CoBox

Github : github.com/hackersoft13/Cobox
Site web : cobox.paulitow.fr

RAPPORT D'ACTIVITE

1ère et 2ème itération

Projet de système d'exploitation

Moyse – Desmarets – Ilhe - Legeas

ISEN | école d'ingénieurs
ALL IS DIGITAL!

TABLE DES MATIERES

I. Rappel

I.1 Besoins

Le but du projet Cobox est de concevoir une solution de domotique pour particuliers, permettant le contrôle et la supervision de différents éléments (tout équipement pouvant être commandé par un relais) au sein d'un foyer.

Cette solution se décomposera en une centrale domotique, et X clients. Les clients étant les actionneurs / capteurs.

A ce jours les besoins concernant la centrale sont :

- Echanger des informations à ses actionneurs en mode serveur via Wifi directe (mode hotspot)
- Afficher de manière claire les mesures des différents éléments supervisés par la centrale

Et concernant la partie client :

- Echanger des informations à la centrale en mode client via Wifi directe
- Piloter les actionneurs via des GPIO*
- Acquisition de données via des capteurs (Température / humidité…)

1.2. Contrainte

Les contraintes de ce projet ont été définies en accord avec notre client et sont susceptibles d'être modifiées avec l'évolution du présent document.

Contraintes	
CENTRALE	La centrale doit comporter un système d'exploitation linux
	La centrale doit utiliser un programme développé en C pouvant dialoguer avec les différents actionneurs
ACTIONNEUR	L'actionneur doit être capable de dialoguer via Wifi
	L'actionneur doit être capable de piloter jusqu'à 8 relais
	L'actionneur doit être capable de monitorer un capteur et de transmettre sa valeur à la centrale

I.3. Roadmap

Ci-dessous, la Roadmap définissant les étapes cruciales de notre projet

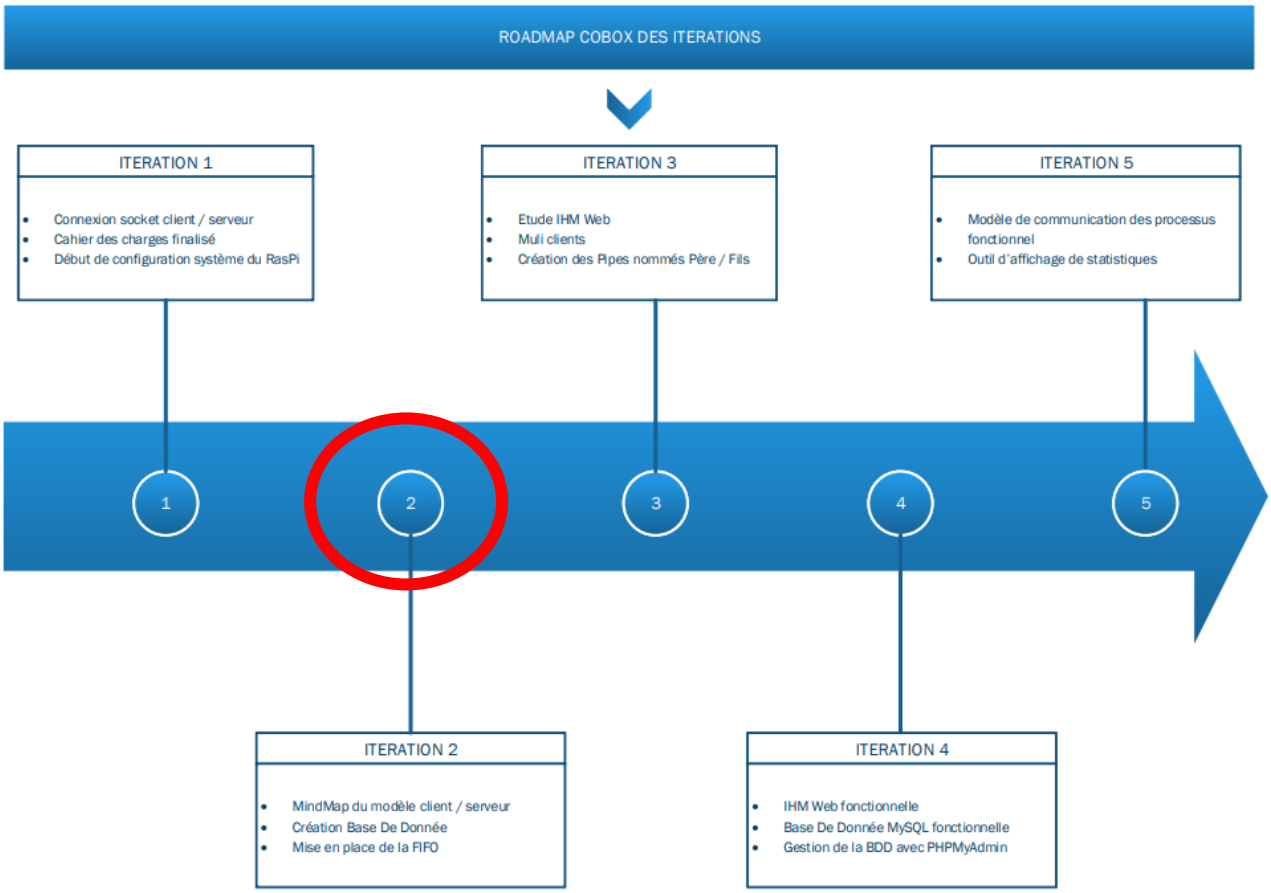


Figure 1 : Roadmap Cobox

Nous sommes donc à l'heure de la rédaction à l'itération n°2

2.1. 1ère itération – 30/11/2019 12:30

Ci-dessous sont détaillées les avancés récentes sur notre projet

2.1.1. Cobox Serveur - Hardware

2.1.1.1. Effectif à ce jour

La configuration de la RaspBerry est fonctionnelle :

- Raspbian est installé
- La configuration du serveur SSH est opérationnelle
- Le Wifi en mode AP est opérationnelle
- Les librairies nécessaires au fonctionnement sont installées
- La structure de la base de données est créée

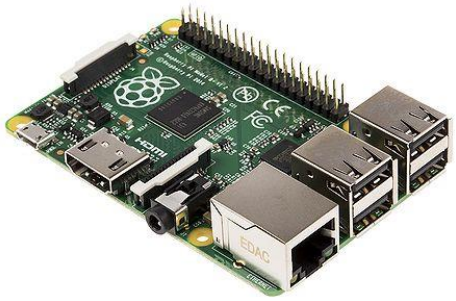


Figure 2 : RaspBerry 3 model B+

2.1.1.2. Prochaine étape

Les prochaines taches à effectuer sont :

- Installation des composant Web pour réaliser l'étude de l'IHM Web
- Intégration des sources sur la raspberry

2.1.2. Cobox Serveur - Software

Le logiciel qui se chargera de gérer les connexions des clients est en construction voici l'avancement global à la première itération.

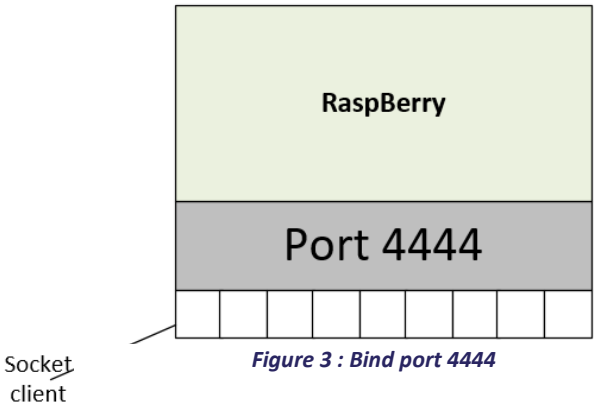


Figure 3 : Bind port 4444

Le serveur est capable d'interagir avec l'actionneur (ESP8266) au travers de trames TCP/IP en allouant un processus fils à chaque connexion.

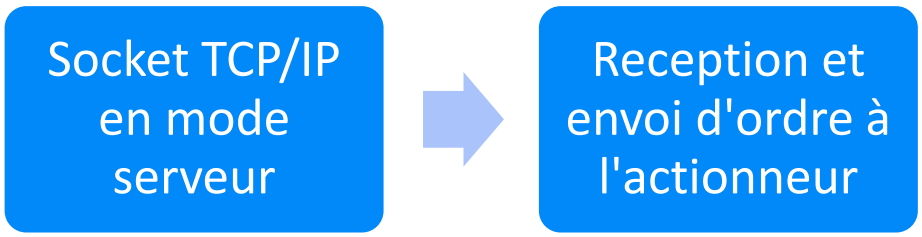


Figure 4 : état actuel

Création de socket

```
sockfd = socket(AF_INET, SOCK_STREAM, 0);
      if(sockfd < 0){
            printf("[-]Error in connection.\n");
            exit(1);
      }
      printf("[+]Server Socket is created.\n");

      memset(&serverAddr, '\0', sizeof(serverAddr));
      serverAddr.sin_family = AF_INET;
      serverAddr.sin_port = htons(PORT);
      serverAddr.sin_addr.s_addr = inet_addr("0.0.0.0");

      ret = bind(sockfd, (struct sockaddr*)&serverAddr, sizeof(serverAddr));
      if(ret < 0){
            printf("[-]Error in binding.\n");
            exit(1);
      }
      printf("[+]Bind to port %d\n", 4444);

      if(listen(sockfd, 10) == 0){
            printf("[+]Listening....\n");
      }else{
            printf("[-]Error in binding.\n");
      }


      while(1){
            newSocket = accept(sockfd, (struct sockaddr*)&newAddr,
&addr_size);
            if(newSocket < 0){
                  exit(1);
            }
            printf("Connection accepted from %s:%d\n",
inet_ntoa(newAddr.sin_addr), ntohs(newAddr.sin_port));

            if((childpid = fork()) == 0){
```

La suite du code correspond au traitement interne des données et est alimentée en temps réel via notre github commun de travail dont vous retrouvez l'url en première page.

2.2. 2nd itération – 30/11/2019 19:30

2.2.1. Documentation

Vous pouvez retrouver la mindmap du fonctionnement global de notre programme Cobox serveur ci-dessous :

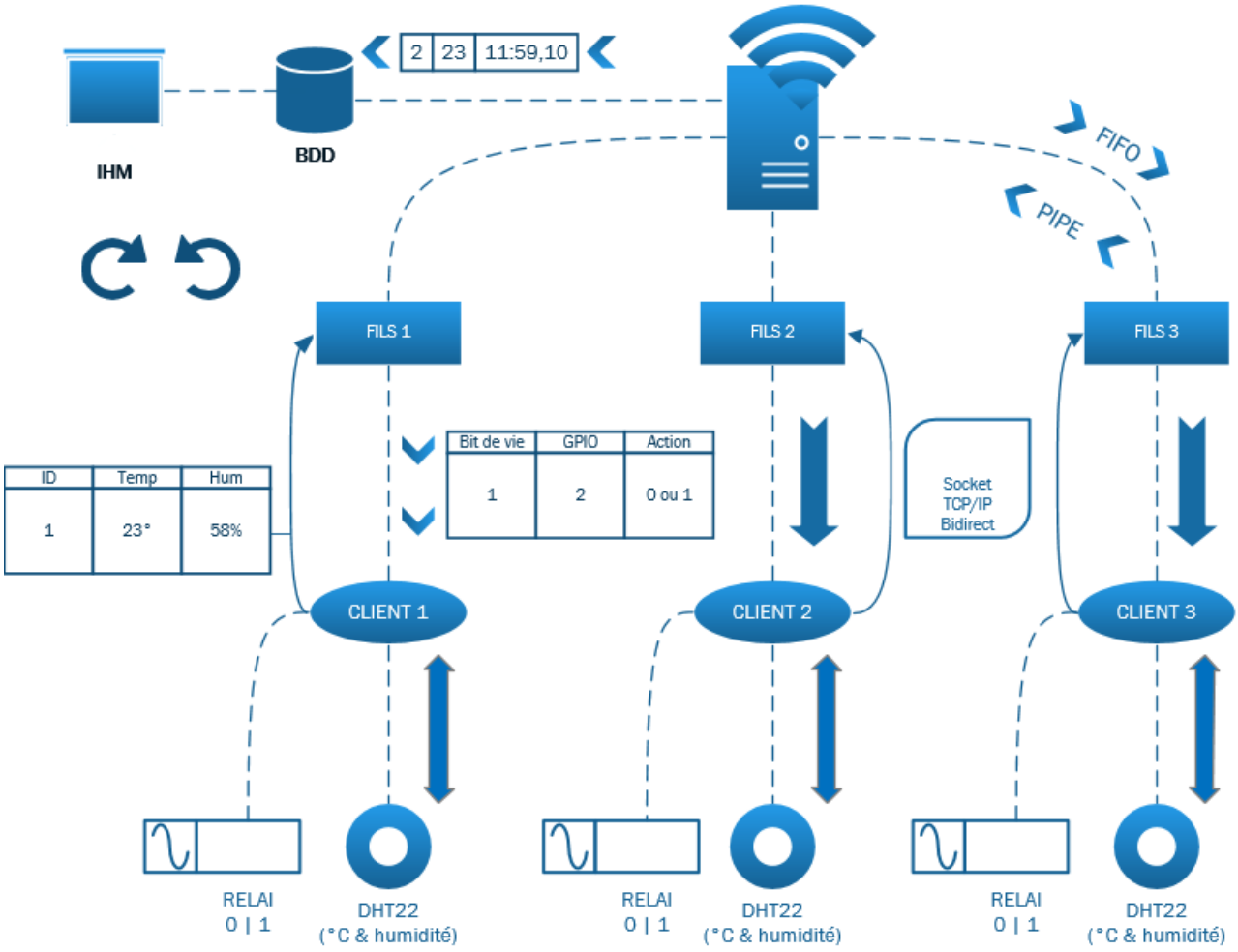


Figure 5 : Synoptique de fonctionnement

2.2.2. Base de donnée

La structure de base de donnée est établie comme détaillées ci-dessous :

Id_mesure	Id_Device	Temp	hum	timestamp
1	1	23	52	1575139370

Elle a été créer sur une VM Ubuntu via le système de gestion de base de données MySQL. Elle est complétée, pour des facilités d'utilisation par une instance PhpMyAdmin sur une serveur Web Apache2.

2.2.3. Cobox Serveur – Software

L'itération de ce jour consiste en la mise en place d'une FIFO pour le retour des clients vers le serveur. Chaque client peut avoir accès à la FIFO.

La fifo actuelle, créée avec mkfifo, est susceptible d'être remplacée par une méthode de communication plus performante.

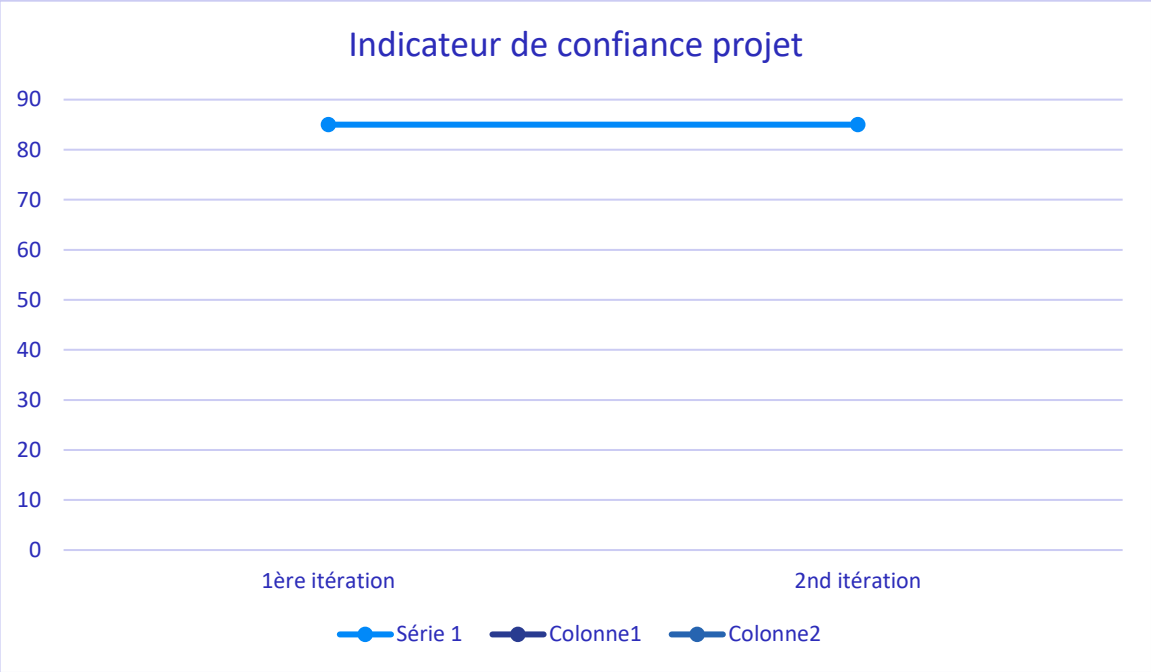
Utilisation de la FIFO en écriture

```
fd=open("fifo", O_RDWR);
write(fd, trame, (sizeof(trame)+1));
```

Utilisation de la FIFO en lecture

```
fd = open("fifo", O_RDONLY | O_NONBLOCK);
read(fd, trame, (sizeof(trame)+1));
```

2.2.2. Indicateurs



- 1ère itération : **85 %**
- 2nd itération : **85%**