# Rapport Conception logiciel TP4

## 1) Contexte

L'objectif de ces TPs est la mise en place d'une architecture microservices pour la gestion automatique d'une maison. On a mis en place des scénarios qui permettent de gérer automatiquement la température, l'humidité et la luminosité dans une pièce.

Pour ce faire, nous avons utilisé un module ESP8266, un capteur de température et d'humidité, un capteur de luminosité, une LED, un bouton poussoir et un afficheur LCD.

## 2) Conception

Diagramme cas d'utilisation du micro-service de température.

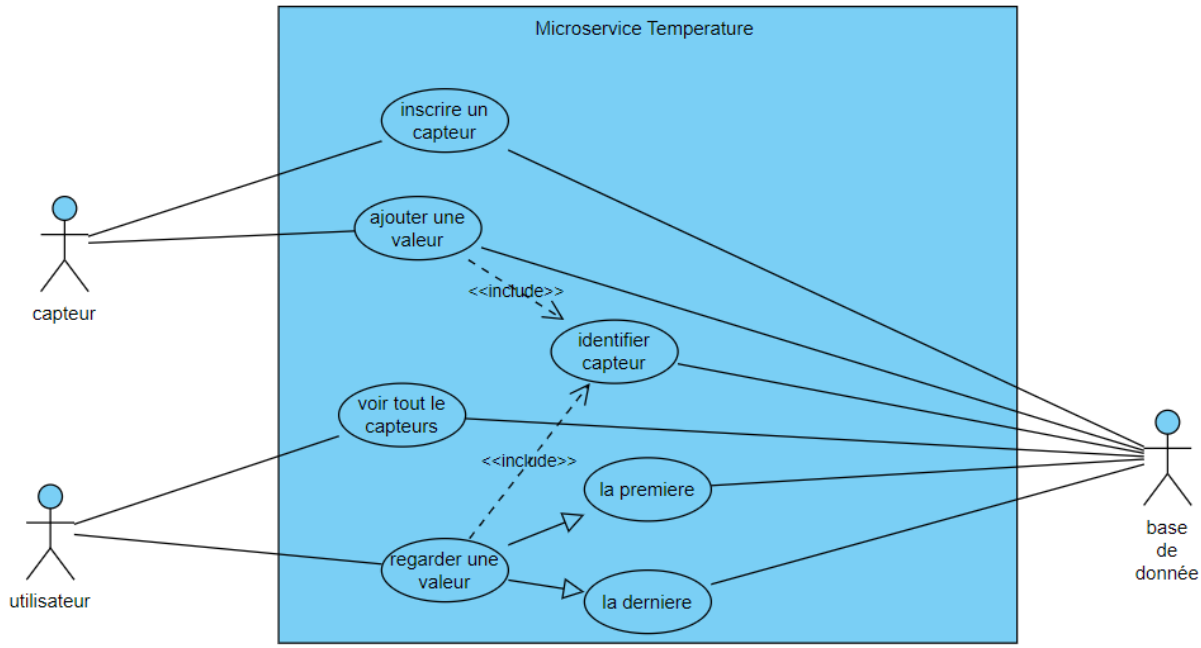
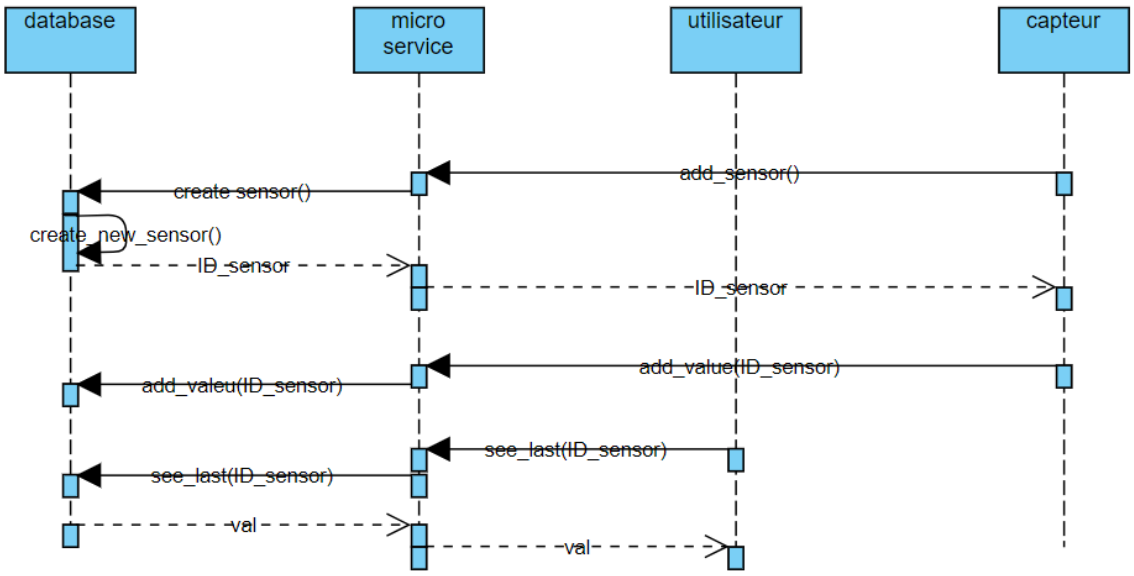


Diagramme séquence de l'utilisation du micro service température




Le programme du nodeMCU :

- Initialise la connexion wifi prédéfinie
- Etablie une connexion avec les 4 micro-services pour créer un nouveau capteur de chaque micro-services et récupérer leurs UUID.
- Ouvre une écoute pour pouvoir changer la couleur de la LED ou l'affichage du LCD
- Envoie au micro-services les données collectées toutes les secondes.

Voici l'API d'un microservice.
GET à http://localhost:8081/Luminosity/all : affiche la liste de tous les capteurs
GET à http://localhost:8081/Luminosity/{id} : affiche un capteur par son Id.
GET à http://localhost:8081/Luminosity/{id}/la : affiche la dernière valeur mesurée.
GET à http://localhost:8081/Luminosity/{id}/ol : affiche la plus ancienne valeur mesurée.
POST à http://localhost:8081/Luminosity/addSensor : ajoute un capteur. N'a pas besoin de body.

POST à http://localhost:8081/Luminosity/addValue/{id} : ajoute une valeur pour un capteur. body: un float

Nous avons utilisé une database pour chaque micro-service pour les rendre facilement portable et résiliant aux pannes du micro-service.

application IoT :

Il y a d'abord un bouton relier à un if pour déterminer si le bouton est en on ou off.

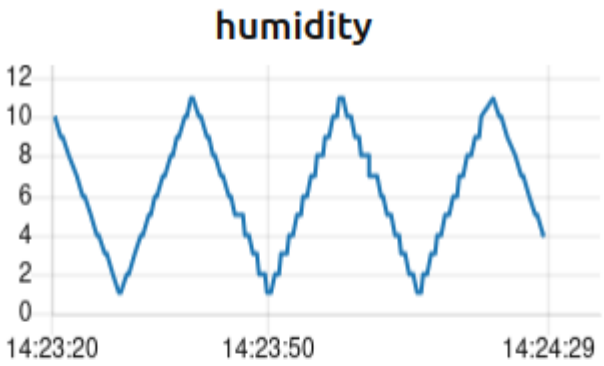
Ce if est relier a deux requête http qui allume ou éteigne la LED.

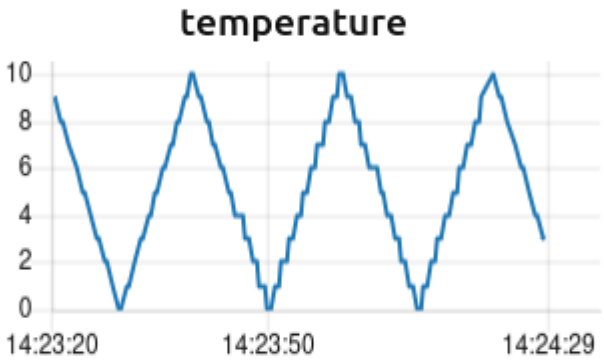
Il y a un autre bouton qui permet de démarrer les diagramme.

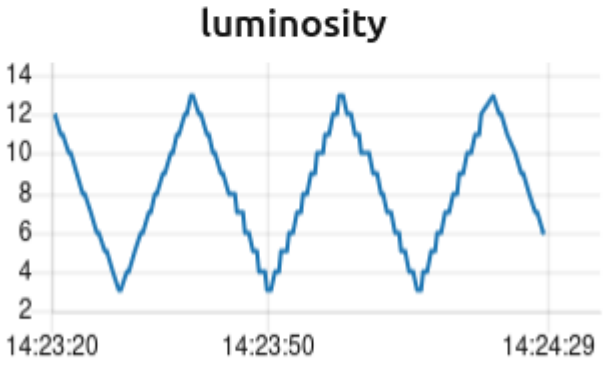
Pour chaque diagramme une requête Get est effectuer pour récupérer la nouvelle valeur. La valeur est ajoutée au diagramme.

On attend 1 sec et on recommence.

Default

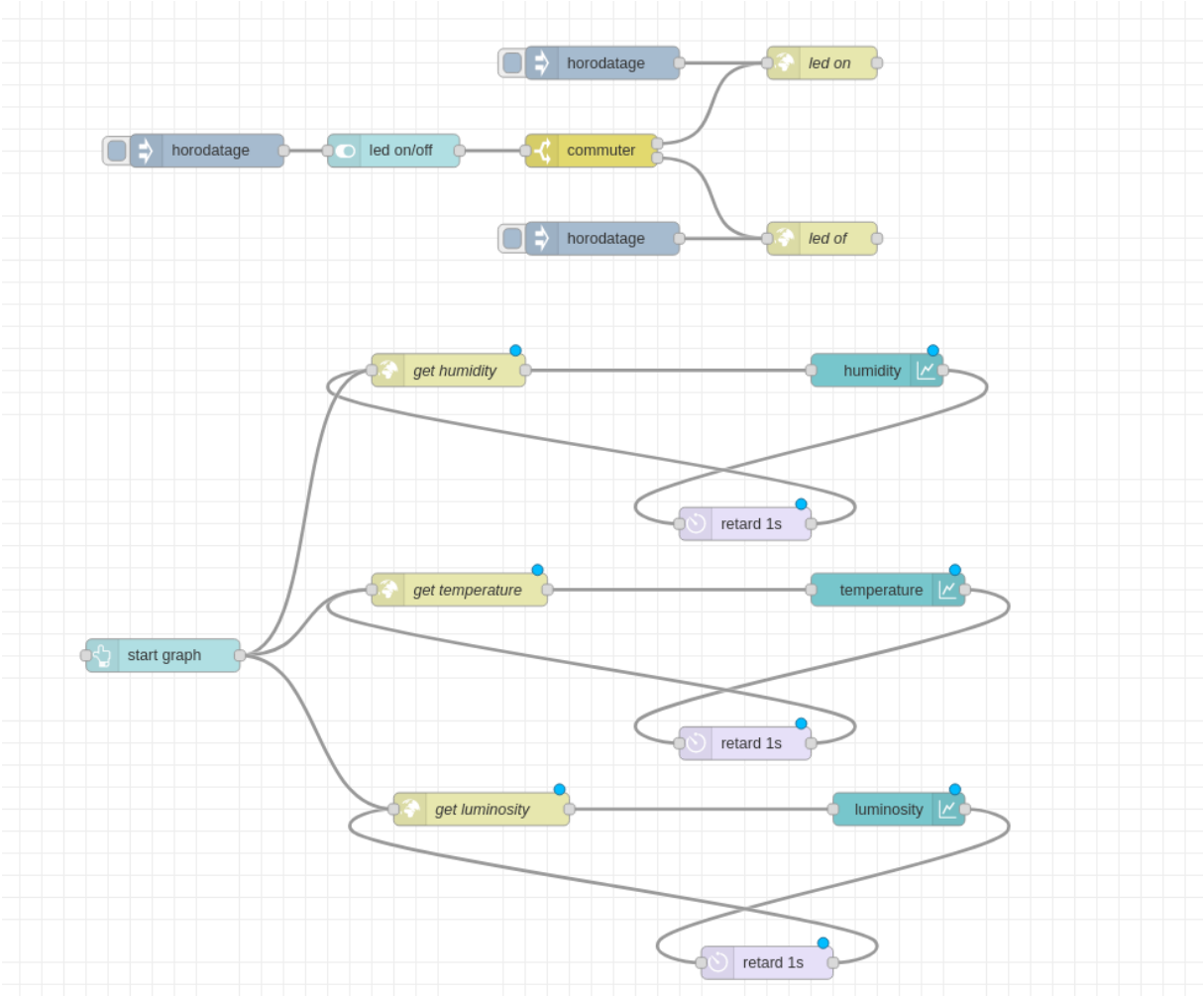



START GRAPH

led on/off




Limit :

IL n'y a pas vraiment de problème à déployer la solution à grande échelle.

Tant que tout les nodeMCU sont connecté à internet et que les micro-service sont ouvert sur une adresse public tout les bodeMCU pourront fonctionner correctement et les micro-services devraient supporter la charge tant qu'on les fait tourner sur des machines corrects.

Il y a deux problèmes principaux :

- Le manque de visibilité de l'ensemble :
  - On ne sait pas si un node a un problème ou non
  - On voit mal la globalité
- Il faut un orchestrateur par node se qui commence à être un problème.

Proposition :

Un modèle MVC serai plus adapté pour englobé l'orquestrateur et le node-red dans le service se qui serait beaucoup plus léger et beaucoup plus simple à mettre en place.

Lien vers le repo git : git@github.com:Nastaryn/ING2-TP3-Conception.git

DI SANTO Alexandre

ROYER Julia