Mise en place d'un capteur de température de l'eau d'une serre,

Lola Dussosoy, Benoit Ribeaud, Sophie Bodart

Univ. Grenoble Alpes, Grenoble INP, ENSE3, F-38000 Grenoble, France

RÉSUMÉ. La maîtrise de l'énergie dans les bâtiments est une clé de la réussite de la transition énergétique. Pour cela la domotique est très utile, car elle permet de monitorer les consommations d'un bâtiment puis de le piloter intelligemment à distance. Le système mis en œuvre dans cette étude, utilisant un LoPy, le module LoRa, le serveur The Things Network (TTN) et une Raspberry Pi permet de créer un système de domotique peu cher, puissant et facile à prendre en main. Il permet de récupérer des données de température d'un capteur grâce à la LoPy, de transmettre les données par wifi à un serveur TTN grâce au module LoRa, puis de stocker les données sur InfluxDB, et enfin de les visualiser et traiter sur Grafana.

MOTS-CLÉS : capteur, LoPy, LoRa, serveur TTN, base de données, visualisation

ABSTRACT. Energy management in buildings is a key to a successful energy transition. To do this, home automation is very useful, because it allows monitoring the consumption of a building and then controlling it intelligently from a distance. The system implemented in this study, using a LoPy, the LoRa module, the TTN server and a Raspberry Pi, makes it possible to create a home automation system that is affordable, powerful and easy to use. It allows to retrieve temperature data from a sensor thanks to the LoPy, to transmit the data by wifi to a TTN server thanks to the LoRa module, then to store the data on Influx DB, and finally to visualize and process them on Grafana.

KEYWORDS : sensor, LoPy, LoRa, TTN server, database,visualization

1. La domotique pour maîtriser ses consommations énergétiques

1.1. Importance de la maîtrise de l'Énergie dans les bâtiments

Alors que la consommation globale d'électricité ne fait qu'augmenter, la maîtrise de l'énergie dans les bâtiments apparaît comme l'une des mesures les plus efficaces pour lutter contre le changement climatique puisque la moitié de l'énergie mondiale est consommée dans les bâtiments, et que le secteur des bâtiment est le plus émetteurs de CO2 tout en ayant les réductions d'émissions les moins chères à mettre en œuvre (voir figure 1).

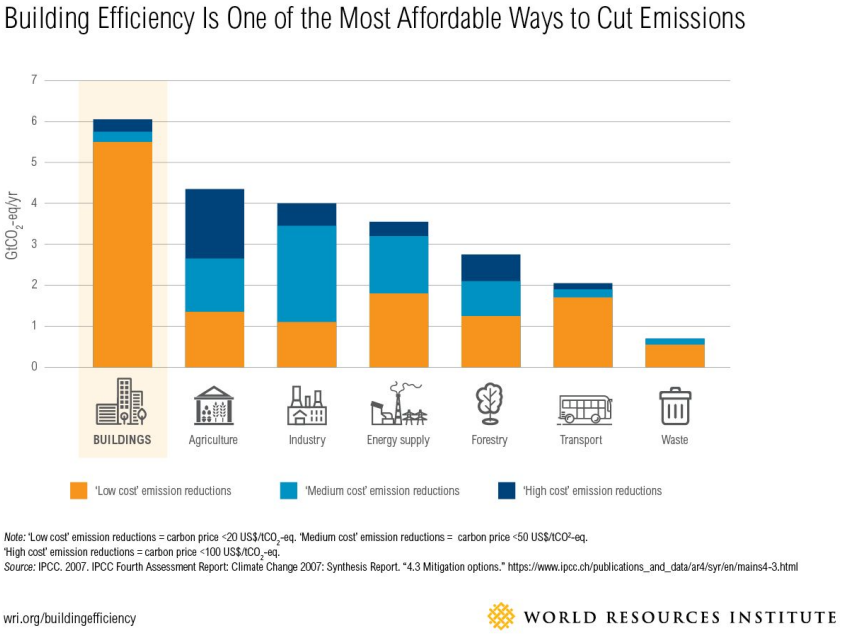


Figure 1 : Répartition des émissions de CO2 par secteur

Une des solutions les moins chères et les moins invasives pour réduire la consommation énergétique des bâtiments est d'utiliser la domotique et de rendre son bâtiment intelligent.

La domotique consiste à utiliser l'informatique et les télécommunications pour optimiser les consommations de la maison et améliorer le confort des usagers.

La première étape est d'analyser la consommation du bâtiment en disposant des capteurs à différents endroits de la maison. On pourra par exemple installer des capteurs de présence, de luminosité, de consommation électrique au niveau des prises, de température et humidité, de CO2. Le processus de création des capteurs, récupération des données et analyse sera détaillé plus bas.

L'analyse des données des capteurs permettra ensuite d'identifier les postes de réductions des consommations, puis programmer des scénarios de commande du chauffage, des volets, de l'éclairage, etc. afin d'optimiser la consommation énergétique de la maison en fonction de la présence ou non d'usagers et des conditions météorologiques.

Automatiser la maison, prédire la consommation, comprendre sa consommation, identifier les postes de réductions, chauffage intelligent en fonction de l'occupation du bâtiment, idem éclairage, contrôle des appareils ménagers pour consommer aux creux de consommation afin de soulager le réseau. Fermer les volets pour garder la chaleur à intérieure,

Dans notre cas d'étude, nous nous éloignons un peu de la problématique de l'énergie dans les bâtiments pour chercher à installer un capteur pour récupérer la température de l'eau d'un bassin, aquarium ou serre d'aquaponie. Mais une fois le processus acquis, la méthode est la même pour installer des capteurs utiles dans la maison.

1.2. Un système domotique très peu coûteux grâce au LoPy et LoRa

Afin d'étendre l'utilisation des capteurs à des utilisations domestiques, il est important de développer des capteurs à bas coûts, de petites tailles et grandes puissances, avec une batterie longue durée. C'est ce qu'on peut faire avec une LoPy. Il faut ensuite rendre la récupération, le stockage et le traitement des données accessible à tous grâce à une prise en main très facile et des logiciels open sources.

Grâce à une LoPy, on peut réaliser chez soi facilement et à bas coût un système domotique qui pourra piloter une maison, adapté aux besoins précis de chacun, puis agréger les données des capteurs.

La technologie LoRa nous permet ensuite de télécommuniquer les données.

2. Présentation de l'étude

L'objectif de notre projet est d'installer un capteur autonome de température d'eau dans un bassin, de récupérer les données sur un serveur, puis de pouvoir les stocker et les visualiser.

2.1. Le matériel

2.1.1 Le Capteur

On utilise un capteur de température submersible DS18B20. Ce capteur possède une plage de température allant de -55°C à 125°C avec une tolérance comprise entre -0.5°C/+0.5° si celle-ci se situe entre-10°C et 80°C. Cela est largement suffisant pour acquérir la température de l'eau d'un bassin extérieur.

L'acquisition des données se fait grâce à la bibliothèque OneWire. Nous avons trouvé le code pour récupérer les données sur internet (Voir bibliographie source 2).

C'est un capteur numérique ce qui permet de le brancher sur n'importe quelle borne de notre microcontrôleur, et il est moins sujet aux parasites.

C'est un capteur OneWire ce qui signifie qu'on peut brancher plusieurs capteurs sur la même broche du microcontrôleur. Cet avantage ne sera pas utile dans le cadre de notre projet, mais il peut être très intéressant dans un projet de domotique où l'on souhaite installer de nombreux capteurs.

Figure 2 : le capteur DS18X20 installé dans le bassin

2.1.2 Le micro contrôleur

Nous utilisons un Pycom LoPy4 comme nano ordinateur qui va lire les données du capteur de température. Comme énoncé précédemment, il a l'avantage d'être peu cher, de petite taille et suffisamment puissant pour le travail qu'on lui demande.

Nous ajoutons une plaque d'extension Pycom makr pour la programmation.

L'avantage de ce microcontrôleur est qu'il consomme très peu d'énergie (3 mA, 3,3 à 5,5 Vcc) .

La LoPy est dotée d'un module LoRa qui permet de télécommuniquer les données à longue portée et bas débit pour des objets à faible consommation électrique. Ce module permet de transmettre les données au serveur par l'intermédiaire d'une gateaway.

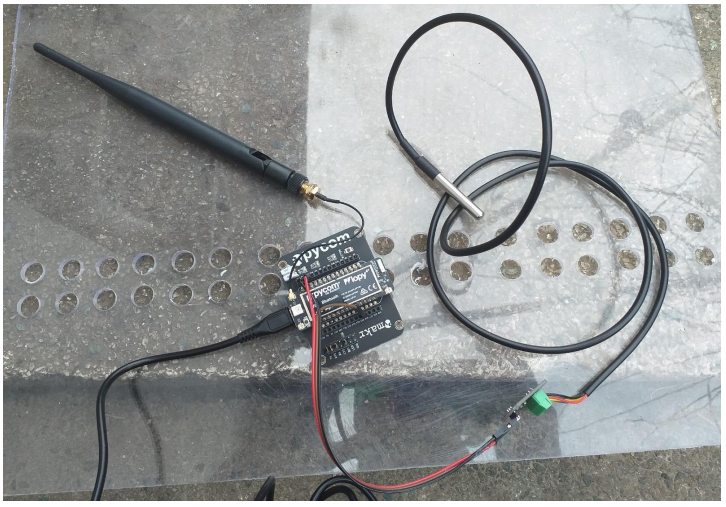


Figure 3 : Photo du système complet : le capteur, la LoPy et son antenne pour envoyer les données sur le serveur TTN

2.1.3 Le serveur The Things Network (TTN)

The Things Network est le serveur que nous avons choisi d'utiliser lors de notre projet, c'est un réseau communautaire permettant à des objets de transmettre des petits volumes d'informations en consommant très peu d'énergie. Il utilise la technologie LoRa en adéquation avec les standards LoRaWAN. L'un des avantages de cette technologie est le déploiement de passerelles ou gateway[1].

En effet, chaque passerelle est utilisable par tous les utilisateurs, ainsi chaque objet peut continuer à émettre des données vers nos applications. Inversement les passerelles que nous installons chez nous sont susceptibles de recevoir des données d'objets qui ne nous appartiennent pas. The Things Network développe une passerelle à petit prix pour aider au déploiement communautaire du réseau mais il est aussi possible d'utiliser des passerelles faites soi-même ou bien d'autres fabricants.

Les données envoyées par la LoPy sont transmises au serveur TTN en passant par la gateway.

La gateway est une porte d'entrée accessible à tous sur son rayon d'action (quelques km), car le système LoRa est open source. Elle est reliée à un câble éthernet qui envoie les données sur le serveur TTN.

Figure 4 : Photo d'une passerelle développée par The Things Network

---

[1] Passerelle : porte d'entrée accessible à tous sur son rayon d'action. Technologie Open Source.

2.1.4 Rasberry Pi

Tout d'abord il convient d'expliquer ce qu'est un Raspberry Pi, c'est un nano ordinateur de très petite taille (en moyenne de la taille d'une carte de crédit) avec les connectiques nécessaires pour le brancher à un écran et pouvoir l'utiliser comme un ordinateur standard. Il est néanmoins beaucoup moins cher qu'un ordinateur standard et beaucoup plus compact ce qui en fait son intérêt. Il peut être utilisé pour différentes applications et notamment la création de serveur Web chez soi. Du fait de sa taille il ne faut pas s'attendre à des performances exceptionnelles néanmoins il est suffisant pour notre projet de mise en place d'un capteur d'eau dans un bassin.

En effet nous allons nous servir de la Raspberry Pi pour récupérer les données présentes sur le serveur TTN et les renvoyer vers notre base de données InfluxDB et notre plateforme pour les visualisées Grafana.

2.1.5 InfluxDB

InfluxDB est un time series database (TDSB). Elle est utilisée pour stocker des larges volumes de données chronologiques ou des séries temporelles venant de sources différentes. Elle a été créée dans le but de collecter le volume croissant de données issues de l'internet des objets notamment dans le domaine de la domotique.

Dans notre projet, elle servira à stocker les données collectées par notre capteur. Nous allons aussi connecter cette base de données avec l'outil de visualisation qu'est le logiciel Grafana pour avoir un monitoring et une visualisation de nos données dans le temps.

2.1.6 Grafana

Grafana est une plateforme open source servant à la surveillance, l'analyse et à la visualisation de différentes données. Elle permet de générer des graphiques et des tableaux de bord à partir de bases de données de séries temporelles (TDSB) telles que Graphite, InfluxDB ou OpenTSDB.

Elle va nous permettre d'afficher sous forme de graphique les données renvoyées par notre capteur.

2.1.7 Architecture des différents matériels pour notre projet

Voici ci-dessous l'architectures des différents matériels composant notre système :

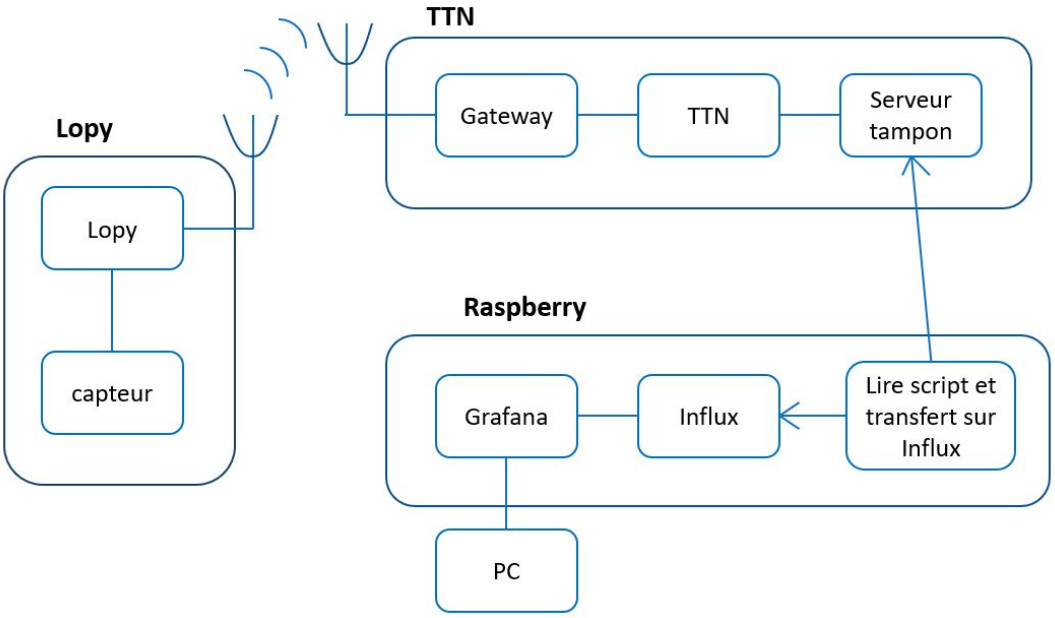


Figure 5 : Architecture des différents matériels composant notre projet

Concernant la partie LoPy, notre LoPy va récupérer les données du capteur dans notre bassin et les transmettre à notre serveur TTN.

Concernant la partie TTN, les données transmises par notre LoPy vont être récupérées par le serveur à l'aide de clés API dont nous détaillerons le fonctionnement dans les parties suivantes.

Concernant la partie Raspberry Pi, notre nano ordinateur va venir récupérer les données présentes sur TTN et les envoyer sur InfluxDB pour pouvoir les stocker.

3. ACQUISITION DES DONNÉES GRÂCE À LA LOPY

La partie concernant l'acquisition des données grâce à la LoPy a été très chronophage pour nous. En effet, il nous a fallu chercher sur internet comment notre capteur émettait la donnée et sous quelle forme pour ensuite pouvoir écrire un code qui allait pouvoir permettre à notre LoPy de récupérer l'information et ensuite la transmettre. Tout d'abord, nous avons essayé de nous inspirer des programmes que nous avions utilisés auparavant pour la mise en place du capteur DHT22 mais les données n'étaient pas codées sous le même format.

Finalement après avoir passé en revue quelques forums, nous avons trouvé le code de Dani Campora publié sur github en libre accès avec la librairie permettant de faire de récupérer les données de notre capteur (voir bibliographie source 2). Une fois implémenté dans notre éditeur de texte Atom,

nous avons pu avoir accès aux données récupérées par notre capteur et nous avons pu les transmettre à notre plateforme TTN.

4. TRANSFERT DES DONNÉES SUR LE SERVEUR TTN

Une fois les données récupérées avec la LoPy, nous voulons envoyer celle-ci vers le serveur TTN afin de les visualiser en ligne et les stocker sur une base de données.

La première étape est de connecter notre LoPy au serveur TTN après avoir créé une nouvelle application sur le serveur. Pour cela, il suffit de créer un nouvel appareil lié à l'application et de générer automatiquement trois identifiant qui permettent de connecter la LoPy à notre serveur :

- Device EUI (Extended Unique Identifier) : globally unique device ID by the network server
- Application EUI : globally unique application ID
- App Key : Is used to calculate the cryptographic signature for join-request message

Nous rentrons ensuite ces informations sur Atom afin de relier notre LoPy à notre serveur :

```
# create an OTA authentication params
dev_eui = binascii.unhexlify('00B8145FE2A87341')
app_eui = binascii.unhexlify('70B3D57ED003AF15')
app_key = binascii.unhexlify('F7D02A6E7637AF330DA6D953DFC3F55F')
```

Nous récupérons ensuite les données du capteur DS18B20 pour les envoyer sur le serveur TTN. Nous appliquons un « time.sleep » de 900 secondes, ce qui signifie que notre LoPy récupèrera les données du capteur toutes les 900 secondes, c'est-à-dire toutes les 15 min. En effet, il n'est pas pertinent d'avoir un temps plus court entre deux acquisitions car la température du bassin ne varie pas rapidement.

```
#DS18B20 data line connected to pin P10
ow = OneWire(Pin('G16'))
temp = DS18X20(ow)

while True:
    temp.start_conversion()
    time.sleep(1)
    print(temp.read_temp_async())
    time.sleep(1)
    s.send(str(temp.read_temp_async()).encode())
    time.sleep(900)
```

Il est important de noter que les données ne sont pas envoyées vers le serveur TTN sous la forme de flottant, mais elles sont en réalité transformées en chaine de caractère Unicode puis transformées une seconde fois en une représentation bytes de la chaîne Unicode, codée dans l'encodage demandé grâce à la fonction « .encode ». Une fois transmis à notre serveur, nous allons donc devoir décoder ces données afin de les lire dans un format compréhensible.

Pour cela, la fonction inverse de la fonction decode est la fonction encode. Nous rentrons donc le code suivant dans la partie « Payload Format » de notre application :

```
function Decoder(bytes, port) {
  var decoded = {};
  decoded.temperature_eau = "";
  for(i=0; i<bytes.length; i++){
    decoded.temperature_eau = decoded.temperature_eau + String.fromCharCode(bytes[i])
  }
  return decoded;
}
```

Nous visualisation ainsi la température en direct sur le serveur :

Afin de conserver ces données qui disparaissent une fois le serveur fermé, nous allons les stocker sur une base de données. Pour cela, nous allons dans la partie « Integrations » de notre application TTN et créons cette base de données. Cette base de données ne permet cependant de stocker les données que pendant 7 jours. Nous allons maintenant voir comment les stocker sur un serveur permanent grâce à l'outil InfluxDB et les visualiser sur Grafana.

5. Traitement et visualisation des données sur Grafana

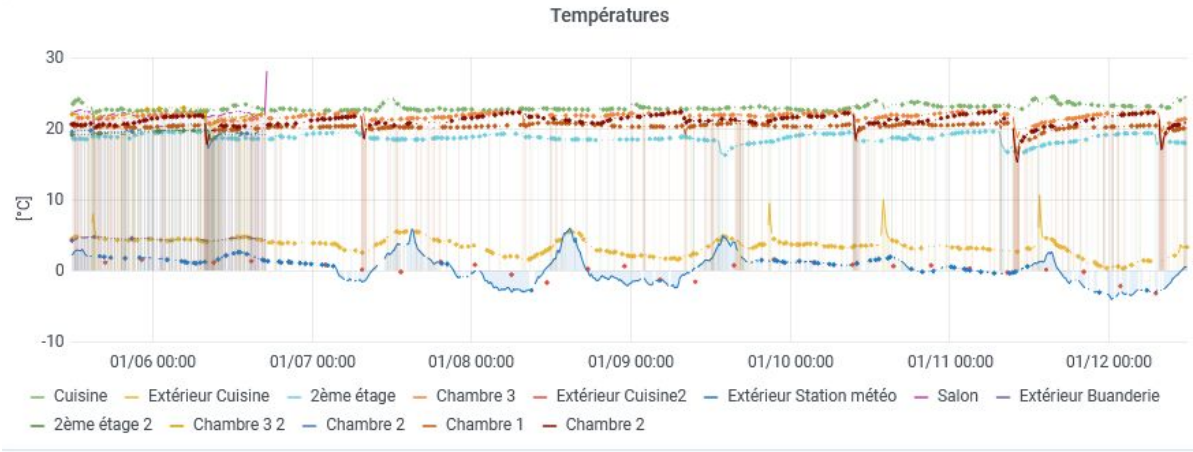
InfluxDB est un outil de gestion de base de données qui va nous permettre de stocker les données de manière permanente. Pour cette partie, nous aurons besoin d'un Raspberry Pi. Nous installerons InfluxDB sur le Raspberry Pi en s'inspirant des travaux de Simon Hearne (voir bibliographie source 3).

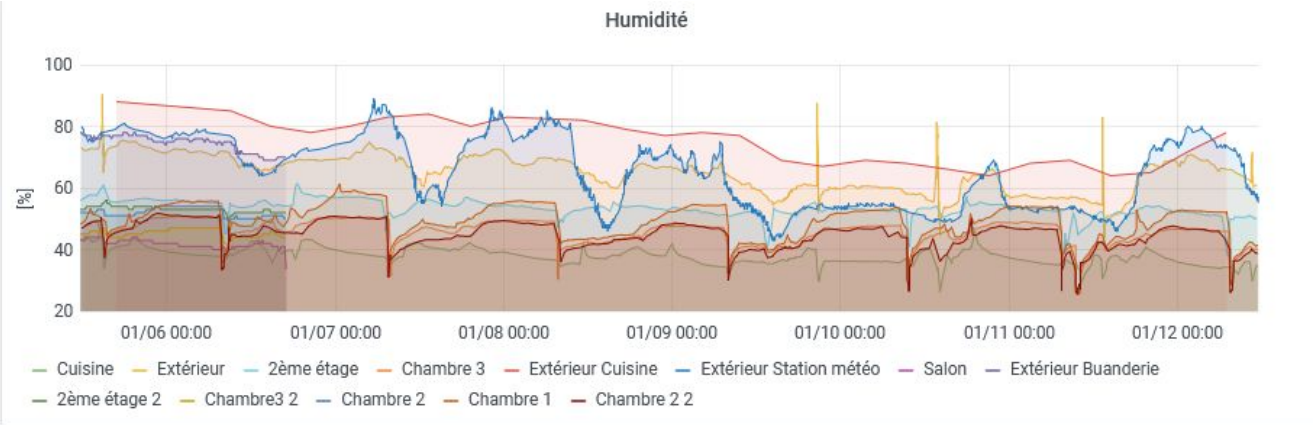
Une fois InfluxDB installé, nous allons visualiser les données sur Grafana. Grafana est un logiciel libre qui permet de visualiser des données depuis des bases temporelles comme InfluxDB. Une fois Grafana installé sur le Raspberry Pi, nous pouvons nous connecter sur le serveur et ajouter tout type de graphique afin de visualiser nos données. Vous trouverez ci-dessous un exemple de tableau de bord traitant des données issus de plusieurs capteurs différents :

6. RÉSULTATS DE L'ÉTUDE

Nous n'avons pas pu récupérer les données de températures du bassin sur Grafana.

Cependant on peut voir sur les graphiques ci-dessous les données de température et d'humidité telles qu'elles s'affichent sur Grafana. L'analyse de ces données constitue la première étape pour apprendre à maîtriser ses consommations.





7. Conclusion

Dans cet article nous montrons l'intérêt des microcontrôleurs de type LoPy et Raspberry Pi pour faire de la domotique peu chère, adaptée aux besoins de chacun et aisée à prendre en main et à terme réduire ses consommations énergétiques dans son logement. Ces systèmes trouvent aussi des applications dans l'industrie, en remplaçant des systèmes industriels excessivement chers.

Le serveur The Things Network est aussi très intéressant car c'est un réseau communautaire permettant à des objets de transmettre des petits volumes d'informations en consommant très peu d'énergie.

De manière générale, nous nous sommes rendus compte qu'il est assez facile de créer des systèmes utilisant l'internet des objets de nos jours. Il existe beaucoup de plateformes communautaires sur internet où l'on peut trouver différents types d'aides tel que github pour la programmation,des forums ainsi que des sites web tel que Miniprojets.net. Nous avons trouvé ce projet très enrichissant et instructif avec de très fortes ouvertures sur le monde de demain auquel nous allons participer en tant qu'ingénieur(e).

8. Bibliographie

(Miniprojet 2019) Sylvain, Monitoring d'une serre, du capteur jusqu'au serveur via le trio LoRa, InfluxDB, Grafana
http://miniprojets.net/index.php/2019/07/23/monitoring-dune-serre-du-capteur-jusquau-serveur/

(Code DS18X20) https://github.com/pycom/pycom-libraries/tree/master/examples/DS18X20

Simon Hearne. « Installing InfluxDB & Grafana on Raspberry Pi ». Consulté le 18 janvier 2021. https://simonhearne.com/2020/pi-influx-grafana/.