|  |
| --- |
| RAPPORT DE PROJET FINAL |

|  |
| --- |
| CoBox  Github :  Site web : cobox.paulitow.fr |

|  |  |
| --- | --- |
| Projet de système d’exploitation  Moyse – Desmarets – Ilhe - Legeas | Résultat de recherche d'images pour "isen" |

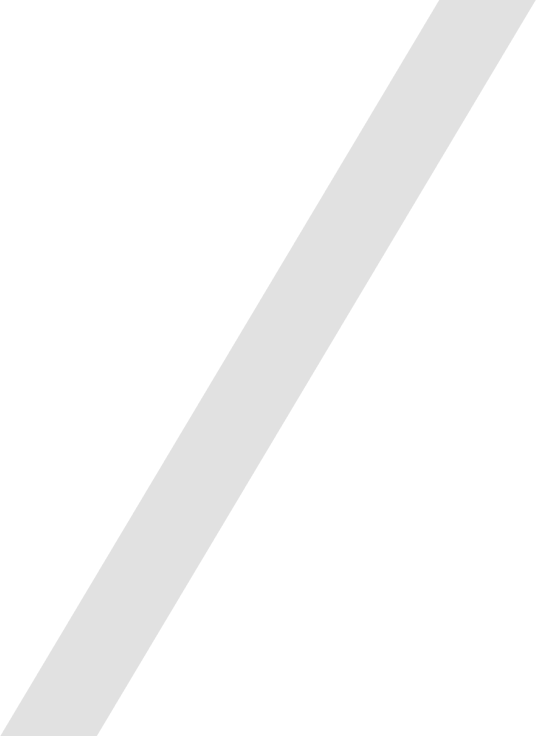
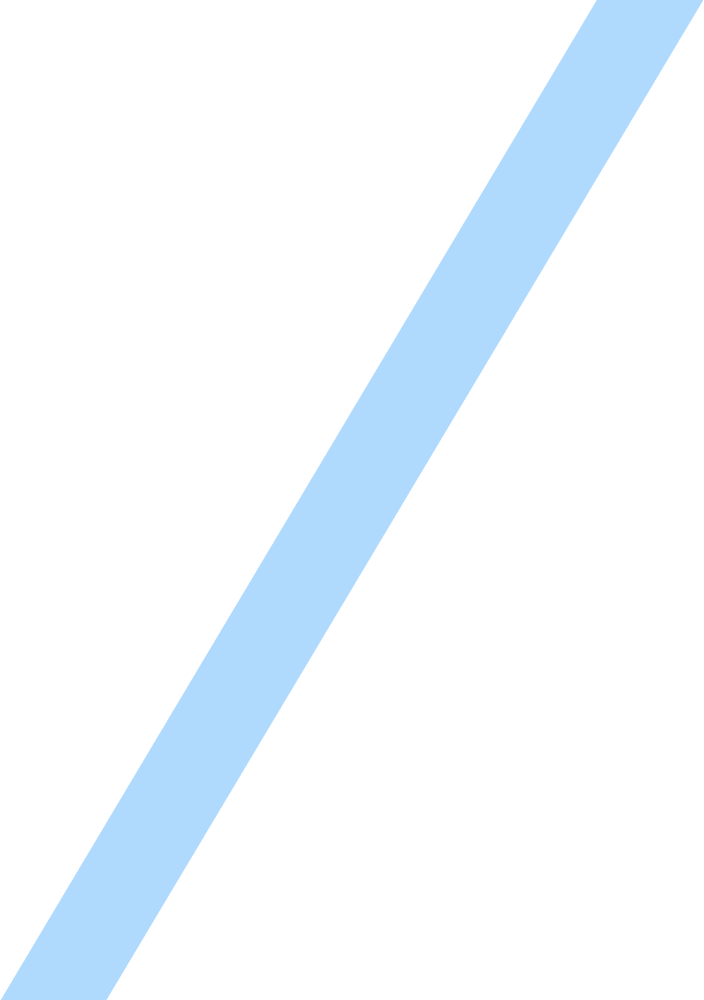


Table des matières

[I. Etat des lieux (AS IS) 2](#_Toc27814486)

[1.1 Rappel des Besoins 2](#_Toc27814487)

[1.2. Rappel des Contraintes 3](#_Toc27814488)

[1.3. Roadmap 4](#_Toc27814489)

[II. Complétion 5](#_Toc27814490)

[2.1. Fonctionnalités opérantes 5](#_Toc27814491)

[2.2. Problématiques rencontrées 5](#_Toc27814492)

[2.3. Aspect Projet 6](#_Toc27814493)

[III. Glossaire 8](#_Toc27814494)

# Etat des lieux (AS IS)

## 1.1 Rappel des 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…)

## Rappel des Contraintes

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 |
| On doit pouvoir se connecter sur la Raspberry via SSH sur Wifi direct |
|  | |
| 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 |

## Roadmap

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

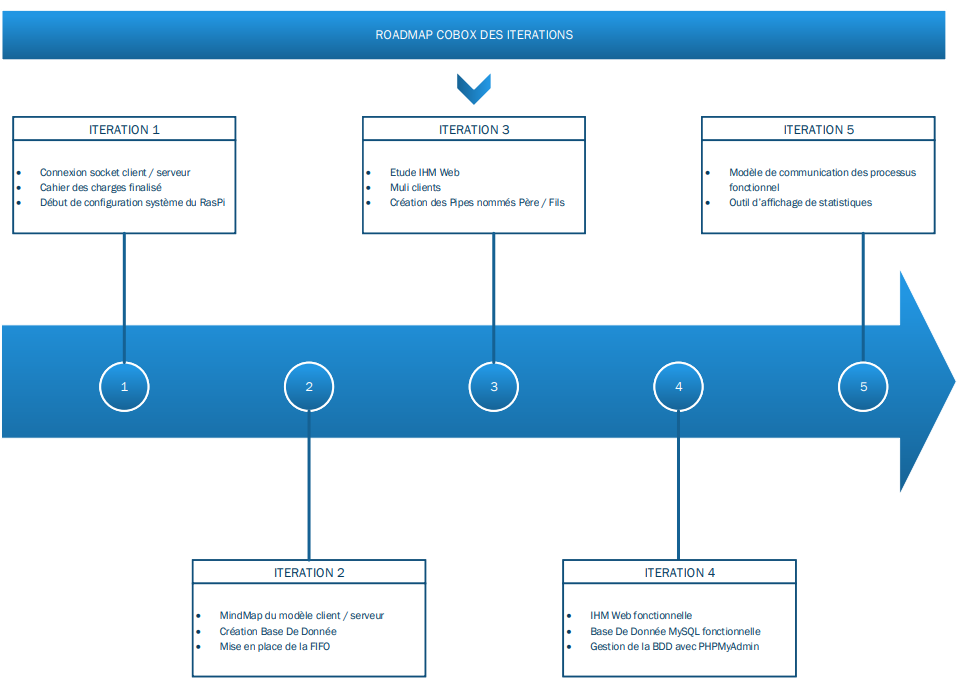


Figure 1 : Roadmap Cobox

# Complétion

## Fonctionnalités opérantes

|  |  |
| --- | --- |
| **Work Packages** | **Détails** |
| **Couche matérielle** | Raspberry 3B+, optimisable sur une Raspberry PI 2 pour diminuer les coûts |
| **Couche logiciel** | Gestion client / serveur, interface vers BDD, interface vers actionneurs puis relais |
| **Base De Données** | Archivage des données acquises, fonction mémoire d’ordres envoyés au chauffage (allumer / éteindre), gestion par PHPMyAdmin |
| **Interface Homme Machine** | Serveur Web affichant les données remontées sous forme de tableau |

## Problématiques rencontrées

|  |  |
| --- | --- |
| **Fonctions** | **Détails** |
| **Gestion de plusieurs chauffages** | Fonction trop complexe à mettre en place dès la première construction, planifiée dans la prochaine mise à jour |
| **Base de Données** | Suppression accidentelle de la machine virtuelle de test suite à une expansion de l’espace disque |

## 2.3. Aspect Projet

Depuis le départ, nous utilisons pour nous coordonner la plateforme collaborative RedMine pour assurer le suivi du projet.

Une image contenant capture d’écran, intérieur

Description générée automatiquement

Figure 2 : Interface RedMine de gestion

Cette plateforme permet de savoir très rapidement le travail à effectuer, avec un indice de priorité par tâches.

De plus, à chaque mise à jour, une notification par mail est envoyée au responsable du sujet.

Une image contenant capture d’écran

Description générée automatiquement

Figure 3 : Gantt de l'ensemble du projet

Une image contenant capture d’écran

Description générée automatiquement

Figure 4 : Vue liste des tâches par collaborateur

Une image contenant texte, intérieur, carte, mur

Description générée automatiquement

Figure 5: Planning du projet

Une autre visualisation possible est le planning, avec accès direct aux informations importantes en un simple clic.

## Fonctionnement du programme

Ci-dessous, nous pouvons voir le programme lancé. Toutes les 5 secondes, une nouvelle connexion se lance et transmet les données via l’encapsulation réseau. Elle se ferme par la suite par soucis d’économie de bande passante et de ressources matérielles.

Les données acquises sont ensuite insérées dans la base de données après authentification sécurisée sur celle-ci.

Une image contenant capture d’écran, texte

Description générée automatiquement

Figure 6: exécution du programme dans la console

## Aspect Sécurité & Qualité

|  |  |
| --- | --- |
| **Méthodes** | **Détails** |
| Authentification Wi-Fi | La méthode de connexion sur la carte Raspberry est une authentification forte WPA2 PSK |
| Contrôle d’accès | Un Fail2Ban gère les adresses IP pour l’administration sécurisée (blocage au bout de 3 tentatives) |
| Firewall Iptables | Gestion sécurisée des ports réseau et du masquage |
| Chiffrement | Utilisation du protocole de connexion SSH, Chiffrement de la base de données, implémentation de la suite HTTPS sur l’interface Web (certificat racine, TLS 1.3, HTTP 1.1), hachage des mots de passe |
| Gestion de l’obsolescence | Désactivation des protocoles obsolètes |
| Pentest | Un Pentest a été réalisé avec les méthodes blackbox et whitebox par un organisme externe |
| Stabilité du logiciel | Gestion utilisateur, test opératoire et aléatoire |
| Endurance | Nous avons laissé tourner l’infrastructure pendant 1 mois |

# Glossaire

* GPIO : General Purpose Input / Output
* BDD : Base De Données
* IHM : Interface Homme Machine
* Wi-Fi : Wireless Fidelity
* SSH : Secure Shell
* HTTP (S) : Hyper Text Transfer Protocol (Secure)
* TLS : Transport Layer Security
* WPA : Wi-Fi Protected Access