Rapport Projet Fas : Domotique et sécurité

I. Objectifs

A. Objectif de départ

L'idée initiale du projet était d'élaborer un système de surveillance pour le domicile de particuliers visant à détecter une intrusion lors de l'absence du propriétaire. Pour cela, nous voulions créer une alerte lors d'une intrusion, qu'il était possible de désactiver via à un code numérique émis à l'oral, reconnu ensuite grâce à un logiciel de reconnaissance vocale installé sur le Raspberry. Aussi, nous voulions prévenir le propriétaire par SMS et email lors de chaque intrusion.

B. Objectif rendu

Finalement, nous avons réussi à développer un système de surveillance capable de détecter un mouvement dans le domicile du propriétaire, d'en avertir ce dernier par SMS et email, et de déverrouiller l'alarme par un code donné via un pavé numérique (ici clavier classique).

Malgré le fait que nous ayons réussi à créer un dictionnaire reconnaissable par le logiciel de reconnaissance, nous ne sommes pas parvenu à l'appliquer à notre projet.

Nous avons aussi mis en ligne un site web lié au Raspberry qui informe le propriétaire de chaque étape du processus : à savoir l'état de l'alarme en temps réel, ainsi que le statut du dernier incident (en cours, code erroné, résolu).

De plus, nous avons ajouté une option disponible sur le site qui permet d'obtenir la température effective du domicile, celle-ci s'actualisant à chaque connexion à l'espace personnel.

II. Mode d'emploi

Tout d'abord, il est nécessaire de brancher les différents capteurs et la clé GSM sur les ports indiqués du Raspberry, de connecter ce dernier à Internet et de l'alimenter. Il faut placer tout le système près de l'entrée du domicile afin de repérer toute intrusion (sachant que la portée maximum du capteur est de 3 mètres).
L'utilisateur lance ensuite le programme de surveillance sur le Raspberry à son départ du domicile, il bénéficie de quelques secondes pour sortir du logement sans que l'alarme ne se déclenche. Ainsi, le programme tourne en boucle jusqu'à ce qu'un mouvement significatif soit détecté (intrusion ou propriétaire qui rentre).
Le suivi des incidents est disponible sur l'espace personnel du propriétaire grâce à un écran de contrôle. Un incident est créé sur le site web et prend alors le statut "en cours". Un SMS et un email sont alors envoyés au propriétaire pour l'informer de l'incident.

On demande ensuite au visiteur d'entrer un code, il dispose de 3 tentatives pour entrer le code juste. Si celui ci est faux au bout des 3 essais, le propriétaire est averti par SMS et email, et l'état de l'incident passe au statut "échec code". Dans le cas où le code entré est bon, le statut en ligne indique à l'utilisateur que l'incident est clos. Dans les 2 cas, l'alarme est coupée.

Pour obtenir la température en temps réel de son domicile, l'utilisateur doit simplement se connecter à son espace personnel. La température sera automatiquement récupérée depuis le Raspberry et indiquée sur son écran de contrôle en ligne.

III. Moyens utilisés

Pour ce projet, nous avons eu besoin d'un Raspberry mais aussi de plusieurs capteurs et émetteurs : un détecteur de mouvement qui va permettre de repérer tout mouvement pendant l'absence de l'utilisateur dans son domicile; un buzzer et une LED qui informent celui-ci des différentes étapes du processus; une clé GSM qui va permettre d'envoyer des SMS au propriétaire selon l'état des incidents; un capteur de température qui affichera la température du domicile en temps réel sur l'espace personnel (et un microphone pour la reconnaissance vocale qui n'a pas aboutie). Tous ces capteurs sont branchés au Shield (GrovePi), lui-même relié directement au Raspberry.

Concernant la partie logicielle, nous avons utilisé le système Gammu pour l'envoi des sms depuis le Raspberry; l'interface cURL en ligne de commande pour effectuer des requêtes http depuis un script; un hébergeur web pour l'espace personnel; le serveur Apache qui permet d'exécuter des fichiers en perl à distance pour demander la température depuis le serveur web (et le système PocketSphinx pour la reconnaissance vocale).

IV. Moyen humains

James a géré l'utilisation de l'alarme, avec les controllers correspondant sur le serveur web et le fichier exécutable sur le Raspberry.
Ilias a géré l'utilisation de la température, avec les controllers correspondant sur le serveur web et les fichiers exécutables sur le Raspberry.
Toute la mise en place des logiciels, les idées d'améliorations et les recherches nécessaires au bon fonctionnement du système ont été effectuées en commun.

V. Architecture

Le site web a été hébergé chez OVH. Il a été développé en html/css pour le visuel des différentes pages web, php pour les fonctions de vérifications et les appels à la base de données, mysql pour le stockage de la base de données. Afin d'offrir à l'utilisateur un espace personnel intuitif et agréable, nous avons utilisé le framework materialize.
Pour coder le système d'alarme, nous avons utilisé du C, ainsi que des commandes shell. Pour la température, nous avons aussi eu besoin de perl.

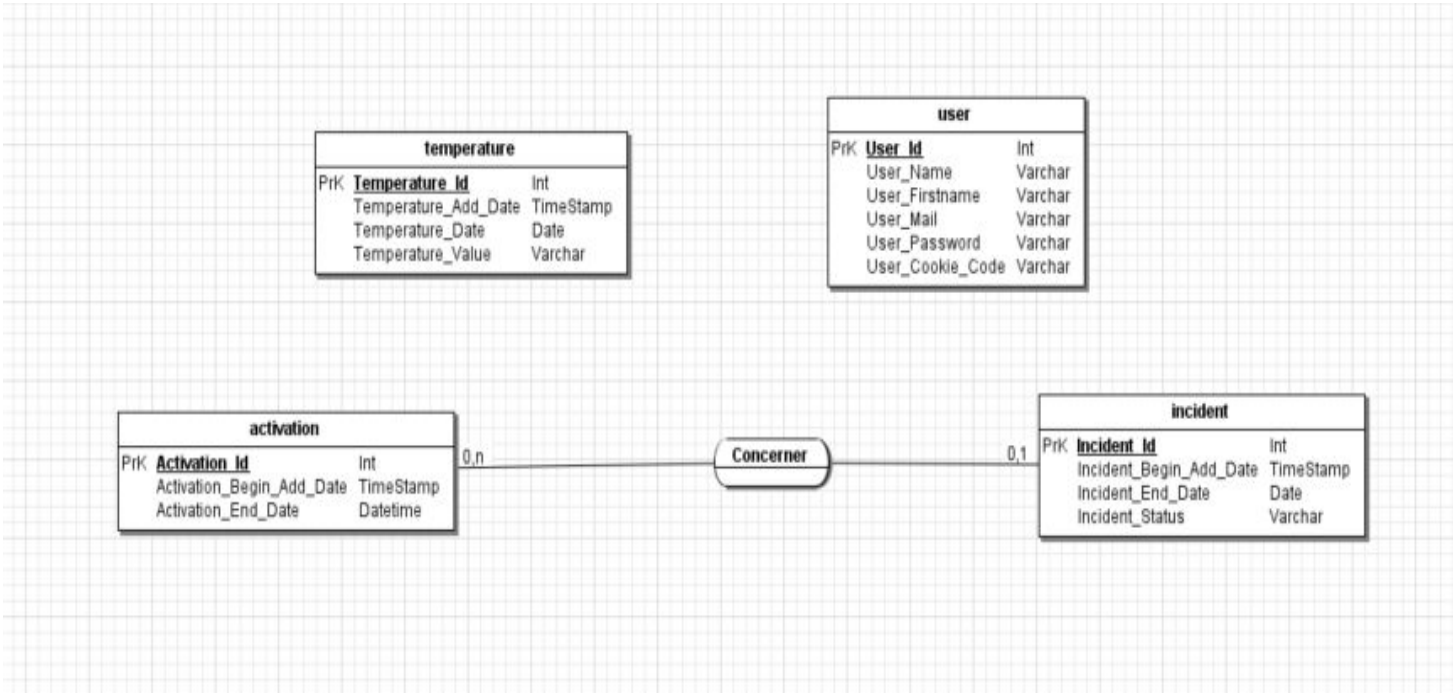
VI. Code

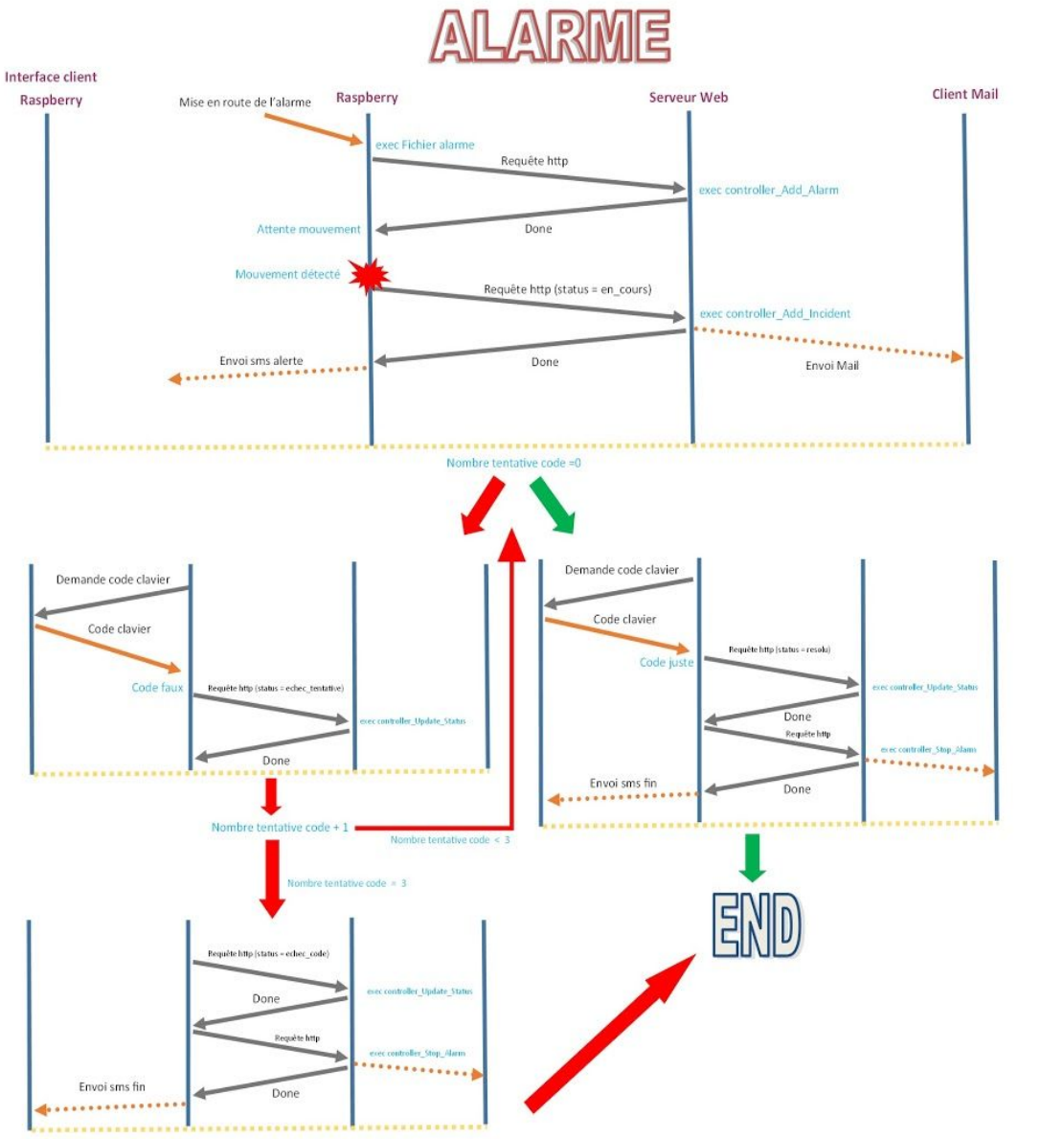
Pour l'alarme :

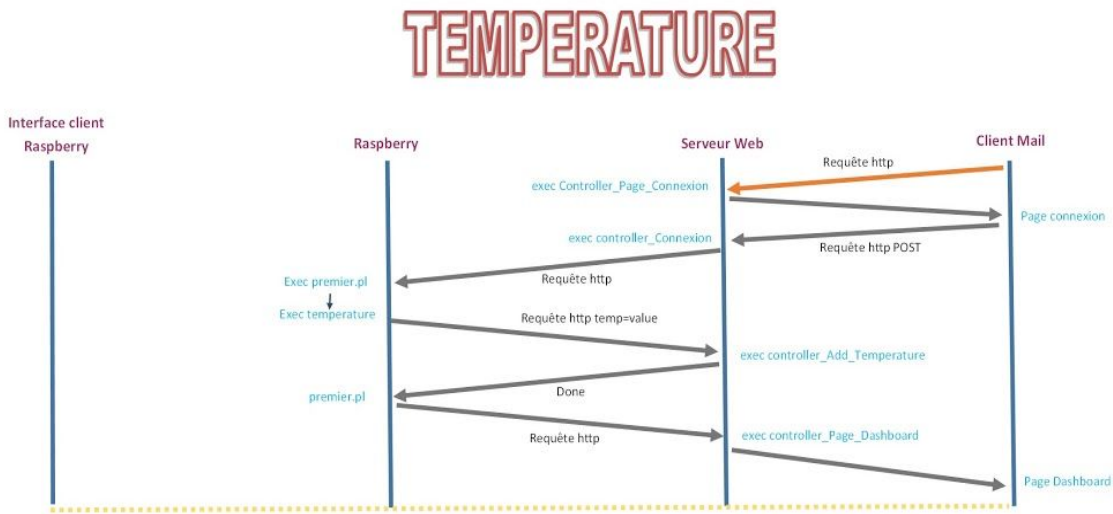
Nous avons un seul programme principal (alarme.c qui est compilé en alarme). Ce programme a été créé à l'aide des fonctions corrigées "ultrason.c" et "grovepi.c", mises à disposition sur Moodle. Nous avons intégré la gestion des 3 capteurs utilisés : le détecteur de mouvement à infrarouge (qui renvoie 1 si il y a un mouvement et 0 sinon), la LED et le buzzer. Dans ce programme, nous avons aussi ajouté les commandes shell (exécutées par la commande system() en c), afin d'envoyer des SMS et de générer des requêtes http. Les requêtes http appellent des contrôleurs hébergés sur le serveur, permettant l'ajout des notifications (incident, statuts et alarme). Il nous suffit donc de compiler ce code, puis d'exécuter le fichier produit pour lancer le programme.

Pour la température :

Lorsque l'utilisateur se connecte au site web (jamesterrien.fr), le contrôleur connexion est appelé pour vérifier les identifiants saisis. Si ceux-ci sont justes, le contrôleur envoie une requête http au Raspberry pour exécuter le fichier premier.pl. Cette requête est possible grâce au serveur Apache installé sur le Raspberry. Le fichier premier.pl exécute tout d'abord le fichier compilé temperature, qui récupère la valeur du capteur de température branché au Raspberry en temps réel. Ensuite le fichier perl affiche une vue à l'utilisateur avec un lien cliquable "continuer", qui génère une requête en GET contenant la température vers le contrôleur d'ajout de température. Une fois que celle-ci est ajoutée à la base de données, le contrôleur renvoie le dashboard à l'utilisateur. Celui-ci est actualisé avec la dernière température.




ALARME
Interface client
Raspberry
Mise en route de l'alarme
Raspberry
Serveur Web
Client Mail
exec Fichier alarme
Requête http
exec controller_Add_Alarm
Attente mouvement
Done
Mouvement détecté
Requête http (status = en_cours)
exec controller_Add_Incident
Envoi sms alerte
Done
Envoi Mail
Nombre tentative code =0
Demande code clavier
Code clavier
Code faux
Requête http (status = echec_tentative)
exec controller_Update_Status
Done
Nombre tentative code + 1
Nombre tentative code < 3
Nombre tentative code = 3
Demande code clavier
Code clavier
Code juste
Requête http (status = resolu)
exec controller_Update_Status
Done
Requête http
exec controller_Stop_Alarm
Envoi sms fin
Done
END
Requête http (status = echec_code)
exec controller_Update_Status
Done
Requête http
exec controller_Stop_Alarm
Envoi sms fin
Done


TEMPERATURE
Interface client
Raspberry
Raspberry
Serveur Web
Client Mail
Requête http
exec Controller_Page_Connexion
Page connexion
exec controller_Connexion
Requête http POST
Exec premier.pl
Requête http
Exec temperature
Requête http temp=value
exec controller_Add_Temperature
premier.pl
Done
Requête http
exec controller_Page_Dashboard
Page Dashboard

VII. Perspectives

Nous estimons que notre projet peut être considéré comme abouti car il fonctionne et accompli les objectifs essentiels que nous nous étions fixés en terme de surveillance et d'information à l'utilisateur.

Toutefois, nous souhaiterions apporter quelques améliorations :

- la sécurité : rendre notre système plus fiable et le verrouiller, notamment toutes les requêtes que nous effectuons entre le Raspberry et le serveur web
- la reconnaissance vocale : remplacer la saisie du code par un clavier par une reconnaissance vocale
- le dashboard : ajouter des données, concernant l'évolution de la température, et des fonctionnalités telles que l'activation/désactivation de l'alarme à distance
- le design : l'aspect du Raspberry et des capteurs pourrait être travaillé pour inclure tout le système dans une boîte compacte et élégante en faisant bien attention de laisser le champs libre au détecteur de mouvement et à la visibilité de la LED

Du fait que nous ayons investi dans notre propre matériel, cette liste d'améliorations correspond à l'idée de l'achèvement complet que nous nous faisions du projet. Nous espérons trouver d'autres fonctionnalités pour notre station de domotique connectée.

VIII. Sources

Général :

- http://stackoverflow.com/
- https://help.ubuntu.com/
- https://openclassrooms.com/forum
- http://www.commentcamarche.net/
- https://www.raspberrypi.org/forums/
- http://www.w3schools.com/

Pour l'envoi des SMS :

- https://www.supinfo.com/articles/single/2892-mise-place-raspisms
- https://arno0x0x.wordpress.com/2015/05/02/envoyer-sms-avec-rpi/

Pour l'interaction du serveur vers le Raspberry :

- https://openclassrooms.com/courses/ecrivez-votre-site-web-en-c-avec-la-cgi
- http://perldoc.perl.org/CGI.html
- https://doc.ubuntu-fr.org/installer_un_serveur_debian
- http://raspbian-france.fr/installer-serveur-web-raspberry/

Pour le cURL :

- https://curl.haxx.se/docs/httpscripting.html
- http://www.zem.fr/curl-15-commandes-pratiques-avec-curl/

Pour l'utilisation des capteurs :

- https://www.dexterindustries.com/
- https://www.seeedstudio.com/
- http://www.lextronic.fr/

Pour la reconnaissance vocale :

- https://wolfpaulus.com/embedded/raspberrypi2-sr/
- https://pypi.python.org/pypi/SpeechRecognition/
- https://www.raspberrypi.org/forums/viewtopic.php?t=25173
- http://cmusphinx.sourceforge.net/wiki/tutorialpocketsphinx
- http://cmusphinx.sourceforge.net/wiki/tutorialdict