2247.1 Internet des Objets– Projet– ISC2il-b

|  |
| --- |
| **Projet IoT** |

|  |
| --- |
| Étudiants participant à ce travail :  **Lucas GOSTELLI, ISC2il-b Nicolas AUBERT, ISC2il-b**  Présenté à :  **Nabil OUERHANI**  Restitution du rapport : **09.06.2022**  Période : **2022**  École : **HE-Arc, Neuchâtel** |

**Groupe 9 – Détection d’incendie à l’aide des capteurs de température, de pression et d’humidité, ainsi que la caméra**

Table des matières

[1 - Introduction 1](#_Toc105057440)

[2 - Spécifications 2](#_Toc105057441)

[2.1 - Cahier des charges 2](#_Toc105057442)

[2.1.1 - Contexte 2](#_Toc105057443)

[2.1.2 - Objectifs 2](#_Toc105057444)

[2.1.3 - Éléments de solution 2](#_Toc105057445)

[3 - Conception 3](#_Toc105057446)

[3.1 - Architecture du projet 3](#_Toc105057447)

[3.2 - Diagramme de classe 4](#_Toc105057448)

[4 - Réalisation 5](#_Toc105057449)

[4.1 - Client MQTT 5](#_Toc105057450)

[4.1.1 - Récupération des mesures 5](#_Toc105057451)

[4.1.2 - Analyse des mesures 5](#_Toc105057452)

[4.1.3 - Envoi au brocker 5](#_Toc105057453)

[4.2 - Stockage des mesures sur InfluxDB 5](#_Toc105057454)

[4.3 - Dashboard sur Grafana 5](#_Toc105057455)

[5 - Tests / Résultats 6](#_Toc105057456)

[6 - Conclusion 7](#_Toc105057457)

[7 - Bibliographies et références 8](#_Toc105057458)

[8 - Tables des illustrations 9](#_Toc105057459)

1. Introduction

Dans le cadre du cours de *2247.1 Internet des Objets*, il nous a été demandé de réaliser un projet pouvant se greffer au *MILL*, apportant alors une plus-value.

Nous avons à disposition un *Raspberry PI 4* ainsi qu’un *Sense Hat* permettant d’acquérir des données sur l’environnement dans lequel il se trouve (Image, température, humidité, pression, …).

Étoffer

1. Spécifications
   1. Cahier des charges
      1. Contexte

* Lorsque l'air s'assèche et que les températures sont enlevées, les risques d'incendie augmentent (copeaux, bois de la structure, ...).
* La combinaison entre les capteurs de température et d'humidité permet de déterminer les risques d'incendie.
  + 1. Objectifs
* Avoir la capacité de lire les données des capteurs d'humidité et de températures disposés à l'intérieur de la machine. De combiner l'information pour en déduire un indice de risque d'incendie.
* Être capable de capturer des images à l'intérieur de la machine et de les traités pour détecter de la fumée ou flamme ce qui indique un incendie.
* Être dans la capacité d'alerter, dans un Dashboard à l'aide de voyant de couleur, le superviseur si la machine est dans des conditions propices à un incendie.
  + 1. Éléments de solution
* PI-Cam, Capteur d'humidité / température
* MQTT
* InfluxDB
* Dashboard avec couleurs (vert = risque faible, orange = risque modéré, rouge = risque élevé, noir = incendie)
* Notifications en cas de risques élevés ou d'incendie (alarme, mail, voyant de couleur)

1. Conception
   1. Architecture du projet

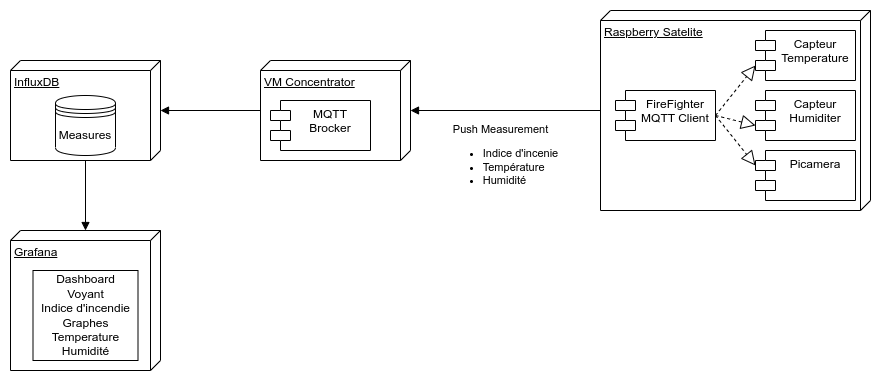


Figure : Diagramme d'architecture du projet

Notre structure ressemble passablement à celle étudiée en cours. Elle se compose d’un client, le *Raspberry* *Pi*, qui récupère les mesures, les traite et les envoie au *Brocker* *MQTT* sur un Topic spécifique.

Ensuite, une instance *d’InfluxDB*, qui est abonnée au Topic correspondant, stocke les mesures.

Finalement, Un Dashboard réalisé avec *Grafana* récupère les données sur l’instance *InfluxDB* et les affiche.

* 1. Diagramme de classe

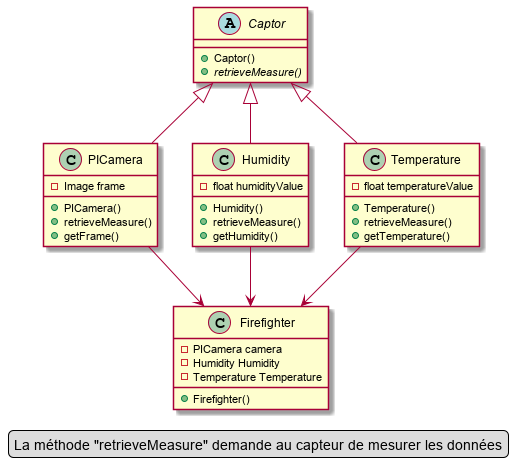


Figure : Diagramme de classe du projet

La classe abstraite *Captor* possède la méthode abstraite *RetrieveMeasure*. Chaque capteur possède sa propre classe, héritant de la classe *Captor*. Ces classes doivent alors obligatoirement implémenter la méthode.

La classe *Firefighter*, littéralement « Pompier » en anglais, est l’élément central de notre projet. Il utilise les classes des capteurs afin d’obtenir les différentes mesures, qu’il traite par la suite afin d’obtenir une évaluation des risques d’incendie. C’est également elle qui publie les données sur le *Broker MQTT*.

1. Réalisation
   1. Client MQTT
      1. Récupération des mesures

Afin de faciliter la manipulation des mesures, nous avons mis en place une classe *Measure*, ayant comme attribut : La température, l’humidité, la pression, la détection de flamme (booléen) ainsi que la date à laquelle ces valeurs ont été mesurées.

Nous avons repris et adapté les implémentations des différents capteurs réalisé en classe lors de ce cours. Toutes nos classes capteurs, comme expliqué précédemment, héritent de la classe abstraite *Captor*.

* + 1. Analyse des mesures

Afin d’économiser de l’espace de stockage, nous avons décidé d’effectuer la détection de flamme sur le Raspberry Pi, et d’envoyer uniquement le résultat (booléen).

Nous avons utilisé le programme *fire-detection-cnn*, qui permet de détecter des feux en temps-réel sur une vidéo / image, en utilisant un réseau neuronal de convolution (deep learning). Nous n’expliquerons pas les détails du fonctionnement, mais nous vous invitons à lire les documents réalisés par les chercheurs et programmeurs.

Comme stipulé dans la licence (disponible sur leur repository GitHub), nous avons le droit de récupérer, modifier et publier une copie du programme.

* + 1. Envoi au brocker
  1. Stockage des mesures sur InfluxDB
  2. Dashboard sur Grafana

1. Tests / Résultats
2. Conclusion
3. Bibliographies et références

[*Experimentally defined Convolutional Neural Network Architecture Variants for Non-temporal Real-time Fire Detection*](https://breckon.org/toby/publications/papers/dunnings18fire.pdf) (Dunnings, Breckon), In Proc. International Conference on Image Processing, IEEE, 2018. <https://breckon.org/toby/publications/papers/dunnings18fire.pdf>

[*Experimental Exploration of Compact Convolutional Neural Network Architectures for Non-temporal Real-time Fire Detection*](https://breckon.org/toby/publications/papers/samarth19fire.pdf) (Samarth, Bhowmik, Breckon), In Proc. International Conference on Machine Learning Applications, IEEE, 2019. <https://breckon.org/toby/publications/papers/samarth19fire.pdf>

1. Tables des illustrations