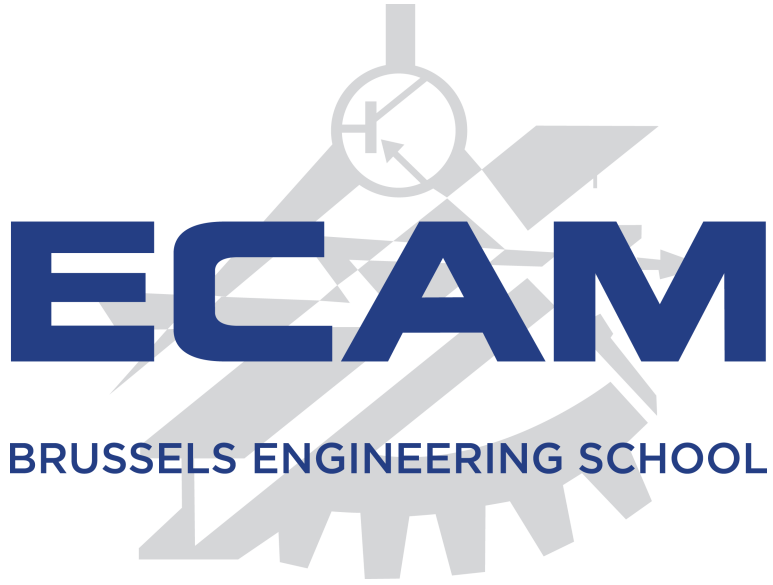
Ecole Centrale des Arts et Metiers



Rapport de Projet IoT

The Windstation

Smits Victor 16107
Snyers Harold 16243
Delvaux Thibault 16 ; ; ;

Professeur : C Marchand & F Dekimpe

1er juin 2020

Table des matières

Projet IoT - The WindStation

1 Introduction au projet

Étant tous trois des amateurs de voiles, l'idée nous est venu sachant qu'avoir les conditions de vents et de la houle en temps réel serait très intéressant. L'idée serait d'avoir des bornes la plupart des plages utilisées pour la voile afin de pouvoir y mesurer la direction du vent, la force du vent et plus encore comme la taille de la houle, la température dans et en dehors de l'eau.

Pour cela, nous utilisons pour le moment dans le projet un anémomètre et une girouette. Ces deux éléments sont à leur tour connecté à un module arduino, le feather 32u4, qui transmettra les données captées via le moyen de communication LoRaWan à partir de The Think Network.

2 Architecture mise en place

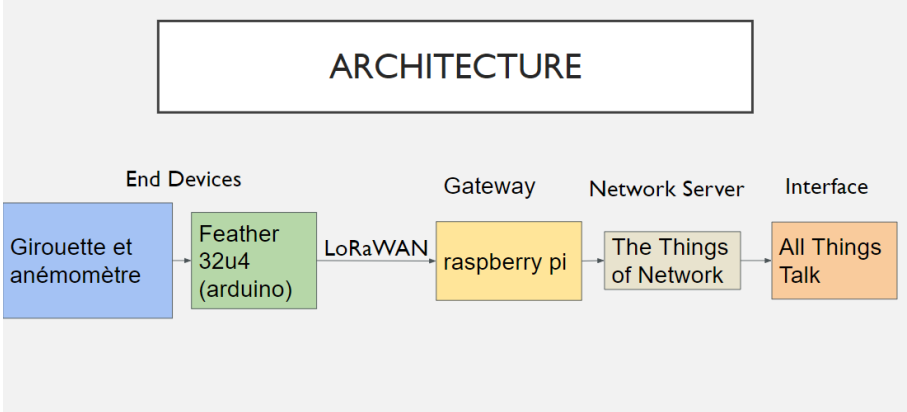


FIGURE 1.2.1 – Architecture du projet

Ici nous n'expliqueront que la partie gateway et LoRaWAN de l'architecture car on explique par après la partie arduino, the things of network et all things talk au point 4

2.1 LoRaWAN

LoRaWAN est un protocole de télécommunication permettant la communication à bas débit, par radio, d'objets à faible consommation électrique communiquant selon la technologie LoRa et connectés à l'Internet via des passerelles, participant ainsi à l'Internet des objets.

informations intéressantes :

— Origine militaire : plus robuste, résiste au brouillage, plus sécurisé
— Authentification des objets par un id unique IEEE EUI-64
— Chiffrement AES 128 bits de bout en bout
— Capteurs ultra faible consommation : >5ans
— Coûts du module radio produit par semtech : <2€

La portée d'une communication varie en fonction :

— de la bande passante
— la puissance de sortie du signal
— le facteur d'étalement utilisé
— la distance par rapport à la gateway

2.2 Le gateway

Le gateway permet la communication entre le réseau internet et le réseau LoRaWAN.

Le gateway :

— Est dédié à la centralisation des communications de plusieurs dizaines d'objets

— Fait office de lien entre les objets et le réseau Ethernet

— Est alimenté via secteur et situé à un endroit stratégique (ici l'endroit n'était pas très stratégique car à l'intérieur dans les locaux de l'ECAM, un toit par exemple aurait été plus intelligent

— a une architecture plus puissante

3 Hardware

3.1 Liste des composants

— Girouette *LEXCA002*

— Anémomètre *LEXCA003*

— Interface RJ11 vers cable arduino

— Adafruit Feather 32u4

3.2 Montage

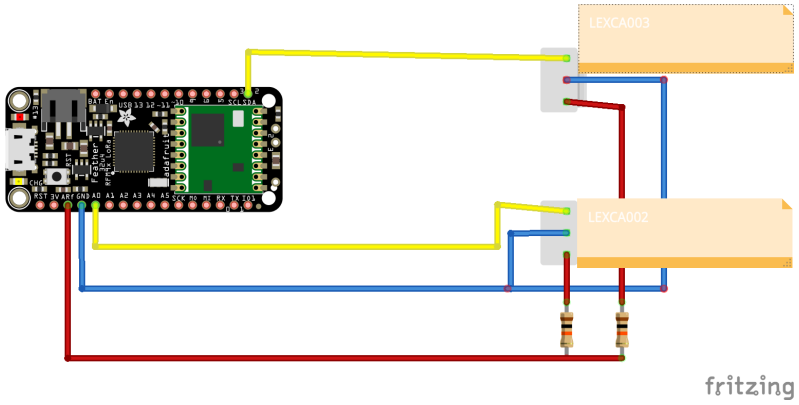


FIGURE 1.3.2 – montage du projet.

4 Software

4.1 Installation

Afin de pouvoir envoyer les données sur notre interface, il nous a fallu développé un script capable de reprendre les données mesurée avec les deux capteurs, la girouette et l'anémomètre pour ensuite les envoyé par LoRaWan. Puisque nous avions comme module arduino la Feather 32u4, nous avons opté d'utiliser la librairie "TinyLora.h".

Avant de procéder à la communication LoRaWan et le reste du script, il faut installer la librairie 'TinyLora.h". Avec l'IDE arduino ceci peut se faire en allant dans l'onglet "sketch", ensuite "Include libraries" et finalement "manage Librairies".

Une dernière chose avant de pouvoir continuer avec le reste reste à faire, c'est à dire vérifier si TinyLora est bien configurer pour la bonne région. Pour ce faire, dans la dossier "librairie/TinyLora", on ouvre le fichier le fichier "tinyLora.h".

tinyLora.h

```
1 /** Region configuration*/
2 #if !defined(EU863) && !defined(AU915) && !defined(AS920)
3   // #define US902 ///< Used in USA, Canada and South America
4   #define EU863 ///< Used in Europe
5 #endif
6 //#define AU915 ///< Used in Australia
7 //#define AS920 ///< Used in Asia
8
9 #define RFM9x_VER    0x12 ///<Expected RFM9x RegVersion
```

Comme on peut le voir dans le code ci-dessus, la région pour l'Europe est décommenté.

Ensuite il faut aussi installer une board compatible avec la Feather 32u4. Pour ceci on va dans l'onglet "Tools", puis "boards" et finalement "Manage boards". Dans ce dernier on recherche "Arduino AVR Boards".

4.2 Scripts

Variable globale ligne 7-38

Les premières variables qu'on retrouve dans le code sont nécessaire pour la communication LoRaWan avec le gateWay. Ensuite nous avons une variable qui contiendra toutes les valeurs à envoyé au GateWay.

Ensuite une variable qui détermine l'interval de temps qu'on veut entre chaque envoie des données au Gateway. La prochaine variable est nécessaire pour la configuration LoRaWan avec le module Feather 32u4.

Et pour finir nous avons des variables globales pour les calcules de vitesses du vent et de la direction du vent.

Void Setup : ligne 40-68

Dans cette partie nous avons l'initialisation de LoRa et le setup de la pin 3 pour l'anémomètre avec un attachInterrupt afin pouvoir compter le nombre de tour. Ce nombre de tour est compté grâce à une petite fonction, "countWind() (ligne 120-122) qui incrémente une variable (représente le nombre de tour dans un laps de temps)

Void loop() : ligne 71-118

Premièrement nous lisons la valeur mesurer sur le pin A1, valeur renvoyée par la girouette (représente une direction). Ensuite nous passons par une petite boucle while contenant 16 valeurs dans l'ordre variant entre 0 et 1023 (nottament la variable "ValeurNumerisee". À la fin de la boucle un certain index est enregistré dans la variable "k" (représente l'index d'une liste qui sera vu plus tard)

Ensuite nous avons l'appel d'une méthode, à savoir "measure()" (ligne 124-133). Dans cette fonction on remet le compteur de tour à zéro et on attend à certain moment pour après calculer la vitesse du vent à partir de nombre de tour mesuré.

Par après, on ajoute les variables contenant la vitesse du vent en km/h et l'index k dans le loraData (data packet). Ce dernier est ensuite ajouté dans une fonction de la librarie Lora chargé d'envoyer le packet.

4.3 The Things Network

Pour réaliser notre communication LoraWan nous avons fait appelle au fournisseur The Things Network (TTN) via la gateway mis a disposition par l'ECAM. Pour pouvoir utiliser la gateway nous avons du enregistrer notre node IoT auprès d'elle.

Pour enregistrer notre device nous avons utilisé la console en ligne de TTN. Via les données que nous avons obtenus sur la console, nous avons pus réaliser la communication entre notre device et la gateway.

4.4 All Things Talk

Afin de pouvoir fournir un interface utilisateur, nous avons décidé de travailler avec la platforme en ligne All Things Talk qui permet de réaliser des affichage interactif des données que nous envoyons sur le réseau de TTN. L'avantage d'utiliser All Things

Talk par rapport à d'autres systèmes, est l'intégration déjà existante entre TTN et l'interface.

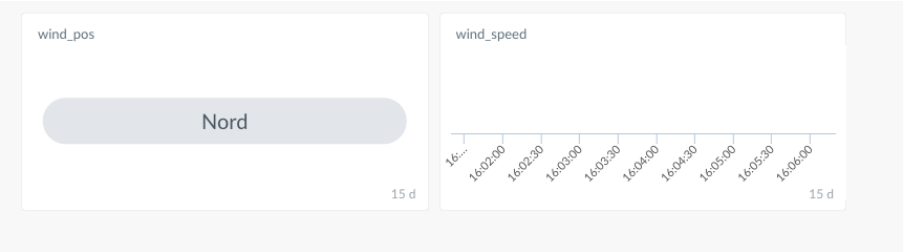


FIGURE 1.4.3 – Interface utilisateur All Things Talk

Pour pouvoir utiliser All Things Talk, nous avons dû programmer un format pour notre payload qui sera lus et interprété par après dans notre interface All Things Talk.

Nous envoyons deux données sur le réseau TTN, une donnée de vitesse et une donnée de position. La donnée de vitesse sera simplement transformer de hexadécimal à décimal. En ce qui concerne la donnée de position, cela c'est avérer plus compliquée que prévus. En effet nous avons dû interpréter le nombre obtenu en sortie du capteur en une orientation, nous avons décidé de réaliser cette interprétation au niveau du payload format. Pour ce faire nous avons réalisé une série de tests et ainsi décidé une marge de valeur correspondant à une orientation.

Après avoir reçu la donnée de position, nous transformons cette position en index qui nous donnera l'orientation de la girouette. Après transformation des données reçues nous obtenons un JSON.

Payload Format

```
function Decoder(bytes, port) {
  // Decode an uplink message from a buffer
  // (array) of bytes to an object of fields.
  var speed = {};
  var pos = {};
  var out = {};

  var Orientation = ["Est-Sud-Est","Est-Nord-Est","Est","Sud-Sud-Est","
   Sud-Est","Sus-Sud-Ouest","Sud","Nord-Nord-Est","Nord-Est","Ouest-
   Sud-Ouest","Sud-Ouest","Nord-Nord-Ouest","Nord","Ouest-Nord-Ouest",
   "Nord-Ouest","Ouest"];

  if (port === 1){
    var k = (bytes[2]*256+bytes[3]);
    if(k == 18){
      k =15;
    }
```

```
    speed.value = (bytes[0]*256+bytes[1])/100;
    pos.value = Orientation[k];
    out.wind_speed = speed;
    out.wind_pos = pos;
  }

  return out;
}
```

Format JSON

```
{
  wind_speed : {
    value : 30
  },
  wind_pos : {
    value : "Nord"
  }
}
```

5 Sécurité mise en place

Pour la sécurité, nous n'en avons pas pour notre applciation car aucune donnée sensible n'est envoyé.

Cependant il serait intéressant de trouver une manière de mieux protéger les données sachant que dans futurs projets il serait envisageable de rajouter une caméra vidéo afin de pouvoir observer la mer et donc la houle. Il serait inconscient de penser que personne ne pourrait reprendre ces images et donc engendrée une atteinte à la vie privé des personne étant sur la plage.

6 Portée et consommation

6.1 Portée

La portée de notre communication est "infini" dans le sens ou il suffit d'être proche d'une gateway à laquelle on est connecté pour pouvoir envoyer quelque chose sur le réseau The Thing Network.

Le range des gateway varie en fonction de leur location, plus précisément s'ils sont situé à l'extérieur ou l'intérieur d'un bâtiment. À l'intérieur, le range est évidemment bien réduit à cause des obstacles. Cependant à l'extérieur le range est grand allant jusqu'à théoriquement 10km.

6.2 Consommation

Nous n'avons malheureusement pas pu calculer la consommation réelle mais si on ne prend en compte que l'envoi. Nous avons calculer grâce à la datasheet que l'arduino est capable d'être alimenté pendant 11 ans avec une batterie de 10000mAh avec une sortie USB 5v

0.32	puissance (W)	transmit time sec
0.0096	energie/trans (J)	0.03
1.728	energie/heure (J)	delay time sec
		20
180000	energie (J)	battrie Ah
104166.6667	heure	10
4340.277778	jours	duty cicle
11.89117199	Annés	180

FIGURE 1.6.4 – Calcul de la consommation de l'envoi

Comme dit dans la conclusion la consommation en électricité pourrait dans le futur être assurée par un panneau solaire...

7 Conclusion

7.1 Objectifs

Atteint

- affichage de la vitesse du vent
- affichage de la direction du vent

Non atteint

—

7.2 Amélioration possible ?

Il est possible d'améliorer ce projet de pleins de façon, nous allons vous citer celle qui nous paraissent le plus raisonnable niveau conception ou le plus utile.

Premièrement nous pourrions ajouté un panneau solaire à notre projet afin de pouvoir rajouté facilement des nouvelles fonctionnalité sans devoir trop se préoccuper de la consommation totale du projet. Cependant, il faudra alors penser à un moyen de sécurisé les bornes contre des intempérie ou des vols.

Une seconde amélioration serait d'améliorer la précision de la direction du vent. En effet pour l'instant la mesure est très approximative et il serait donc intéressant de faire plus de test et de mesure pour certifier une direction juste.

Une autre amélioration possible serait d'améliorer la précision du vent. À nouveau celui-ci devrait passer par des tests pour comparer la formule que nous avons développé avec la vitesse du vent réelle.

Ensuite on pourrait ajouté toute sorte de fonctionnalité à ces bornes comme une caméra qui nous permettrait de voir les conditions avec un visuel en direct (vent, houle). On pourrait rajouté aussi un thérmomètre extérieur pour le température en dehors de l'eau et thérmomètre dans la mer pour avoir la température dans l'eau. Finalement on pourrait aussi ajouté un instrument de mesure capable de mesurer la taille de la houle (hauteur des vagues).

7.3 Problème rencontré

Nous avons rencontré plusieurs problèmes à différents niveaux lors de la conception de notre projet.

Les premiers problèmes rencontrées étaient dû à des capteurs sans datasheet et avec très peu d'information sur internet. C'est pourquoi la mise en place d'une mesure de vitesse avec l'anémomètre a pris beaucoup de temps.

Le second problème est la mesure de la direction qui est pas précise du tout dû à l'absence d'une datasheet. C'est donc pour cela que la mesure de la direction reste très approximative.

Un autre problème qu'on a rencontré est survenu lorsque nous avons essayer de communiquer via LoRaWan avec la GateWay. Heureusement après mainte recherches et l'aide de Monsieur Dekimpe, nous avons trouvé l'origine du problème. Ce dernier étant une mauvaise configuration de la région pour la librairie "TinyLora".

Annexe

1 Script hello Lora Arduino complet

```
#include <TinyLoRa.h>
#include <SPI.h>

// Visit your thethingsnetwork.org device console
// to create an account, or if you need your session keys.

// Network Session Key (MSB)
uint8_t NwkSkey[16] = { 0x7E, 0x7B, 0x51, 0x7F, 0xAD, 0x46, 0x99, 0xD9,
                        0x28, 0x03, 0x1B, 0xCE, 0x15, 0x94, 0x41, 0x44 };

// Application Session Key (MSB)
uint8_t AppSkey[16] = { 0x57, 0x34, 0x8D, 0xC8, 0x30, 0x7C, 0xEF, 0xDE,
                        0x41, 0xCC, 0xF5, 0x20, 0x97, 0xD7, 0x8B, 0x10 };

// Device Address (MSB)
uint8_t DevAddr[4] = { 0x26, 0x01, 0x1A, 0x75 };

// Data Packet to Send to TTN
unsigned char loraData[4];


// How many times data transfer should occur, in seconds
const unsigned int sendInterval = 20;

// Pinout for Adafruit Feather 32u4 LoRa
TinyLoRa lora = TinyLoRa(7, 8, 4);

// Pinout for Adafruit Feather M0 LoRa
//TinyLoRa lora = TinyLoRa(3, 8, 4);

/* Sensor Monitor */
const int m_time = 10;       //Meassuretime in Seconds
int wind_ct = 0;
float wind = 0.0;
unsigned long time = 0;

const int ValeursNumerisees[16] = {
    66,84,93,126,184,244,287,406,461,599,630,702,785,827,886,944};

```

```
void setup()
{
  delay(2000);
  Serial.begin(9600);
  while (! Serial);

  // Initialize pin LED_BUILTIN as an output
  pinMode(LED_BUILTIN, OUTPUT);

  // Initialize LoRa
  Serial.print("Starting LoRa...");
  // define multi-channel sending
  lora.setChannel(MULTI);
  // set datarate
  lora.setDatarate(SF7BW125);
  if(!lora.begin())
  {
    Serial.println("Failed");
    Serial.println("Check your radio");
    while(true);
  }

  Serial.println("OK");

  /* Sensor Monitor */
  time = millis();
  pinMode(3,INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(3), countWind, RISING);
}


void loop()
{
  int16_t sensorValue = analogRead(A1);
  Serial.print("sensor ");
  Serial.println(sensorValue);

  // initialisation de l'indice de boucle 'tant que'
  int16_t k = 0;
  // comparaison de la mesure avec la moyenne de 2 resistances
  // successives
  while(sensorValue > int(
  (ValeursNumerisees[k]+ValeursNumerisees[k+1])/2)){
    k = k+1;
  }

  measure();

  Serial.print("WindSpeed: ");
  Serial.print(wind);        //Speed in Km/h
  Serial.print(" km/h - ");
  Serial.print(wind / 3.6); //Speed in m/s
  Serial.println(" m/s");

```

```
  int16_t wind_km = round(wind * 100);
  int16_t wind_ms = round(wind/ 3.6 * 100);

  // encode int as bytes
  //byte payload[2];
  loraData[0] = highByte(wind_km);
  loraData[1] = lowByte(wind_km);

  loraData[2] = highByte(k);
  loraData[3] = lowByte(k);

  Serial.println("Sending LoRa Data...");
  lora.sendData(loraData, sizeof(loraData), lora.frameCounter);

  Serial.print("Frame Counter: ");Serial.println(lora.frameCounter);
  lora.frameCounter++;

  // blink LED to indicate packet sent
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);

  Serial.println("delaying...");
  delay(sendInterval * 1000);
}

void countWind() {
  wind_ct ++;
}

void measure() {
  wind_ct = 0;
  time = millis();
  delay(1000 * m_time);
  //detachInterrupt(9);
  // tour/s * 2pi * r * 3.6
  wind = ((float)wind_ct / (float)m_time * 2 * 3.14 * 0.072)*3.6 ;
  //wind = 2*3.1415*0.031*5*wind_ct;
  Serial.print("windCt" + wind_ct);
}
```