

Qarpediem:

Petit Guide du développeur

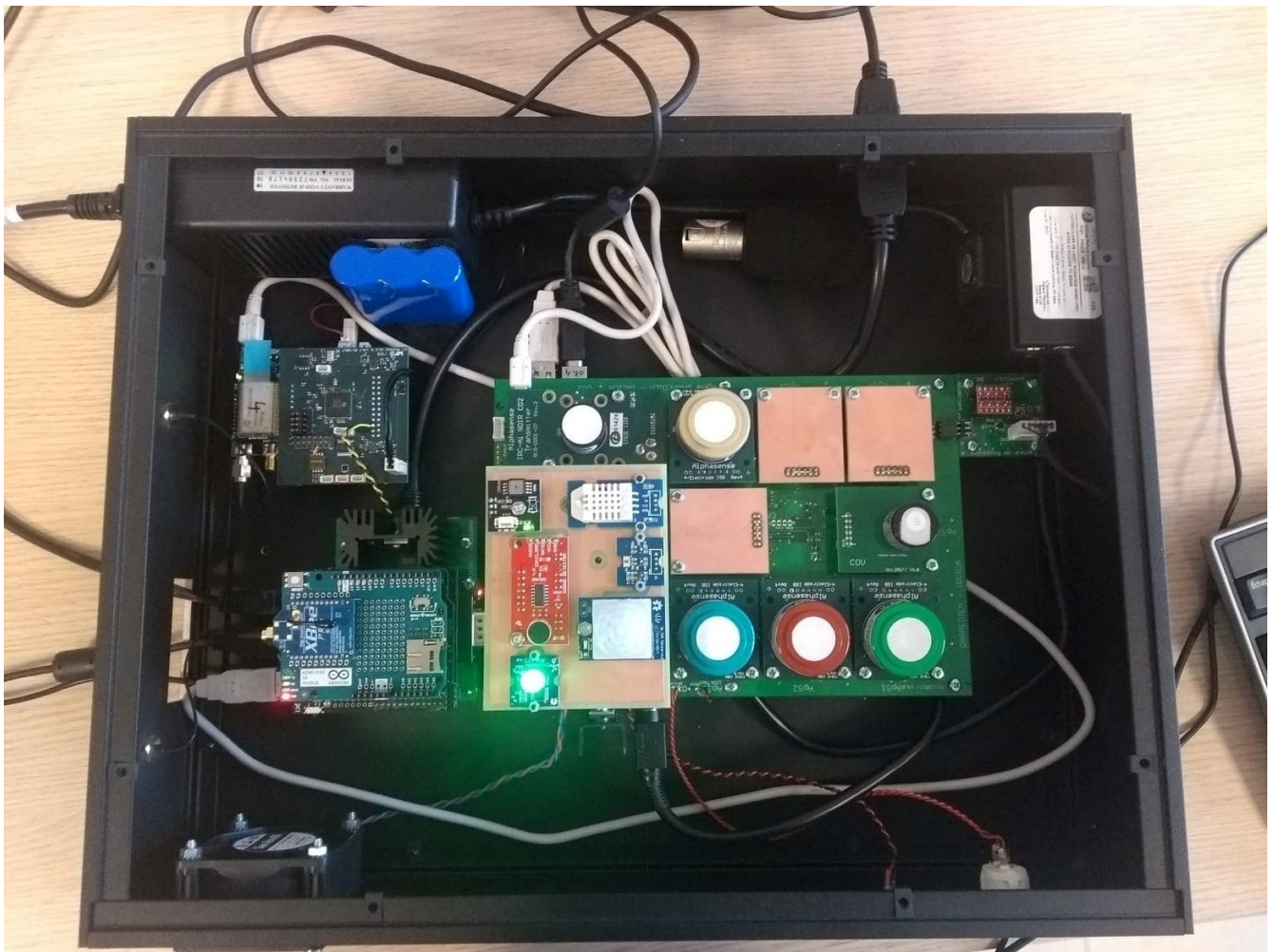


Table des matières

1) Introduction.....	3
2) L'Arduino.....	4
La mise en place de l'environnement de développement.....	4
3) Le Raspberry pi.....	6
Le datapoller.....	6
Le datasender.....	6
Le production de l'exécutable.....	6
Lancement du programme au démarrage du système.....	10
4) Recommandations :.....	12

1) Introduction

Le Projet Qarpediem est un ensemble de nœuds constitués de plusieurs capteurs permettant de mesurer la qualité de l'air. Avec l'évolution du projet il est apparu nécessaire d'écrire un petit guide permettant à chaque nouvelle personne devant y travailler de se retrouver facilement pour se concentrer à l'essentiel.

Ce document a pour objectif donc d'expliquer comment mettre en place l'environnement de développement pour un nœud, et on se concentre plutôt sur la partie software.

Un nœud est composé de deux parties distinctes :

- une partie pilotée par un **Arduino Mega** qui récupère les données depuis les capteurs ;

- une autre partie pilotée par un Raspberry pi3 qui reçoit les données depuis l'Arduino et les renvoi à la station centrale de traitement .

Tout le code source du programme est disponible sur GitHub à cette adresse:

<https://github.com/Alpha-balde/QarpediemProject>

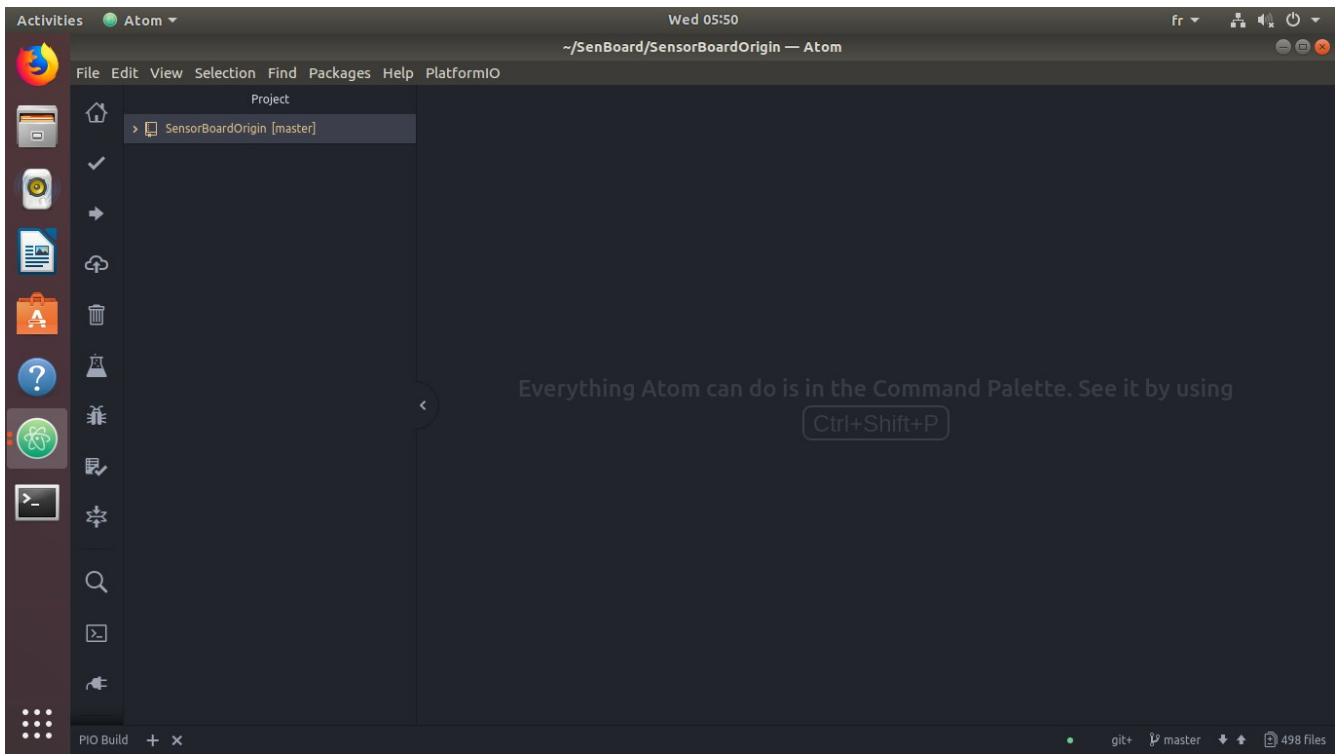
ou <https://github.com/Apolline-Lille/apolline-qarpediem> (avec des erreurs de compilation)

2) L'Arduino

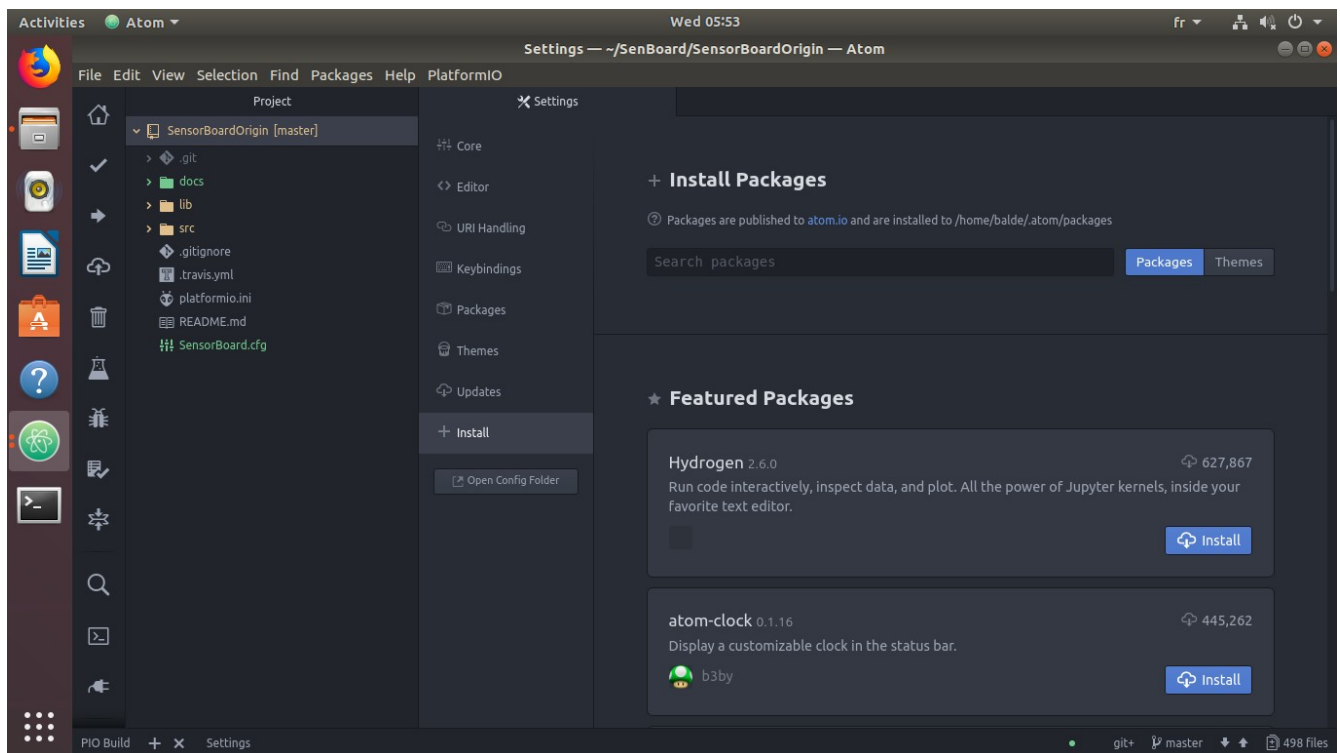
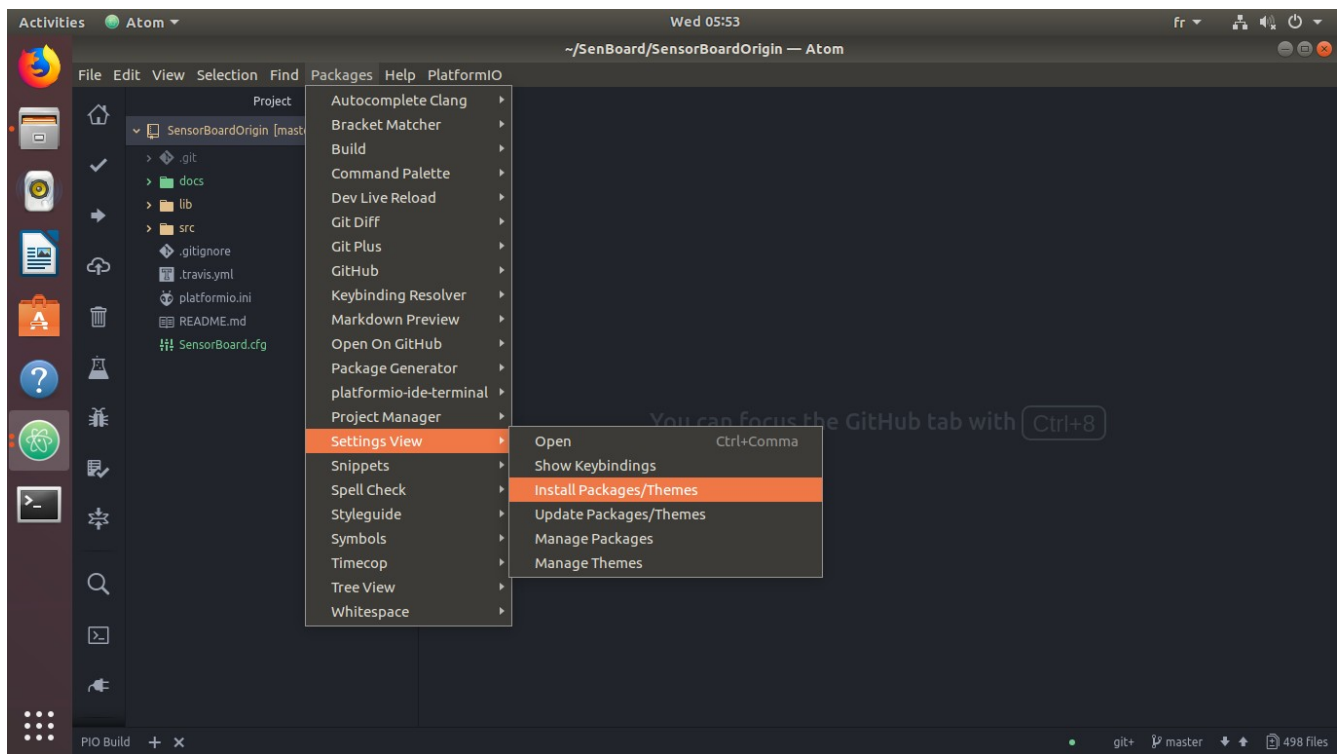
Pour récupérer les données des 15 capteurs, l'Arduino est programmé en c++ avec l'**IDE Atom** et **PlatformIO sous Linux Ubuntu**. Le capteur de particules OPC_N2 utilise une bibliothèque externe écrite en C qui se trouve dans le dossier **lib** du code source.

La mise en place de l'environnement de développement

La première chose à faire c'est d'installer Atom en se référant à la procédure décrite sur le site web officiel(<https://ide.atom.io>).



Ensuite il faut exécuter la commande **apt-get install clang** en root(ou sudo).
PlatformIO utilise Clang pour l'auto-complétion des commandes
En fin chercher et installer le package PlatformIO ide



ce lien explique pas à pas la procédure d'installation : <https://projetsdiy.fr/bien-demarrer-platformio-ide-arduino-esp8266-esp32-stm32> (dernière visite le 10-10-18 à 14h50)

3) Le Raspberry pi

Pour se connecter il faut utiliser le nom d'utilisateur **qarpediem** avec le mot de passe **123**.

Cette partie du projet est composée de trois programmes, et de deux scripts shells de lancement et d'arrêt (Start_qarpediem & Stop_qarpediem). Le SensorView qui est une application graphique qui permet de configurer et visualiser les données ne serra pas aborder ici.

Le Raspberry utilise deux base de données SQLITE , une pour stocker la configuration globale(**config.db**) et une autre pour stocker les données(**sensors.db**).

Le datapoller

c'est le programme qui récupère les données depuis l'Arduino, il interroge ce dernier toutes les 30 secondes et stocke les données dans la base **sensors.db**. Le Raspberry utilise une **liaison Serie** pour communiquer avec l'Arduino. Normalement tout est bien mis en œuvre et on n'a pas à se soucier du protocole de communication.

Le datasender

Ce dernier programme lit les données depuis la base de données **sensors** et les envoie vers la station centrale(serveur) toutes les 30 secondes. Pour l'instant **seul mode de communication IP est disponible**, mais il est envisager d'utiliser à l'avenir Lorawan ou XBEE

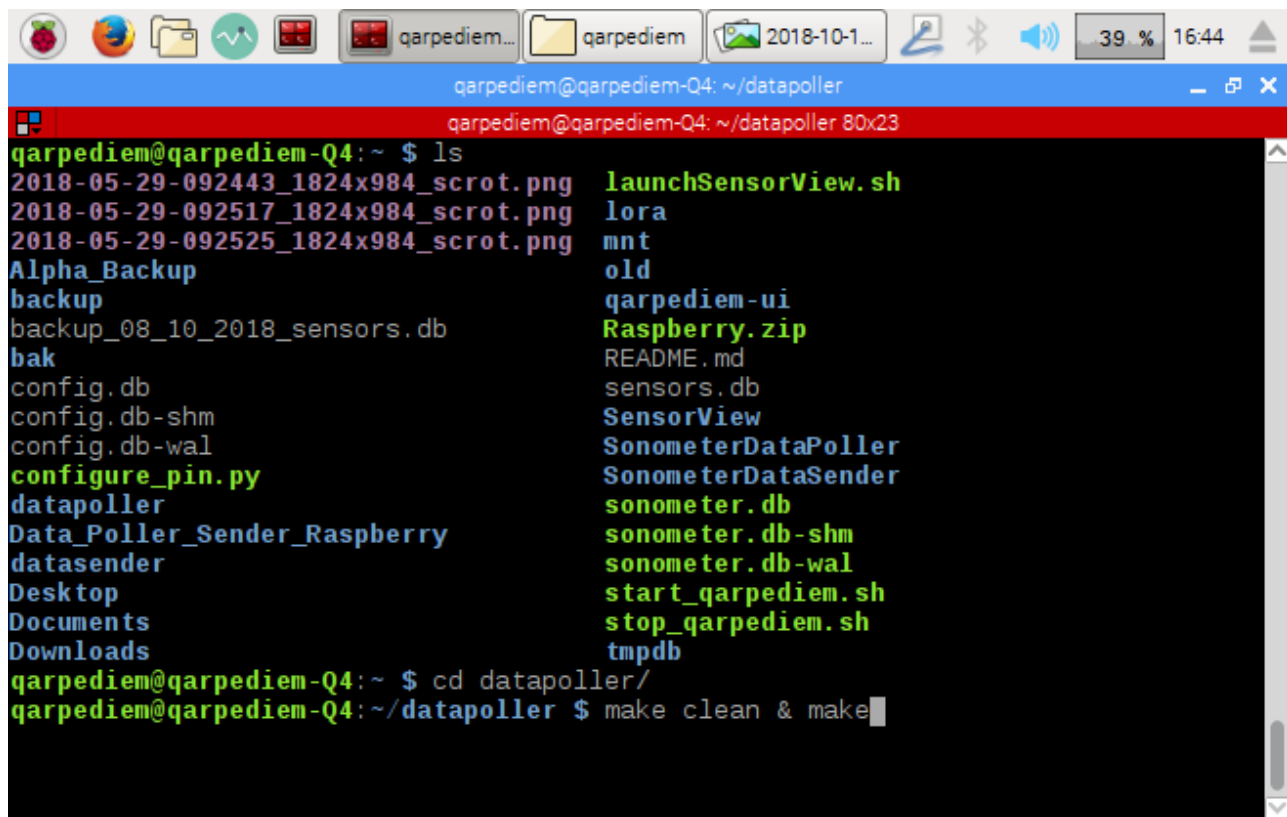
Le production de l'exécutable

Si on modifie le programme et qu'on veut créer un nouveau exécutable, il faut:

➔ arrêter le programme:sudo ./Stop_qarpediem;

```
qarpediem@qarpediem-Q4: ~ $ ls
2018-05-29-092443_1824x984_scrot.png  launchSensorView.sh
2018-05-29-092517_1824x984_scrot.png  lora
2018-05-29-092525_1824x984_scrot.png  mnt
Alpha_Backup                          old
backup                                 qarpediem-ui
backup_08_10_2018_sensors.db          Raspberry.zip
bak                                    README.md
config.db                             sensors.db
config.db-shm                         SensorView
config.db-wal                         SonometerDataPoller
configure_pin.py                      SonometerDataSender
datapoller                           sonometer.db
Data_Poller_Sender_Raspberry          sonometer.db-shm
datasender                           sonometer.db-wal
Desktop                              start_qarpediem.sh
Documents                             stop_qarpediem.sh
Downloads                             tmpdb
qarpediem@qarpediem-Q4:~ $ sudo ./stop_qarpediem.sh
```

→ faire un `:(cd datapoller ou datasender) & make clean et make` pour produire l'exécutable



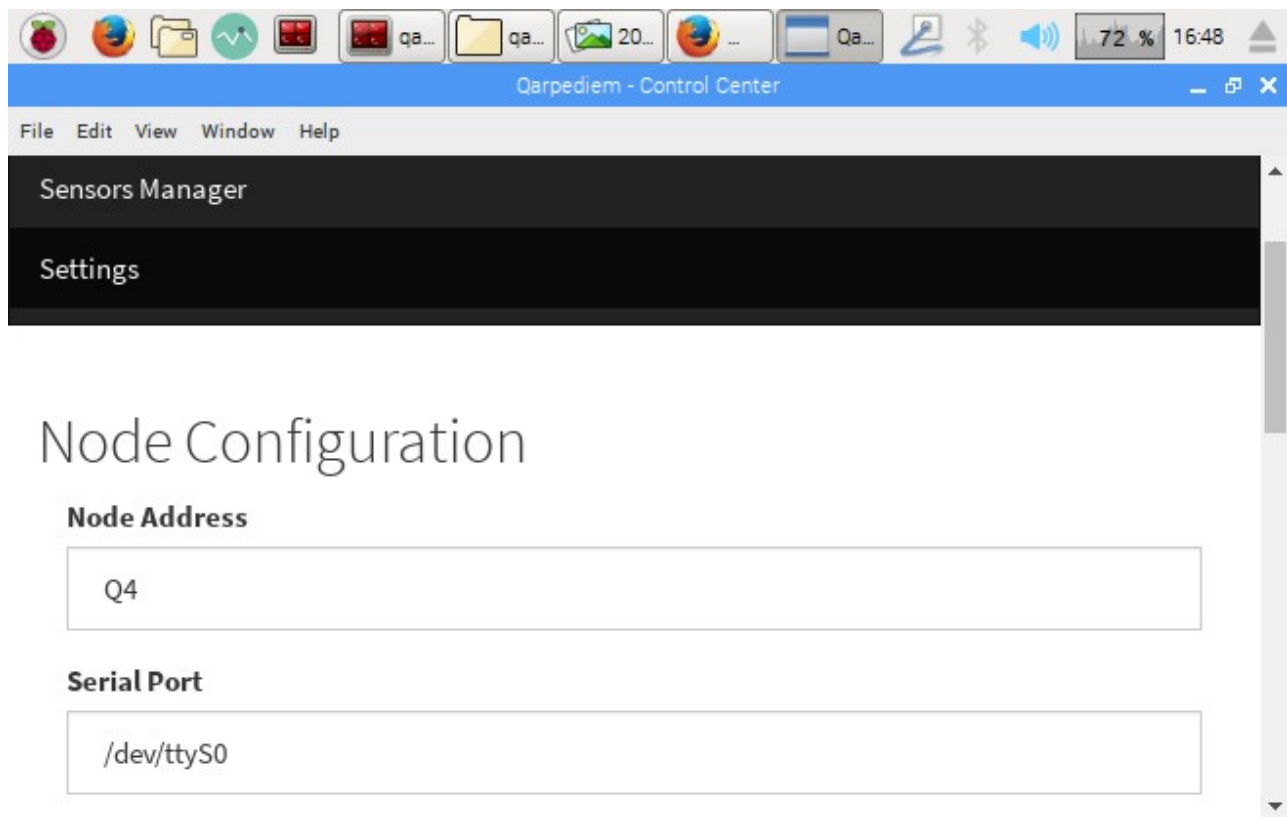
```
qarpediem@qarpediem-Q4: ~ $ ls
2018-05-29-092443_1824x984_scrot.png  launchSensorView.sh
2018-05-29-092517_1824x984_scrot.png  lora
2018-05-29-092525_1824x984_scrot.png  mnt
Alpha_Backup                          old
backup                                qarpediem-ui
backup_08_10_2018_sensors.db          Raspberry.zip
bak                                    README.md
config.db                             sensors.db
config.db-shm                         SensorView
config.db-wal                         SonometerDataPoller
configure_pin.py                      SonometerDataSender
datapoller                            sonometer.db
Data_Poller_Sender_Raspberry          sonometer.db-shm
datasender                           sonometer.db-wal
Desktop                               start_qarpediem.sh
Documents                             stop_qarpediem.sh
Downloads                             tmpdb
qarpediem@qarpediem-Q4:~ $ cd datapoller/
qarpediem@qarpediem-Q4:~/datapoller $ make clean & make
```

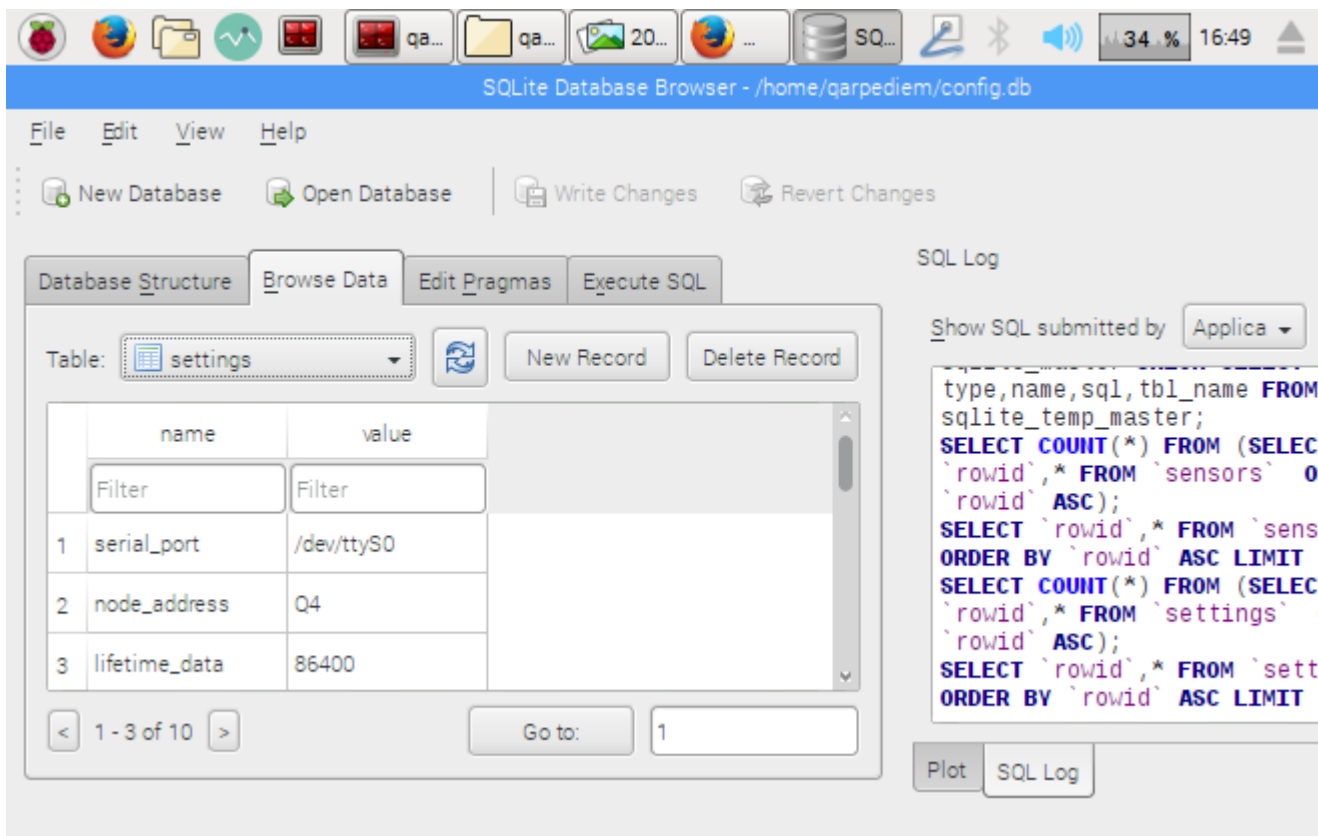
- lancer la commande `sudo ./start_qarpediem` pour créer les bases de données
- changer les droits d'accès afin que l'application graphique puisse lire et écrire dans la base de données `sensors.db`.

```
sudo chown qarpediem *.db*
sudo chgrp qarpediem *.db*
```

Si on oublie d'effectuer cette action on ne pourra pas utiliser l'application graphique correctement.

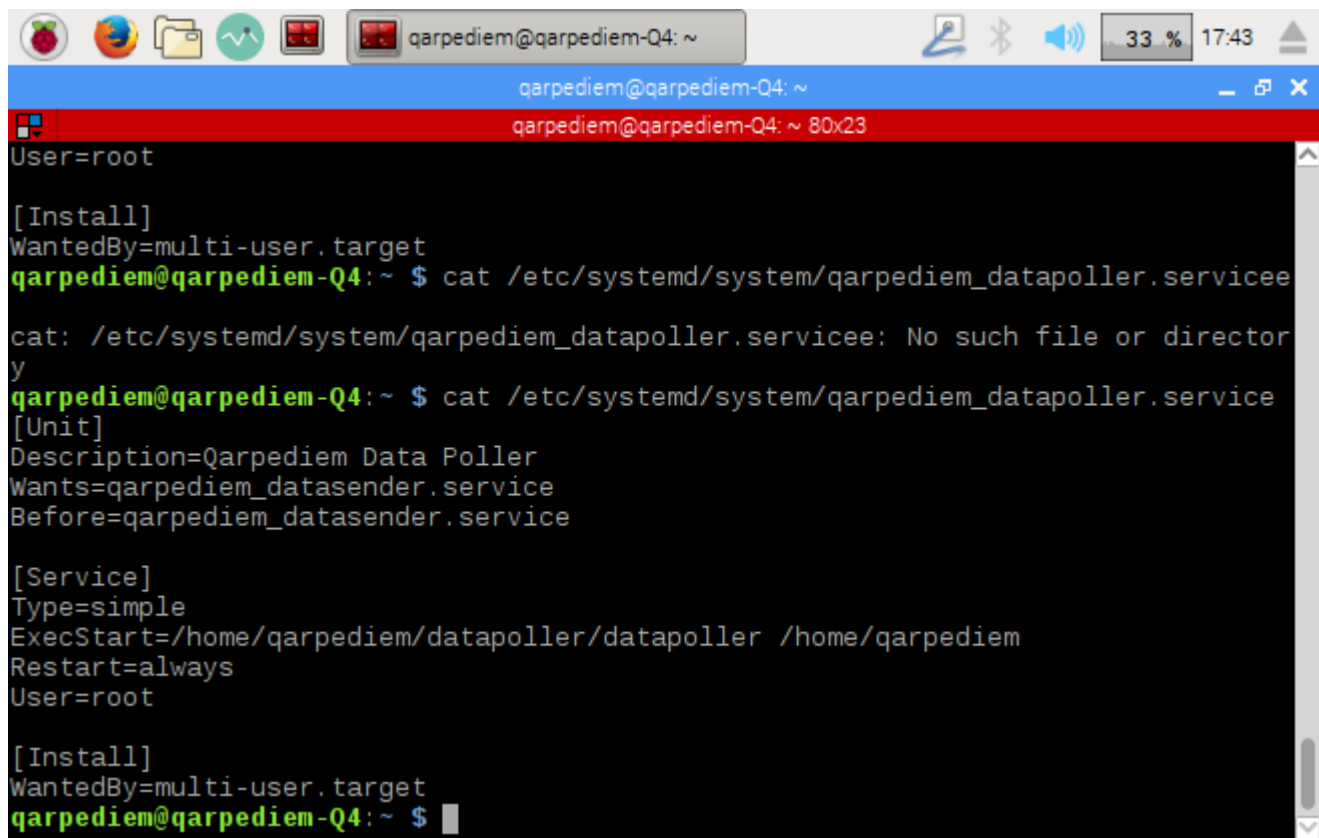
- Accéder à la base de données `config` pour modifier le champ `Node_address` afin de spécifier le nœud qu'on utilise. Pour cela on peut passer soit par l'application graphique `Sensor Manager`, soit en ouvrant directement la base avec `SQLite Database Browser`, en faisant clic droit sur la db





Lancement du programme au démarrage du système

Si on veut arrêter le lancement automatique du programme au démarrage, il faut supprimer ou tout simplement mettre en commentaire le contenu des ces deux fichiers(pour plus d'information se renseigner sur le systemd Linux :<http://lea-linux.org/documentations/Systemd>):

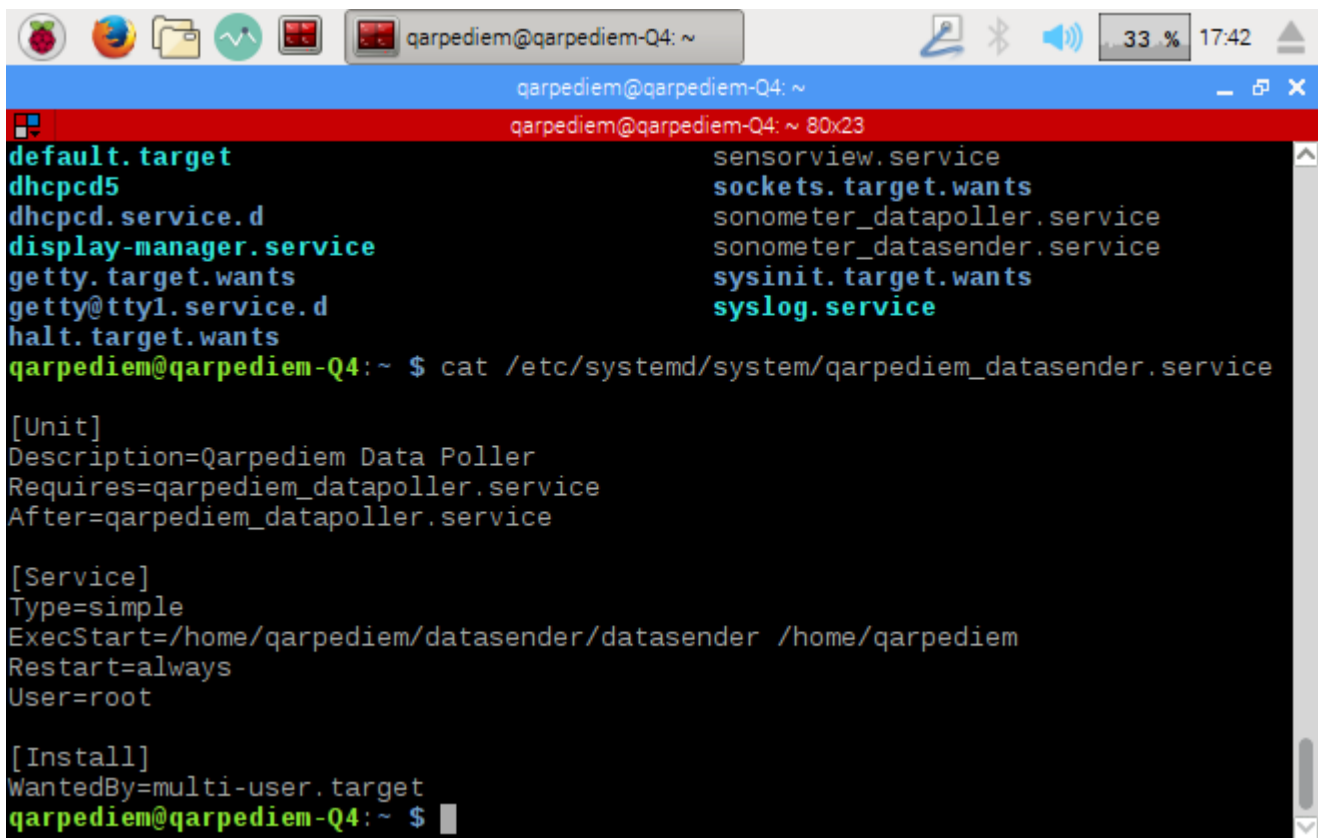
A terminal window titled 'qarpediem@qarpediem-Q4: ~' with a red title bar. The terminal shows the output of 'cat /etc/systemd/system/qarpediem_datapoller.service'. The content includes sections for [Unit], [Service], and [Install]. The [Unit] section has WantedBy=multi-user.target. The [Service] section has Type=simple, ExecStart=/home/qarpediem/datapoller/datapoller /home/qarpediem, Restart=always, and User=root. The [Install] section has WantedBy=multi-user.target. The prompt qarpediem@qarpediem-Q4:~ \$ is visible at the end of the output.

```
qarpediem@qarpediem-Q4: ~
qarpediem@qarpediem-Q4: ~ 80x23
User=root

[Install]
WantedBy=multi-user.target
qarpediem@qarpediem-Q4:~ $ cat /etc/systemd/system/qarpediem_datapoller.service
cat: /etc/systemd/system/qarpediem_datapoller.service: No such file or director
y
qarpediem@qarpediem-Q4:~ $ cat /etc/systemd/system/qarpediem_datapoller.service
[Unit]
Description=Qarpediem Data Poller
Wants=qarpediem_datasender.service
Before=qarpediem_datasender.service

[Service]
Type=simple
ExecStart=/home/qarpediem/datapoller/datapoller /home/qarpediem
Restart=always
User=root

[Install]
WantedBy=multi-user.target
qarpediem@qarpediem-Q4:~ $
```



```
qarpediem@qarpediem-Q4: ~  
qarpediem@qarpediem-Q4: ~ 80x23  
default.target                                sensorview.service  
dhcpcd5                                       sockets.target.wants  
dhcpcd.service.d                             sonometer_datapoller.service  
display-manager.service                     sonometer_datasender.service  
getty.target.wants                           sysinit.target.wants  
getty@tty1.service.d                         syslog.service  
halt.target.wants  
qarpediem@qarpediem-Q4:~ $ cat /etc/systemd/system/qarpediem_datasender.service  
  
[Unit]  
Description=Qarpediem Data Poller  
Requires=qarpediem_datapoller.service  
After=qarpediem_datapoller.service  
  
[Service]  
Type=simple  
ExecStart=/home/qarpediem/datasender/datasender /home/qarpediem  
Restart=always  
User=root  
  
[Install]  
WantedBy=multi-user.target  
qarpediem@qarpediem-Q4:~ $
```

pour afficher les log on utilise la commande systemctl en root(ou sudo):

```
sudo systemctl status qarpediem_datasender.service  
sudo systemctl status qarpediem_datapoller.service -l  
sudo systemctl status qarpediem_datasender.service  
sudo systemctl status qarpediem_datapoller.service -l
```

4) Recommendations :

- Pour l'Arduino si on veut récupérer et envoyer des nouvelles données d'un capteur(existant ou nouveau) on n'a pas besoin de modifier le protocole de communication mise en place, il suffit de lire les données en respectant les caractéristiques soft(ex :bibliothèques externes) et hard(ex :numérique ou analogique) du capteur et modifier les fichiers SensorServer.h(déclaration de fonctions) et SensorServer.cpp(définition de fonctions) ,seulement ces deux fichiers pour inclure les nouvelles données en utilisant la fonction `send_response_ok()` .

- Du côté du Raspberry il y a des choses que j'ai pas compris et dont j'ai pas mentionner avant, c'est notamment la manière dont les données sont envoyées à la station centrale. Au départ j'ai cru que le datasender lit les données dans la base sensors.db et utilise l'adresse IP du serveur renseigner dans la champ server_host de la base config.db pour les envoyées, mais après plusieurs tentatives échouées, j'ai constaté qu'il y a un service qui tourne sur le port 5555 du Raspberry avec lequel le datasender communique et qui(il me semble) est chargé d'envoyées les données au serveur.
- On n'a pas besoin d'installer des nouvelles outils pour programmer en c++ sur le Raspberry, mais pour une question de performance et surtout d'harmonisation je suggère qu'on utilise **le framework Qt Creator** qui est open source. Avec son mécanisme de signaux et de slots il permet d'écrire d'application très performantes .