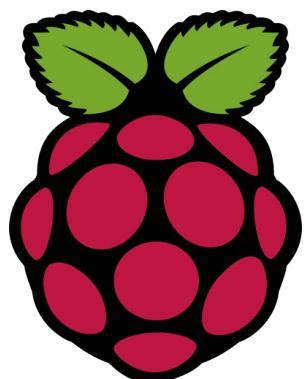
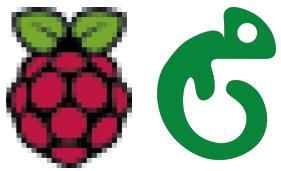




Rapport de CESI-PRF2019 projet 1 RPI  
mené par le Groupe 2,  
porté par Yahya Mazouar,  
Thierry Li et Carole Hannane





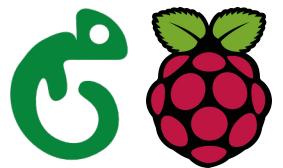
## Présentation

Dans le cadre de notre formation au sein du CESI Bordeaux , nous avons été missionnés pour travailler sur un projet utilisant un appareil Raspberry Pi 3 et des capteurs de température.

Concernant la programmation, il nous a été demandé d'utiliser le langage Python.

Nous avons pour objectif de rendre le livrable sous 1 mois dont 3 jours avec la présence de notre chef de projet, M. Éric Artigala.

Pour se réaliser, ce projet a été amené et porté par l'équipe de développement constitué de Yahya Mazouar, Thierry Li et Carole Hannane.

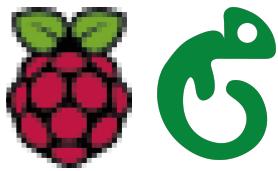


# Table des matières

## Présentation

## Table des matières

<b>Introduction du projet</b>	<b>1</b>
1. Contexte du projet .....	1
2. Reformulation du besoin du client et des solutions proposées .....	1
3. Exigences liées à la demande .....	2
4. Les contraintes et les limites du système de contrôle.....	2
<b>BACKLOG</b>	<b>3</b>
<b>Mise en rappel du matériel utilisé : le Raspberry PI3 et les capteurs</b>	<b>4</b>
<b>Méthodologie</b>	<b>5</b>
1. Définition des tâches et actions à accomplir avec Kanban .....	5
2. Ciblage des fonctionnalités primaires du programme .....	7
<b>Environnement technique</b>	<b>8</b>
1. Définition du langage utilisé : Python .....	8
2. Installation de l'environnement de travail .....	8
3. Notre première fonction, écouter .....	9
4. Le filtrage des données.....	9
5. L'envoi de mail .....	9
<b>Récolte et sauvegarde des données</b>	<b>10</b>
1. Gestion des données récoltées .....	10
2. Export et sauvegarde des données dans un fichier texte .....	10
<b>Notre retour d'expérience</b>	<b>11</b>
<b>Axe de développement</b>	<b>13</b>
<b>Conclusion</b>	<b>14</b>
Annexe 1 .....	15
Annexe 1 (suite) .....	16
Annexe 2 .....	17
Annexe 3 .....	18
Journal de bord .....	19
Glossaire.....	21



## Introduction du projet

### 1. Contexte du projet

La marque VIVALAND est représentée par M. Martin, responsable des animaleries de la chaîne de magasins VIVALAND localisées en France. Le représentant nous a contactés pour la réalisation d'une prestation de contrôle, visant à la surveillance de vivariums implantés dans ses points de vente.

Sa problématique est la suivante :

La société VIVALAND rencontre une hausse de mortalité de certains animaux suite à un problème de gestion du contrôle de la température des espaces de vie de ces derniers, logés dans des vivariums.

Les conditions optimales de vie des animaux dans ces espaces tiennent compte de la température à l'intérieur des vivariums, qui doit se situer obligatoirement entre 24°C et 30°C.

En dehors de cette plage, le taux de mortalité peut augmenter de 30 à 50% selon les espèces d'animaux.

Un nouvel outil de gestion de température assistée pourrait prévenir cette mortalité et la garder sous un meilleur contrôle.

M. MARTIN souhaite :

- ▶ La mise en place d'un système de contrôle de la température des vivariums, par le biais de capteurs fonctionnant par Bluetooth qui seront intégrés dans ces espaces de vie et gérés par un système contrôlé via un Raspberry PI.
- ▶ La mise en place d'un système d'alerte par mail lorsque les seuils minimums et maximums sont atteints ainsi que pour le rétablissement à la normale de la situation.
- ▶ La création d'une base de données constituée d'informations collectées chaque heure et stockées sur 3 mois.

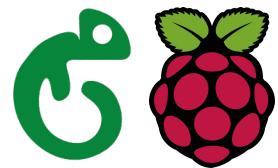
### 2. Reformulation du besoin du client et solutions proposées

Nous avons fait un point sur les demandes de M. Martin et lui avons proposé des solutions en conséquence.

Son besoin est donc la mise en place d'un système de gestion du contrôle de la température des vivariums grâce à l'utilisation de capteurs Bluetooth intégrés à ces mêmes vivariums et reliés à un Raspberry PI.

Nous mettrons en place un programme d'analyse des conditions de vie des animaux vivant dans les vivariums, soient la température à l'intérieur des espaces et le niveau de batterie des capteurs.

# Introduction du projet



Ce dernier point permettra de garantir la constante transmission des données sachant que l'arrêt du fonctionnement des appareils pourrait être gravement dommageable et aura obligatoirement une répercussion sur la survie des animaux.

La mission de ces capteurs sera d'alerter le responsable du magasin concerné, par le traitement des informations récoltées par ces capteurs et qui permettra l'envoi d'un mail automatique, dès que les seuils de température minimums et maximums seront atteints.

Une fois l'anomalie rectifiée et la situation redevenue normale, un mail de rétablissement sera envoyé à ce responsable pour l'informer.

Il souhaite collecter chaque heure les données récupérées et les sauvegarder sur une durée de 3 mois. Nous assurerons donc un programme de récupération des données relatives aux capteurs qui seront disponibles pendant le laps de temps demandé par M. Martin. Ces outils de gestion seront ensuite déployés dans tous les magasins de la marque.

## 3. Exigences liées à la demande

M. Martin a insisté sur la présence obligatoire de certaines informations :

- ▶ Le mail envoyé doit contenir : un timestamp (heure, date), les identifiants liés aux capteurs, le nom de l'objet, la température constatée et la température de seuil.
- ▶ Nous devrons utiliser un Raspberry PI et des capteurs de température Bluetooth fournis par la société VIVALAND

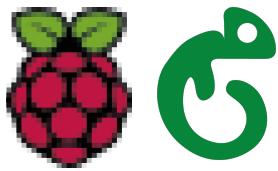
## 4. Les contraintes et les limites du système de contrôle

Une seule notification par mail doit être envoyée lors d'un dépassement de seuil. Cette limitation assurera le bon traitement de l'information reçue sans toutefois saturer la boîte de réception du responsable de la sécurité du magasin.

Il devra en être de même pour signaler le rétablissement des bonnes conditions de vie des animaux, soit un retour à la normale de la situation. Le Raspberry doit rester alimenté en électricité et doit disposer d'une connexion constante à Internet.

La portée du système Bluetooth est limitée à 40-60 mètres.

Le Raspberry doit automatiquement reconnaître les capteurs enregistrés qui l'entourent ainsi que ceux qui pourront éventuellement être rajoutés ultérieurement dans l'optique d'une intégration possible d'autres vivariums



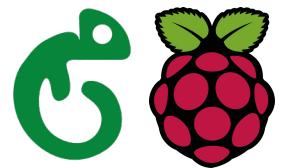
# Introduction du projet

Nous allons procéder de la façon suivante, de manière décomposée :

- ▶ Installation Raspbian et Raspberry Pi
- ▶ Scan des capteurs enregistrés
- ▶ Décrypter les données qui nous intéressent, les stocker pour une réutilisation.
- ▶ Création d'un script pour l'envoi des mails.
- ▶ Création d'un algorithme qui détecte les dépassements des seuils de température.
- ▶ Test et Debug
- ▶ Création d'une documentation et rapport de suivi du projet
- ▶ Déploiement et livraison

## BACKLOG

1. Raspberry Pi 3
  - A. Matériel
  - B. Installation du matériel
  - C. Installation de l'environnement du Système d'exploitation RASPBIAN Stretch
  - D. Installation de Python 3.5
  - E. Installation des librairies Python nécessaires au programme
  - F. Test de l'environnement avec un print('hello world')
2. Capteurs
  - A. Crédit d'un script qui va écouter les puces Bluetooth
  - B. Repérage des adresses MAC des capteurs données
  - C. Crédit du script qui ciblera seulement les capteurs utilisés
  - D. Décrypter les données que nous recevrons des capteurs et les convertir
  - E. Crédit d'un algorithme qui va délimiter les seuils de températures des capteurs
3. Mail
  - A. Crédit d'un compte e-mail de test
  - B. Crédit d'un script d'envoi de mail qui envoie les données demandées
  - C. Test fonctionnel du mail
  - D. Relier le script du mail avec le script des capteurs qui va s'exécuter si la température de l'un des vivariums sort du seuil défini.
  - E. Ajout de la date et de l'heure dans le mail.
  - F. Ajout du nom du capteur
4. Stockage des données
  - A. Crédit d'un algorithme qui va enregistrer les valeurs des capteurs renvoyées dans un fichier consultable.
5. Test et Debug
6. Livrable client



## Mise en rappel du matériel utilisé : le Raspberry PI3 et les capteurs

Le Raspberry PI3 (RPI) est un nano-ordinateur assemblé sur un processeur se rapprochant de la taille d'une carte de crédit.

Il peut être branché à un écran et utilisé comme un ordinateur standard. Plusieurs systèmes d'exploitation peuvent être installés dessus, dans notre cas, il sera sur le système Raspbian Stretch.

Bien que son prix de vente, attractif, se situe aux alentours de 30€ pour un Raspberry PI3 nu (composé d'une carte mère seule, sans boîtier, alimentation clavier, souris ni écran), c'est un outil efficace, s'avérant suffisamment puissant pour délivrer une variété d'utilisations et d'applications selon les besoins.

Il est également équipé d'une puce Wifi et Bluetooth.

Le Bluetooth est une norme de communication permettant l'échange de



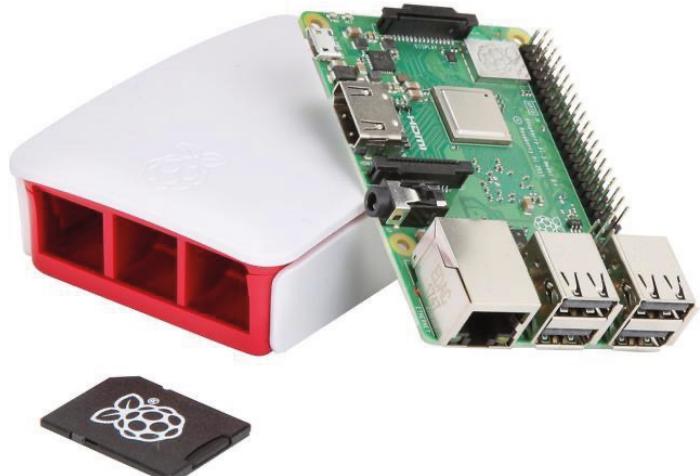
Raspberry PI3 nu

données à courte distance en utilisant des ondes radio basées sur une bande de fréquence de 2,4 GHz.

Son utilisation permet une connexion aisée entre plusieurs appareils en supprimant les liaisons filaires, tout en garantissant une faible consommation d'énergie.

Le RPI sera utilisé en lien avec les capteurs de chez Digital Technology TZONE.

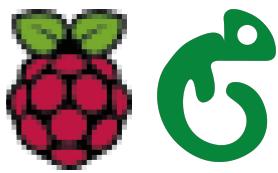
Ces derniers permettent de capturer une plage de température comprise entre -25°C à +60°C, une plage d'humidité de 0 à 100%, le tout fonctionnant avec une autonomie de 1,5 ans et avec une pile remplaçable.



Raspberry PI3 avec kit boîtier



Capteur de température de TZONE



## Méthodologie

### 1. Définition des tâches et actions à accomplir avec Kanban

Nous sommes une jeune équipe de développement constituée de 3 personnes, Yahya Mazouar, Thierry Li et Carole Hannane, soutenus par notre chef de projet, M. Éric Artigala.

En collaboration avec M. Artigala, nous avons pris connaissance du projet et avons établi les besoins de notre client M. Martin. Nous les avons traduits et découpés en actions permettant de créer un axe de travail cohérent.

Ces actions ont été détaillées à travers la plateforme de gestion de projet TRELLO, inspiré de la méthode Kanban de Toyota.

Avec la mise en place d'une liste de tâches, chaque étape sera suivie plus efficacement et permettra de mieux visualiser l'avancée du projet.

Au démarrage d'une nouvelle session de travail, nous faisons un « Daily meeting », autrement dit une réunion de lancement de journée, en répondant à 3 questions qui sont les suivantes :

- ▶ Qu'ai-je fait hier ?
- ▶ Que vais-je faire aujourd'hui ?
- ▶ Quels sont les problèmes que je rencontre ?

Ces questions sont posées au sens personnel mais elles peuvent être traduites par :

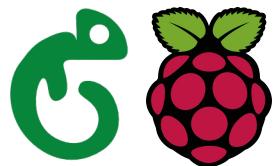
- ▶ Qu'ai-je fait hier qui a aidé l'équipe de développement à atteindre l'objectif fixé ?

La durée du "Daily" est définie par notre groupe à 10-15 minutes. Nous nous repen-chons sur notre tableau Kanban sur Trello et révisons les actions de manière transpa-rente.

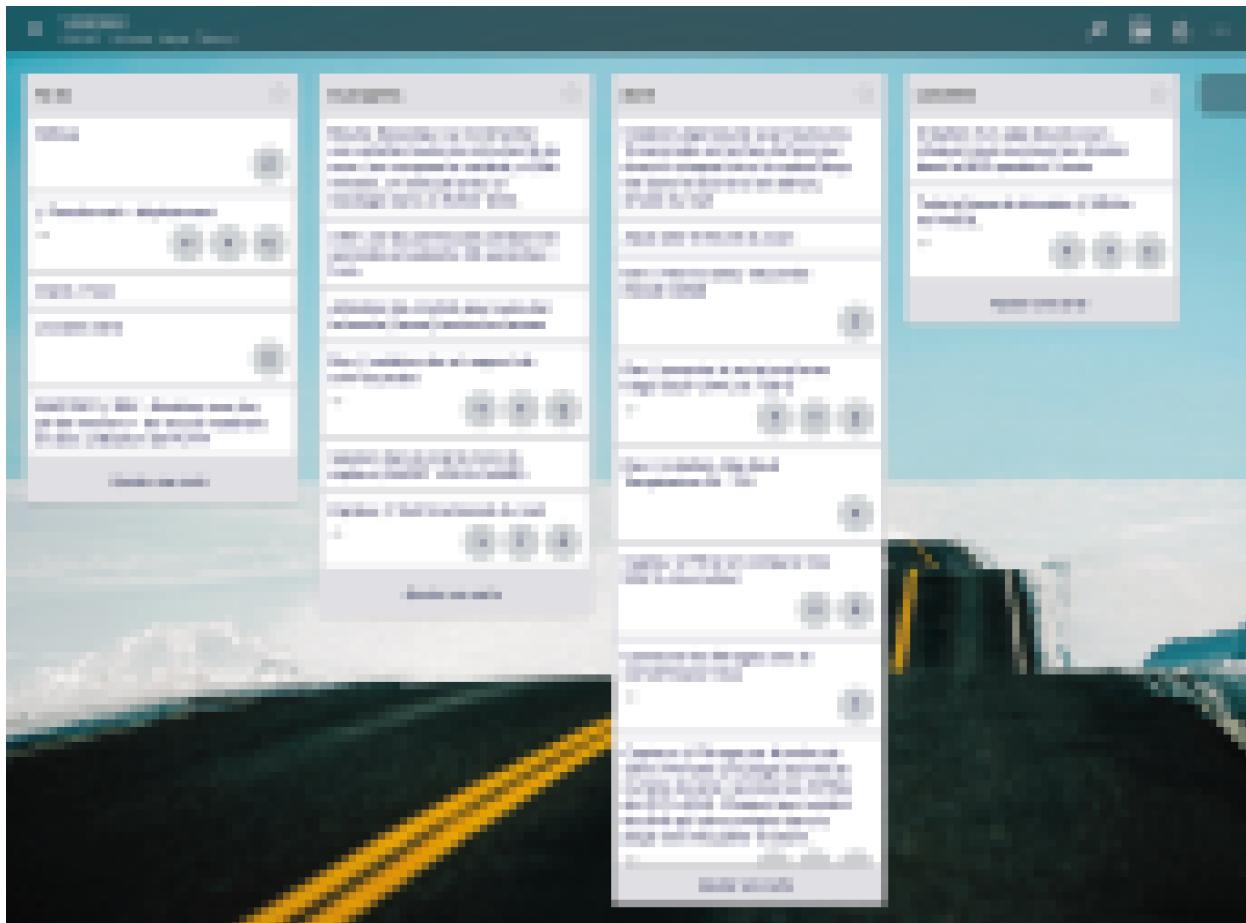
C'est une méthode qui permet de recentrer les attentions sur des objectifs atteignables sans perdre le fil du projet car nous assignons à chaque membre de l'équipe, une ou des tâches pour la journée. Notre chef de projet, M. Artigala, nous aide à la bonne réalisation de la réunion.

Nous organisons ainsi les jours d'activité et en accord avec chacun, nous prévoyons une dernière réunion de fin de journée afin de débriefer sur ce qui a été produit et où chacun argumente sur les tâches qu'il a effectuées. Ce point nous permet de mieux cibler et analyser les difficultés et de réajuster les objectifs si besoin est.

# Méthodologie



## Trello - Vues du projet



X VIVARIUM  
PROJET 1 (Carole, Yatya, Thierry)

To do

- Debug
- // Fonctionnel = déploiement
- Client // Test
- Livrable client
- RAPPORT // REX : librairies avec les pb de versions + les soucis matériels et donc utilisation de PUTTY

In progress

Bou une scan mini stock

créé sec 1min

util la b

Dev suiv

Cap

**rajouter dans le mail le nom du capteur (SHORT LOCAL NAME)**

Dans la liste : In progress

Cliquer pour ajouter une description

Date limite...

Étiquettes...

Membres...

Checklist...

Ajouter des pièces jointes...

Checklist

Récupérer le adtype-8 et sa valeur en plus de la valeur du adtype-22

Stocker la valeur adtype-8

La rajouter dans le message du mail

Ajouter une tâche...

Activité

Commentaire...

plage sera récupérer et storer.

Ajouter une carte

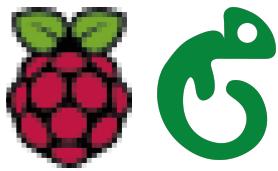
done

canceled

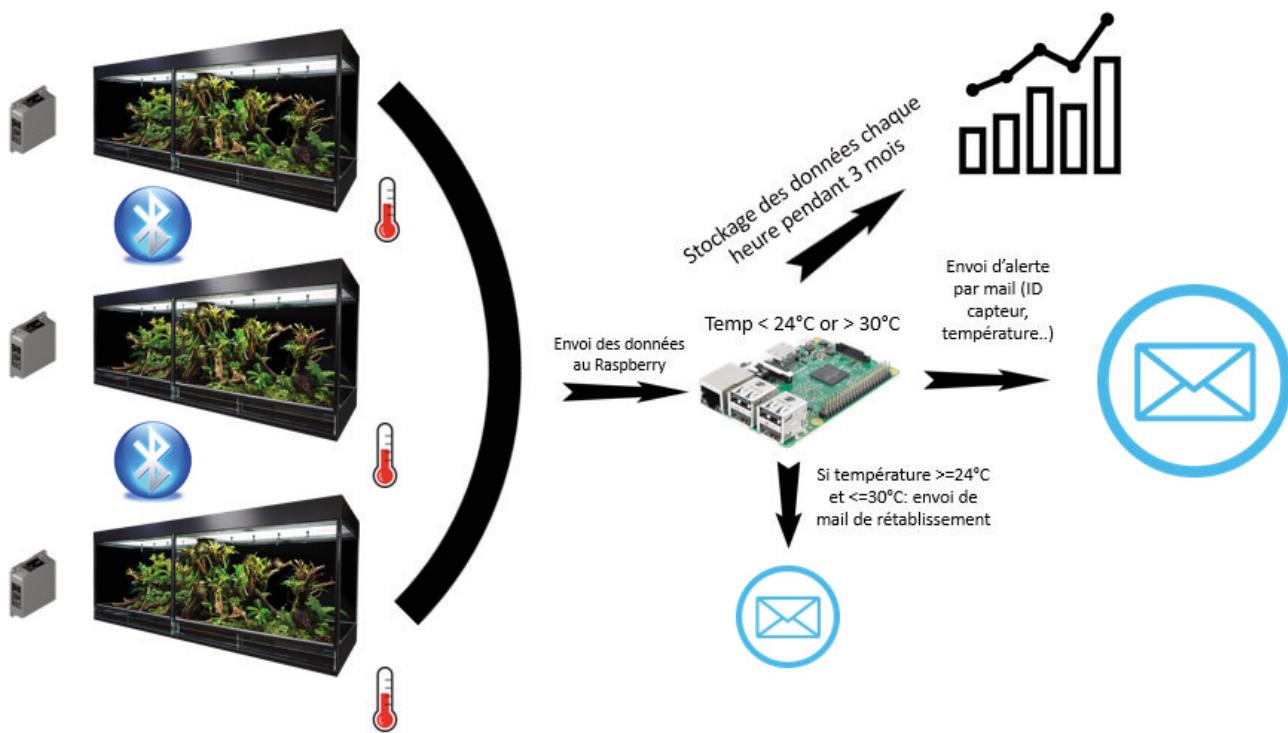
Création d'un algo boucle scan chaque heure qui pour les stocker dans la BDD pendant 3 mois

Tutoriel base de données // SQLite ou NoSQL

Ajouter une carte



## 2. Ciblage des fonctionnalités primaires du programme



Après nos échanges en interne, nous avons pu déterminer les étapes primordiales liées au bon fonctionnement du futur programme. Ce dernier sera composé de :

► La fonction d'écoute des capteurs

Dans cette fonction, le scan écoute les appareils se trouvant autour de lui et récupère toutes les données des capteurs (température, humidité, batterie, ID, adresses MAC)

► La fonction de filtrage des données :

Cette fonction permet de filtrer toutes les données qui nous intéressent (informations citées ci-dessus).

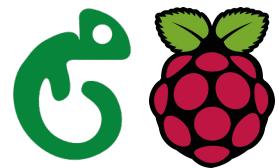
► La fonction d'envoi de mail :

Cette fonction permet d'envoyer un mail d'alerte une fois lancée en action.

► L'algorithme du seuil des températures :

Cet algorithme vérifiera si les températures récupérées correspondent bien à la norme imposée par le client, dans le cas où le seuil est dépassé, une alerte sera envoyée par mail.

Si les températures reviennent à la normale (et respectent le seuil) alors un mail de rétablissement sera envoyé.



## Environnement technique

### 1. Définition du langage utilisé : Python

Dans le cadre du projet, le langage Python sera utilisé. C'est un langage de programmation orienté objet, multi-paradigmes et multi-plateformes. Il favorise une programmation structurée et fonctionnelle.

Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion des exceptions.

C'est également un langage placé sous licence libre, sans frais et fonctionnant sur la plupart des plateformes informatiques. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser.

Cette syntaxe claire et simplifiée fait de ce langage un outil parfait pour l'initiation aux concepts de la programmation.

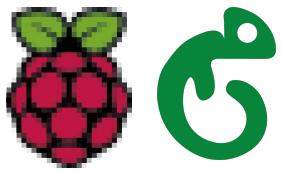
### 2. Installation de l'environnement de travail

Le Raspberry PI3 fonctionnera sous un système d'exploitation Raspbian, lui-même basé sur Debian GNU/Linux. L'installation du système se fait par la gravure d'un fichier image de l'OS dans une carte micro SD que le RPI admet en tant que disque dur. Suite à la configuration du système, nous passerons à l'installation des outils logiciels nécessaires au bon fonctionnement du développement du programme et de son utilisation.

Pour cela, nous avons utilisé la librairie Bluepy, qui est un programme open-source de Ian Harvey (documentation disponible sur internet), et qui nous a permis l'exploitation de la technologie Bluetooth. Il est disponible gratuitement sur le site Github (une plate-forme de versionning et d'échange de codes sources). Son installation se fait par terminal sous Raspbian.

Pour être exploité, il est nécessaire d'installer, au préalable, un outil de gestion des librairies, nommé « PIP » pour Python, toujours via le terminal. Nous ferons appel aux librairies dans notre fichier de programmation Python en réalisant un import et en initialisant une classe provenant de Bluepy.

Cette dernière va permettre de lancer un scan et donc une première écoute de toutes les ondes radio Bluetooth des appareils voisins.



### 3. Notre première fonction, écouter

Notre première approche est d'utiliser la classe scanner de la librairie bluepy.btle qui a été importée au début. Nous pouvons définir la durée du scan qui sera de 30 secondes.

Ce scan permet de récupérer les données suivantes :

- ▶ ID des capteurs et leurs adresses MAC
- ▶ Valeurs en hexa des températures et niveaux de batterie des capteurs
- ▶ Eventuellement, le taux d'humidité à l'intérieur des vivariums

Une fois ces données récupérées en hexa, nous les convertissons pour obtenir clairement les informations à transmettre au responsable de la sécurité et les stocker à la demande du client. Ces informations, une fois converties, seront isolées pour ne retenir que celles qui nous seront vraiment utiles.

Voir Annexe  
1 et 2

### 4. Le filtrage des données

Pour le filtrage nous avons conçu un algorithme qui trie les ID des capteurs (numéros de série), qui définit une plage qui part de 11 000 000 jusqu'à 12 000 000 et qui permettra, à chaque fois qu'il détectera un nouveau capteur, de récupérer ses données (l'adresse MAC, le raw) puis il les convertira pour les rendre compréhensibles.

L'algorithme permet d'obtenir une série de caractères qui sera découpée en petits groupes pour pouvoir être convertie de l'hexadécimal en décimal. Une fois toutes les données traitées, elles seront affichées avec une légende, comme le présente l'exemple suivant :

- ▶ Température : 19,14°C
- ▶ ID : 11162856
- ▶ Batterie : 80%

### 5. L'envoi de mail

Pour la gestion d'envoi de mail, nous avons conçu un algorithme capable de détecter si les températures restent comprises dans le seuil de sécurité ou non.

Ce seuil va de 24° à 30°C.

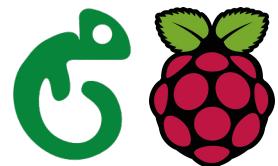
Dans le cas où la température des vivariums sortirait de ce seuil, le programme poursuivra la procédure de scan et parallèlement, enverra un mail d'alerte au

responsable de la sécurité pour l'informer de l'anomalie relevée.

Suite à cela, l'algorithme permet d'enregistrer ces informations mais continue le processus de scan.

Lorsque la température revient à la normalité, une notification de rétablissement est envoyée et le cycle est repris afin que perdure la surveillance des vivariums.

Voir Annexe 3



## Récolte et sauvegarde des données

### 1. Gestion des données récoltées

A la demande du client qui souhaite que les données des capteurs soient stockées, nous avons besoin de les conserver temporairement afin de pouvoir les réutiliser dans d'autres parties du programme.

Nous avons décidé d'intégrer des « dictionnaires », un type de liste d'informations qui permet d'être réutilisé plusieurs fois, et nous allons attribuer des clefs alphanumériques aux données dans le but de pouvoir les récupérer ultérieurement dans un travail annexe.

Cela pourra s'avérer utile pour stocker le nom de nos capteurs et leur associer des valeurs de température qui seront récupérées lors de chaque écoute Bluetooth. Nous pourrons alors les employer dynamiquement pour les envois des mails d'alerte et de rétablissement.

### 2. Export et sauvegarde des données dans un fichier texte

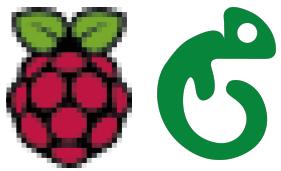
Pour répondre à la demande du client, nous devons exporter les données des capteurs chaque heure pendant 3 mois.

La question de la création d'une base de données a été évoquée en interne, étudiée et expérimentée mais n'a pas été retenue. Malgré un script opérationnel, un manque de temps et d'étude n'a pas permis un aboutissement de cet exercice.

Concernant la mise en place du fichier exporté en format ".txt", nous nous sommes orientés vers des fonctions telles que « with open » par lesquelles nous créons un fichier texte pour y enregistrer les données.

L'utilisation d'un programme supplémentaire (planificateur de tâches), appelé Crontab, dont la programmation se réalise sous Unix et Linux (donc Raspbian), permet d'exécuter automatiquement un script, des commandes, ou des logiciels à une date et une heure spécifiées à l'avance, le tout ayant un cycle défini préalablement.

C'est un choix simple, qui nous a semblé le plus évident à mettre en place pour un temps bien précis.



### Notre retour d'expérience

Tout au long du projet, nous avons rencontré les difficultés suivantes :

► Une indisponibilité de matériel :

Nous nous sommes retrouvés dans une situation où nous ne disposions pas d'écrans pour pouvoir travailler graphiquement sur le Raspberry, nous nous sommes donc rabattus sur une connexion SSH (voir glossaire) avec le logiciel PUTTY (outil de connexion à distance) afin de pouvoir exécuter et tester nos scripts.

Cette absence de matériel a effectivement eu un impact sur notre temps de conception mais ceci nous a permis d'apprendre à utiliser d'autres technologies et d'étendre nos connaissances.

► Difficultés de conception de l'algorithme général et des fonctionnalités.

Etant débutants, nos lacunes ont influencé notre capacité de conception mais nous avons su nous adapter et faire les recherches nécessaires afin de parvenir au meilleur résultat possible.

► Les librairies :

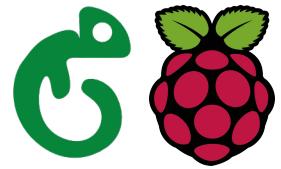
Concernant ce point, nous nous sommes aperçus que certaines d'entre elles compliquaient un peu le travail, comme la fonction Schedule ou BluePy, non exploitables sous Windows car incompatibles.

Il n'était donc pas possible d'exécuter certains scripts en dehors du cadre du CESI comme le scan ou certains algorithmes qui nécessitaient des données réelles, ceci ne nous permettant que difficilement d'expérimenter notre travail.

### “toujours s'adapter”

Nous avons effectivement rencontré quelques difficultés relatives à ce projet mais notre persévérance et notre motivation nous ont donné les moyens de réaliser ce programme du mieux possible.

Les problèmes d'ordre matériel, notamment au niveau des écrans manquants ont certes ralenti un peu l'avancée du projet mais ceci nous a permis, encore une fois, d'acquérir d'autres connaissances puisque nous avons appris à utiliser des logiciels de connexion à distance (utilisation du logiciel PuTTY et du protocole SSH).



L'avantage de la présence d'écrans facilitait nettement la gestion et la visualisation du programme mais nous avons su nous adapter soit en sollicitant l'aide de nos collègues (prêt de matériel) soit en exploitant d'autres moyens.

Nous avons également tenté de fixer une adresse IP statique sur le Raspberry PI3 afin de pallier à l'écran d'ordinateur indisponible pour choisir et sélectionner un réseau Wifi.

Cependant, notre environnement de travail ne nous a pas permis de régler ce détail lié à des problématiques de proxy et pare-feu Internet de l'établissement, la connexion WiFi étant limitée.

## “un algorithme”

Nous avons également été confrontés à des problèmes de méthodologie au sein du groupe qui nous ont ralenti, notamment sur la bonne définition de l'algorithme général et certains détails du programme.

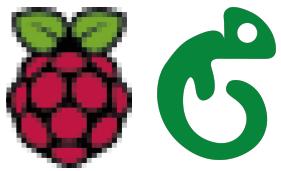
Notre algorithme présentait un manque de fonctionnalités. Au cours du projet, nous sommes revenus plusieurs fois sur nos pas pour étudier l'utilisation d'une librairie « schedule » pour la planification des écoutes Bluetooth dans le but de créer une boucle infinie qui exécuterait ladite planification.

Nous avons eu de nombreuses interrogations pour choisir comment stocker temporairement les données/valeurs que nous renvoient les capteurs et nous n'avons pu faire aboutir cette tâche.

## “Vive GitHub”

Enfin, un petit dysfonctionnement de la carte SD servant pour la sauvegarde du programme nous a poussés à effectuer quelques recherches sur les plateformes d'hébergement et de gestion de développement des logiciels, nous avons pu alors tester GitHub, un outil très pratique et très fonctionnel nous ayant fortement aidés.

En plus de pouvoir sauvegarder certaines versions de chacun des fichiers, ce service permet de faire une rétrospection du programme si besoin, voire même de partager les fichiers au sein d'un groupe facilement. Après une réinstallation du système d'exploitation Raspbian sur le RPI, nous n'avions plus qu'à cloner et télécharger notre projet à son dernier stade.



## Axe de développement

### ► Base de données

Dans un souci de respect du délai, nous nous sommes résolus à opter pour un fichier.txt pour le stockage des données.

Notre première volonté était de travailler sur une base de données relationnelle via SQLite. Il s'est avéré un peu compliqué et chronophage d'utiliser cette solution car la difficulté portait sur le fait que la base ne pouvait se générer avec un serveur mais plutôt avec des fichiers intégrés directement au programme.

Un script a pu être réalisé mais n'a pu être exploité toujours par manque de temps. Nous connaissons cependant la procédure et pourrons l'exploiter ultérieurement.

C'est une solution avec une portabilité pouvant fonctionner sur différentes plateformes et que nous espérons pouvoir étudier prochainement en ayant acquis plus d'expérience.

### ► Lancement du programme automatiquement au démarrage du RPI

Actuellement, le programme doit être exécuté manuellement en utilisant le terminal en ligne de commande. Il s'agit de commandes simples et facilement reproduisibles après une très courte formation à laquelle nous ne manquerons pas d'y recourir.

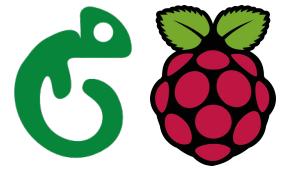
A l'avenir, une automatisation du lancement du programme pourra être configurée avec Crontab (fonctionnalité intégrée à Linux).

En définissant des heures et des jours à l'avance via Crontab, le programme pourra ainsi se lancer au démarrage du RPI.

### ► Développement du programme avec de la programmation "Objet"

La programmation orientée objet permet d'obtenir un code réutilisable et très bien structuré. De plus, cette façon de programmer offre un code plus compréhensible et plus efficace.

Cependant, dans notre cas, nous n'avions pas encore les compétences techniques pour pouvoir développer le programme comme nous l'aurions souhaité, c'est une question qui a tout de même été abordée et à laquelle nous reviendrons plus tard car il est important pour nous d'obtenir la capacité de travailler avec ce type de méthode et de pouvoir évoluer.



## Conclusion

Le projet n'a pu aboutir complètement puisque nous constatons un manque de certaines fonctionnalités, cependant, le programme se présente opérationnel malgré tout.

Bien que nous ayons rencontré diverses difficultés, nous avons pu tester notre adaptabilité et notre réactivité et ceci nous a permis d'acquérir une expérience qui nous sera parfaitement utile pour nos prochains projets et de ne pas rester bloqués sur une action.

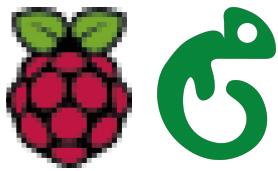
Ce projet bien qu'inachevé, nous aura permis d'en apprendre plus sur différentes méthodes de développement et notre curiosité nous aura poussés à procéder à de nombreuses recherches vraiment formatrices.

Notre souhait est de présenter à notre client, M. Martin, notre solution de la problématique qu'il nous a adressée, dans son ensemble, et avoir ses impressions par rapport à la pertinence de nos décisions vis-à-vis du programme.

Nous lui présenterons également nos axes de développement, s'il venait à juger ces axes pertinents, nous serons alors ravis de pouvoir perfectionner et achever ce projet en intégrant de nouvelles fonctionnalités dans une seconde version.

Nous vous remercions chaleureusement d'avoir pris le temps de lire ce rapport et d'avoir confié ce projet à une jeune équipe débutante qui aura pris beaucoup de plaisir à travailler sur ce programme et particulièrement reconnaissante pour les connaissances supplémentaires qu'elle aura pu acquérir.

Nous remercions M. Éric Artigala pour cette opportunité qui nous a été profitable et d'avoir supervisé ce projet.



## Annexe 1

```
# Import library pour le script acces mail et accès smtp
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
import smtplib
# Import date et heure actuelle
from datetime import datetime

# variable de test pour un numéro vivarium et faux magasin de test en local sans RASPBI
AN
vivarium = 3
animalerie = "Vivaland_Bordeaux_Merignac"
# datetime object containing current date and time
now = datetime.now()
# dd/mm/YY H:M:S
dt_string = now.strftime("%d/%m/%Y %H:%M:%S")

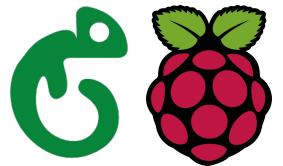
# test temperature en local
value = "ffcb1139012311162856600407981636000000"
valeurTemp = 21
adtype = 22

if adtype == 22:
    if len(value) == 38 and value[4:10] == "113901" and value[32:39] == "000000" :
        if 11000000 < int(value[12:20]) < 12000000:
            valeurId = value[12:20]
            valeurBatterieHexa = value[20:22]
            valeurTempHexa = value[24:28]

            valeurBatterie = int(valeurBatterieHexa, 16)
            valeurTemp = int(valeurTempHexa, 16)
            temperature = valeurTemp * (1/100)

        if valeurTemp < 2400 or valeurTemp > 3000 :
            # creation de l'objet
            msg = MIMEMultipart()

            # On cree une variable 'message' ou on met notre message
```



## Annexe 1 (suite)

```
message = "Bonjour,\nà la date et heure {}, le vivarium n°{} du magasin\n{} a détecté un problème.\nLe capteur avec le numéro d'identification n° {} a une tempé\nrature en dehors du seuil 24 -\n30°C.\nLa température du vivarium est actuellement de {} °C.\nVeuillez vérifier et règl\ner la température du vivarium en conséquence, merci.\n\nNote à part, le pourcentage de\nbatterie du capteur est de {} %".format(dt_string, vivarium, animalerie, valeurId, temp\nerature, valeurBatterie)\nprint(message)\n\n# On déclare 3 variables avec 3 paramètres différents (l'expéditeur, le\ndestinataire et le sujet) + une variable de mdp\npassword = "#####@\nmsg['From'] = "pythontest3333@gmail.com"\n\nmsg['To'] = "#####@gmail.com"\nmsg['Subject'] = dt_string + " Alerta Température vivariums VIVALAND"\n\n# Ajout dans le corps du message\nmsg.attach(MIMEText(message, 'plain'))\n\n# Création du serveur\nserver = smtplib.SMTP_SSL('smtp.gmail.com: 465')\n\n# Identification du script avec l'adresse et le mot de passe dans gmail\nserver.login(msg['From'], password)\n\n# Envoi du message via le serveur\nserver.sendmail(msg['From'], msg['To'], msg.as_string())\n\n# Fermeture du serveur\nserver.quit()\n\n# Message qui s'affiche si le mail a été envoyé avec succès\nprint ("Mail envoyé avec succès a %s" % (msg['To']))\nelse :\n    print(valeurId)\n    print("{0:.2f}".format(temperature))\n    print("pas d'envoie de mail")\n\nelse :\n    print("scanné mais pas trouvé les capteurs que l'on cherche")
```



## Annexe 2

```
recolteDonneesScan.txt - Bloc-notes
Fichier Edition Format Affichage Aide
date et heure : 12/11/2019 16:18:30 / ID : 11161806 / temp : 24.90000000000002°C / batterie : 93%
date et heure : 12/11/2019 16:18:45 / ID : 11162855 / temp : 27.57°C / batterie : 88%
date et heure : 12/11/2019 16:18:45 / ID : 11161806 / temp : 24.90000000000002°C / batterie : 93%
```

**Interface du Terminal à travers  
PuTTY (un client de connexion par protocole SSH et d'émulation de Terminal)**

```
pi@raspberrypi-5:~/Desktop/Projet/mod_scan_filtre $ sudo python3 scanFiltre3.py
pi@raspberrypi-5:~/Desktop/Projet/mod_scan_filtre $ sudo python3 scanFiltre3.py
pi@raspberrypi-5:~/Desktop/Projet/mod_scan_filtre $ sudo python3 scanFiltre3.py
true=====
ffccb11390123111618065f04098617e3000000
11161806
fin d'1 capteur
Le numéro d'identification du capteur est : 11161806
Le niveau de la batterie est de : 95 %
La température du capteur est de : 24.38 °C
pi@raspberrypi-5:~/Desktop/Projet/mod_scan_filtre $ sudo python3 scanFiltre3.py
pi@raspberrypi-5:~/Desktop/Projet/mod_scan_filtre $ sudo python3 scanFiltre3.py
^[[A
pi@raspberrypi-5:~/Desktop/Projet/mod_scan_filtre $ sudo python3 scanFiltre3.py
^[[Atrue=====
ffccb11390123111618065f04098717e3000000
11161806
fin d'1 capteur
Le numéro d'identification du capteur est : 11161806
Le niveau de la batterie est de : 95 %
Le température du capteur est de : 24.39 °C
pi@raspberrypi-5:~/Desktop/Projet/mod_scan_filtre $ sudo python3 scanFiltre2.py
false
Traceback (most recent call last):
```

```
pi@raspberrypi-5:~/Desktop/Projet/mod_scan_filtre $ sudo python3 scanFiltre3.py
true=====
ffccb11390123111618065f04098617e3000000
11161806
fin d'1 capteur
Le numéro d'identification du capteur est : 11161806
Le niveau de la batterie est de : 95 %
La température du capteur est de : 24.38 °C
```

Device d7:ef:13:27:15:29, RSSI=-77 dB  
8 = Short Local Name = DEMO  
22 = 16b Service Data = ffcb 11 3901 23 11162856 60 04 0798 1636 000000

Détail :

d7:ef:13:27:15:29 : adresse Mac  
8 ou 22 : le type de donnée de la ligne (adtype)

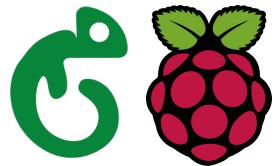
DEMO : nom du capteur

11162856 : ID

60 : batterie (96%), donnée en hexadécimal à déchiffrer en décimal  
0798 : température (19.44°C), donnée en hexadécimal à déchiffrer en décimal  
1636 : humidité (56.86%), donnée en hexadécimal à déchiffrer en décimal

22 = 16b Service Data = ffcb 11 3901 23 11162856 60 04 0798 1636 000000





## Annexe 3

### Algorithme pour l'envoi de mail de rétablissement

**Ce document explique le fonctionnement de l'algorithme qui sert à envoyer un mail de rétablissement, il sera implémenté par la suite dans le script principal. Il se peut que ce texte soit modifié par la suite.**

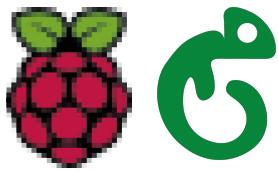
Le scan est lancé normalement, il commence à écouter tout ce qu'il y a autour de lui, il vérifie si les températures reçues des capteurs respectent le seuil qui est  $\geq 24$  et  $\leq 30$  :

- ▶ Si le seuil est respecté il continue de scanner toutes les minutes
- ▶ Si le seuil n'est pas respecté, le programme envoie une alerte par mail (le script du mail est exécuté) et reprend son cycle de scan.
  - └ Nous enregistrons la température relevée dans une variable, ou plusieurs variables si plusieurs températures ont été notées, pour que le programme puisse les comparer avec les informations du scan précédent : ceci permet une comparaison des données pour qu'elles soient contrôlées.

Si la température n'est toujours pas revenue à la normale, aucune notification ne sera envoyée puisqu'une première alerte aura été déclenchée.

Si la situation redévient acceptable, le programme enverra un mail pour signaler le rétablissement des conditions.

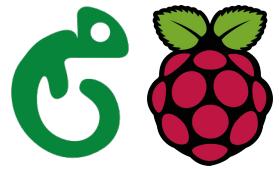
Le cycle se poursuit par la suite..



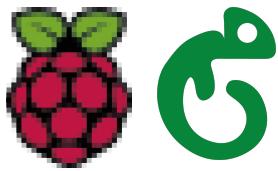
# Journal de bord

DATE	TÂCHES EFFECTUÉES	DIFFICULTÉS RENCONTRÉES
25/10/19	Prise des informations relatives à la demande du client  Reformulation de la demande et analyse des besoins  Rédaction du cahier des charges + envoi au client par mail  Test print Hello World  Tri des solutions possibles  Répartition des tâches	
04/11/19	Initialisation Raspbian et Raspberry  Installation de l'environnement sur Raspberry  Test de la carte BT  Edition du script pour scanner les adresses des capteurs et les récupérer + les filtrer  Import des librairies et installation Bluepy  Rédaction du mail d'envoi des infos	Scan et récupération des adresses des capteurs pour les isoler
06/11/19	Installation Putty et connexion SSH  Test script pour isoler les numéros d'identification des capteurs  Test d'un script pour convertir les données hexa pour récupérer les infos relatives à la température et au niveau de batterie des capteurs  Test de création d'une base de données à partir des infos collectées via un fichier .txt  Rédaction d'un glossaire	Fixer le script pour isoler les numéros d'identification des capteurs et convertir les hexa

# Journal de bord

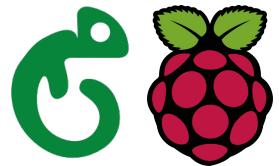


DATE	TÂCHES EFFECTUÉES	DIFFICULTÉS RENCONTRÉES
08/11/19	<p>Fixation adresse IP du Raspberry</p> <p>Test script pour filtrer les capteurs suite rajout condition « filtre sur les n° id compris entre 11000000 et 12000000 » dans le cas d'un rajout d'un capteur supplémentaire</p> <p>Début de rédaction du rapport (note des différentes tâches effectuées par jour + difficultés rencontrées)</p>	<p>Commande pour fixer l'adresse IP du Raspberry</p> <p>Test script filtrage</p>
12/11/19	<p>Script récupération données scan avec conversion des hexa</p> <p>Début création script pour récupérer date et heure des scans effectués</p> <p>Création base de données</p>	<p>Le temps, l'échéance approchait et du retard s'accumulait</p>
18/11/19	<p>Réflexion sur notre algorithme de boucle scan chaque minute et pour chaque heure</p> <p>Rajout de la boucle qui vérifie si une température hors du seuil est déjà existante. Si c'est le cas, ne renvoie pas d'e-mail d'alerte</p>	



## GLOSSAIRE

BDD	Base de données
BluePy	Librairie Bluetooth
Crontab	Outil qui permet de programmer une application de façon régulière, très pratique pour lancer des scripts de sauvegarde
GitHub	Outil gratuit pour héberger du code open source. Avec cet outil on peut communiquer avec d'autres développeurs et signaler des problèmes de code.
Filezilla	File Transfer Protocol (protocole de transfert de fichiers, est un protocole de communication dédié à l'échange de fichiers sur un réseau).  Il permet, depuis un ordinateur, de copier des fichiers depuis ou vers un autre ordinateur du réseau, d'administrer un site web, ou encore de supprimer ou modifier des fichiers sur cet ordinateur.
ID	Identifiant du capteur (ex 1112939)
Import	On utilise l'instruction import lorsque l'on veut utiliser un programme qui regroupe des attributs dans le programme principal
Librairies (BluePy, SMTPlib)	Ensemble de fonctions utilitaires, regroupées et mises à disposition afin de pouvoir être utilisées sans avoir à les réécrire
OS	Operating System = Système d'exploitation (Raspbian, Windows, etc)
Package	Ensemble de logiciels munis d'une documentation, conçus pour répondre à des besoins spécifiques et permettre une utilisation autonome
Procédural	Paradigme qui se fonde sur le concept procédural qui consiste à réaliser une série d'étapes
Raspberry Pi	Nano-ordinateur très petit qui peut être branché à un écran et utilisé comme un ordinateur standard
Raspbian	Système d'exploitation basé sur Debian pour fonctionner sur un Raspberry Pi



## GLOSSAIRE

RPI	Raspberry Pi
RSSI	Force de signal BLUETOOTH
Short Local Name	Nom du capteur
SSH	Secure Socket Shell est un protocole réseau qui permet aux administrateurs d'accéder à distance à un ordinateur, en toute sécurité.
SQLite	SQLite est une bibliothèque écrite en langage C qui propose un moteur de base de données relationnelle accessible par le langage SQL.
SMTP	Protocole utilisé pour transférer les messages électroniques sur les réseaux.
SMTPLib	Librairie pour l'envoi de mails
Test - Debug	Test du programme et analyse des bugs
Versionning	Gestion des différentes versions d'un programme

