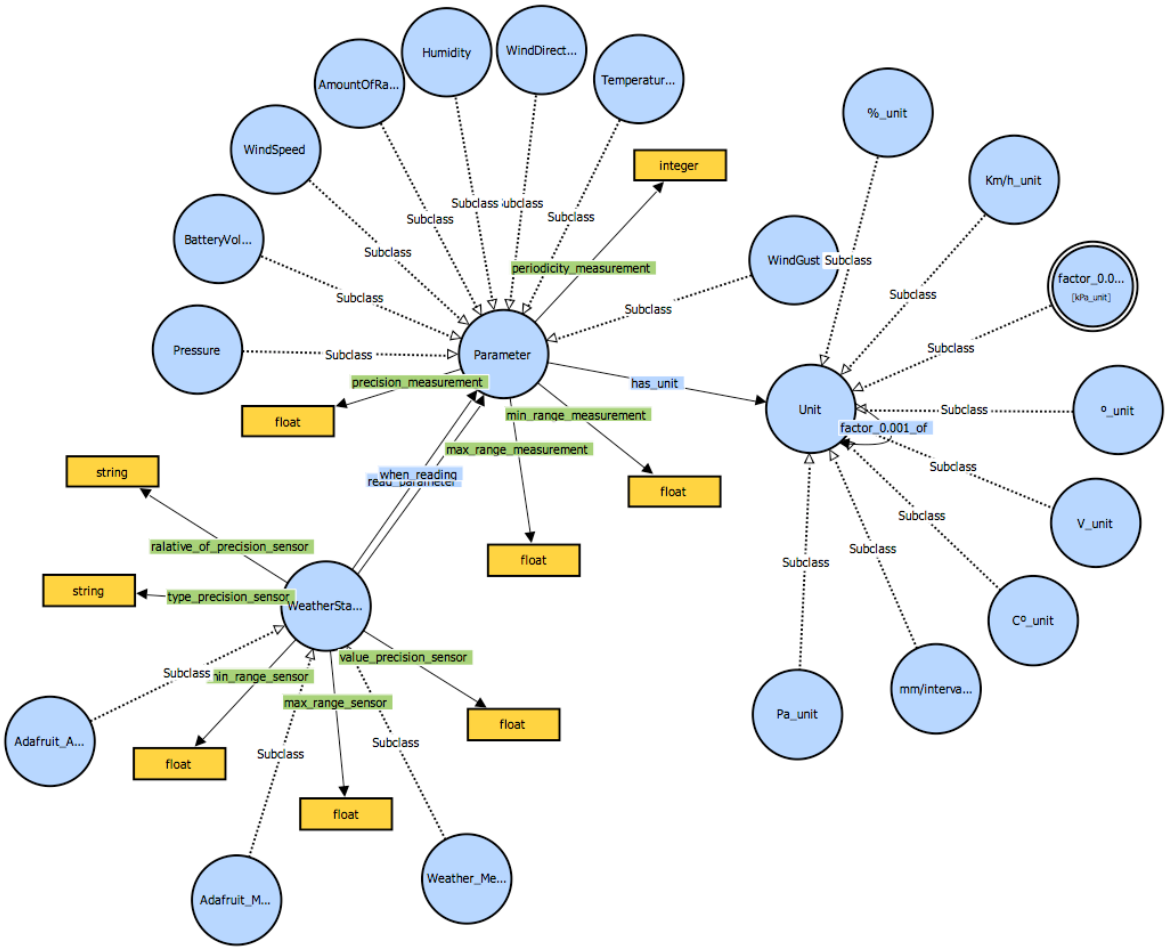

Figure 12 – Weather Station’s OWL data model ontology

3.2.4. Particular Specifications

As already mentioned in this document, the sensors to acquire values of temperature, humidity and pressure use the communication protocol I2C. To control information from the Rain and Wind sensors was necessary to use external interrupts. In this way, it was resorted algorithms in each respective library to execute the operations correctly.

As mentioned in the section 2.3 the rain gauge is self-emptying bucket-type which activates a momentary button closure for each 0.2794mm amount of rain that are collected. This means

that each interrupt corresponds to 0.2794mm of water. It was used software debouncing which means that the system will only receive interrupts every at least 10ms.

To the wind data, as mentioned in the section 2.3 too, the anemometer encodes the Wind Speed by simply closing a switch which each rotation. A Wind Speed of 2.4 km/h produces a switch closure once per second. As an example, imagine that in 1 hour occurs 100 interrupts. First is necessary to get the period between interrupts that in this case is 1 interrupt each 36s (100interrupts/3600ms). This means that the Wind Speed for this period was in average 0.066km/h (2.4km/h / 36s).

Relatively to the Wind Gust, first is necessary to get the period between interrupts and after do the same for the Wind Speed. Note that, once again, was used software debouncing which means that the system will only receive interrupts every at least 10ms. The Wind Direction value comes as a voltage which is produced by the combination of resistors inside the sensor. The vane's magnet may close two switches at once, allowing up to 16 different positions to be indicated. The ADC arrangement is shown in the previous mentioned Table 3.

Note that it is necessary to use a Schottky diode to use a Solar Panel to charge the battery as shown in section 3.3.3 of the "LoRa Weather Station - Assembly Guide" document available in the Waziup GitHub repository[1]. This diode allows that the current to go in the right direction and not to the Solar Panel.

As already mentioned the LoRa Weather Station is dependent on the internal RTC that the "Adafruit Feather M0" has because it is through this sensor that the system controls the time periods. In this case, every minute that the RTC makes an interrupt (alarm) will be acquired the weather values. When the defined period for send the message is reached, the processing of the acquired data will be carried out and the message will be constructed and sent.

3.2.5. Weather Station example code

It is not recommended to change the developed library in order to guarantee the correct operation of the LoRa Weather Station. It is only advisable to change the configuration settings found in the Arduino IDE example as shown in the following Figure 13.

To access this file simply perform the step in section 4.6 of the "LoRa Weather Station - Assembly Guide" document available in the Waziup GitHub repository[1]. It is only recommended to make changes to the field indicated in the point 1 of figure. Incorrect change in the other fields may cause issues in the LoRa Weather Station.

This Weather Station example code has by default configured a period of 10 minutes to send the packet with message. This period can be changed in field 1 for a desired time, always in minutes.

Figure 13 – LoRa Weather Station example code

The inclusion of the “WeatherStation” library as well as the creation of its object is performed in field 2. In this way, it is possible to interact with the methods created for the LoRa Weather Station workflow.

In the "setup" of this code, which is in field 3, it is called the "init" method that was developed with the purpose of initialize the sensors and the necessary resources for the "Weather Station".

In the field 4 is defined the “loop” of this code, where it is always called the “task” method. It is through this method that all the other methods and sub-methods are called in order to allow, with success, the Weather Station workflow. This method can wait until the alarm (interrupt) of the RTC happens every minute and then acquire the measures. At the same time, also allows to receive the interrupts coming from the rain and wind sensors and every half second registers the wind direction. When the send period is reached, it is responsible for calling the methods to process the data and send the package with the message.

3.3. Data Communication

The controller will send the weather data to the LoRa Gateway based on the information collected from the sensors and using LoRa communication. For this, the “WeatherStation” library resort to the "SPI" library to run the communication with the radio module RFM9x LoRa 868/915. Will also resort the “SX1272” library, as previously mentioned to send the message.

LoRa RF it is a radio modulation format that gives significantly longer range than straight FSK modulation. Is a technique that compete with other technologies that significantly improves the receiver and as with other spread-spectrum modulation techniques uses the entire channel bandwidth to broadcast the signal, making it robust to channel noise and insensitive to frequency offsets caused from the use of low-cost crystals. For more information about the LoRa communication can be consulted the LoRa Alliance website[21].

The data will be collected according to the time periods described above. Once the send period has been reached, the package will be sent with the message containing the information collected. The typical message sent to the LoRa Gateway is composed in following format:

TP;18;HU;85;PA;101.10;WD;90;WS;5.55;WG;21.24;RA;0.55;BV;4.19;

The following table outlines the type and content of the information sent in each message package:

Table 7 – Communication message package

TP;18;	Temperature
HU;85;	Humidity
PA;101.10;	Atmospheric Pressor
WD;90;	Wind Direction
WS;5.55;	Wind Speed
WG;21.24;	Wind Gust
RA;0.55;	Amount of Rain
BV;4.19;	Battery Voltage

In that way, the LoRa Gateway receives this message by post-processing python scripts to be able to correctly and sequentially inspect all the information, in order to post in the Waziup Dashboard platform the values to be appreciated.

[21] https://www.lora-alliance.org/

Figure 14 shows, as an example, the received data on the Waziup platform from the LoRa Weather Station deployed on the top of Unparallel Innovation's office building. More information on how to develop and build the LoRa Gateway can be obtained by consult the official Waziup tutorial[22].

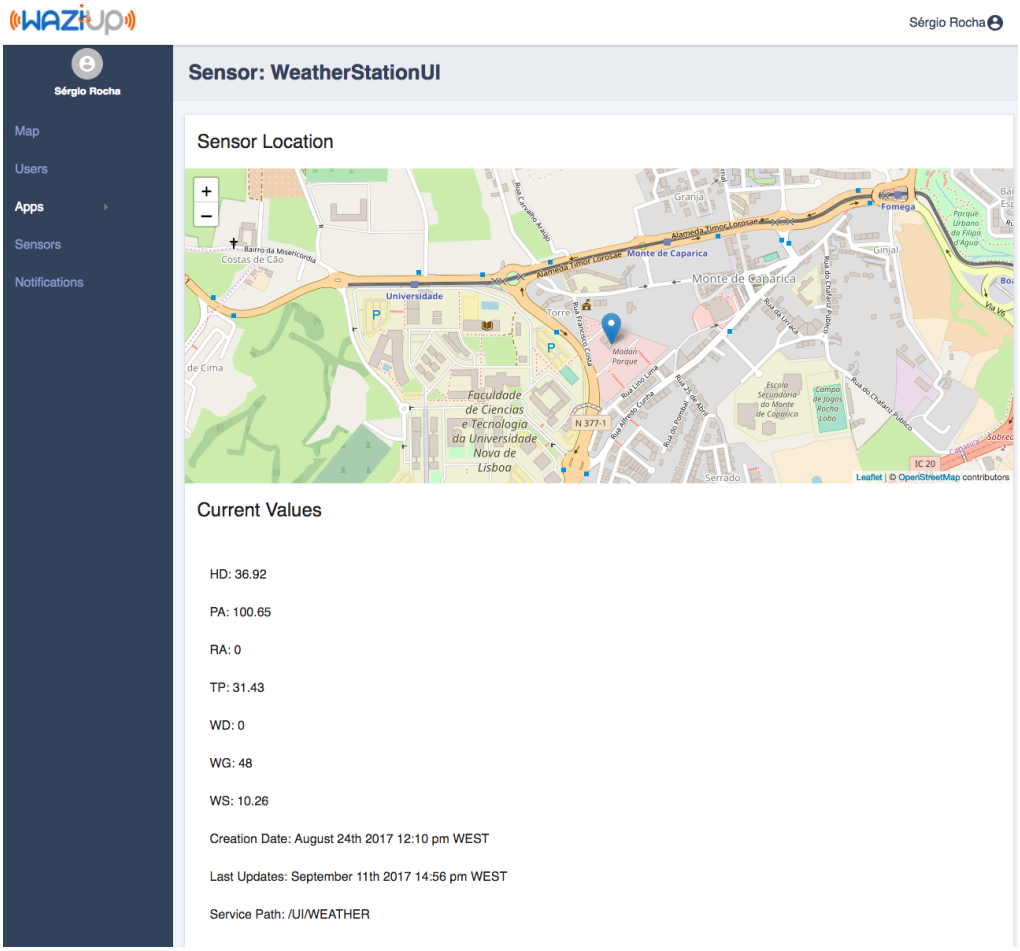

Figure 14 – Weather Station pushing data to Waziup platform

[22] http://www.waziup.io/tutorials/

4. TEST RESULTS

A LoRa Weather Station was deployed on top of the Unparallel Innovation's office, as shown in Figure 15, in order to test the prototype's operation and also to obtain consistent results about energy consumption. Note that their operation behaviour will never be exactly the same due to the weather conditions of the place where the LoRa Weather Station is deployed.

Figure 15 – Deployed LoRa Weather Station

To help in this test, an "INA219 High Side DC Current Sensor"[23] was also added between the battery and the "Adafruit Feather M0" to be able to measure the voltage and current of the battery. This sensor also helped to select the respective solar panel to be used in order to guarantee that the "LoRa Weather Station" solution was energetically autonomous.

In Figure 16 are presented the obtained results over the course of a day. Is presented in "a)" the battery value (in V) from the "Adafruit Feather M0" according to section 3.2.1, in "b)" the battery value (in V) acquired by the "INA219 sensor", in "c)" the maximum battery current value (in mA) and in "d)" the minimum battery current value (in mA).

[23] https://www.iot-catalogue.com/components/58f76c0eb276e500241b9767

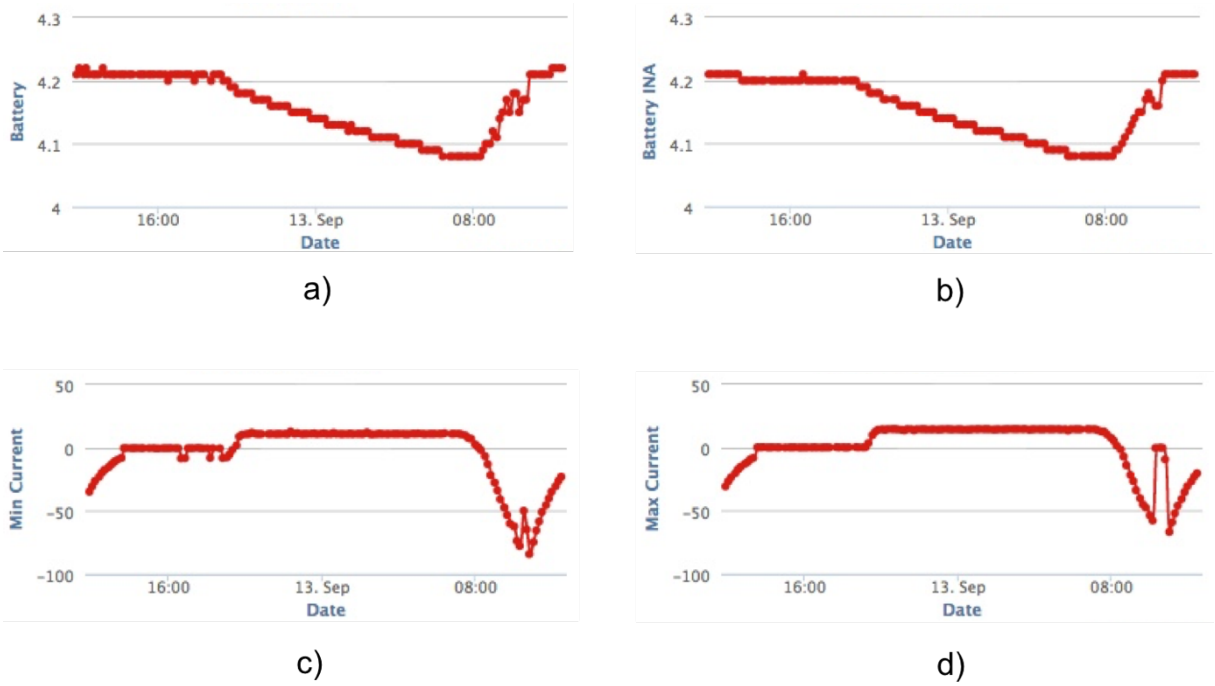

Figure 16 – Power consumption testing results

Performing a joint analysis of the graphs shows that the battery voltage maintains constant at 4.2V and the current decrease. This is one of the typical phases of a charge cycle for a Lithium-Ion battery as is visible in Figure 17. More information about the charger cycle for Lithium-Ion batteries is available at this link[24].

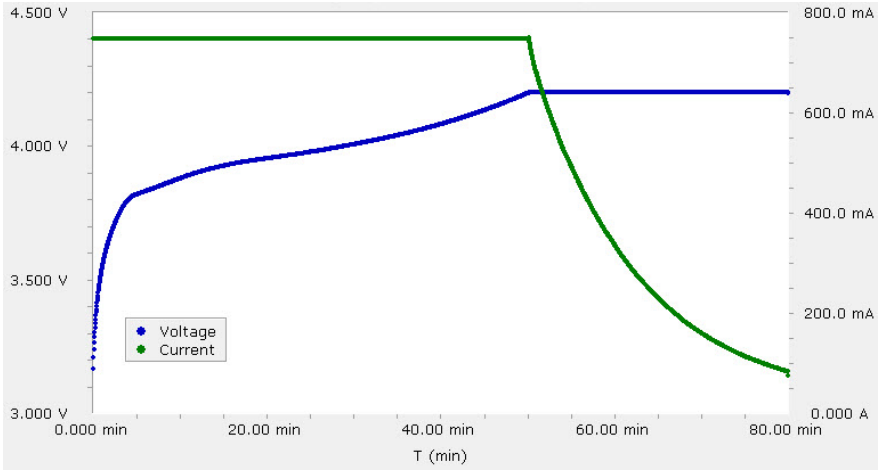

Figure 17 – Charge Cycle for a Lithium-Ion battery[25]

In the phase where the voltage remains around 4.2V and the maximum and minimum current has values around 0mA can mean the occurrence of one of two situations: The battery is full

[24] http://batteryuniversity.com/learn/article/charging_lithium_ion_batteries

[25] http://www.gamry.com/assets/Uploads/Charge-Cycle-Li-Ion-Battery.jpg

and therefore does not enter or leave current to the solution or the solar panel is generated the necessary energy and directly feed the solution.

In the situation where the solar panel is no longer generating energy due to the absence of the sun, the case at night, the battery voltage will decrease and it is visible, by the maximum and minimum current values, that the solution has a power consumption of 15mA.

This analysis allows to conclude that the location where the LoRa Weather Station was deployed allows that this solution was energetically autonomous. The battery voltage just decrease from 4.2v to about 4.09v. With the incidence of the sun, beginning of the day, the battery will start to be charged through the solar panel.

ACKNOWLEDGEMENT

This document has been produced in the context of the H2020 WAZIUP project. The WAZIUP project consortium would like to acknowledge that the research leading to these results has received funding from the European Union's H2020 Research and Innovation Programme under the Grant Agreement H2020-ICT-687670.