

The Perfect Project

Table des matières

Introduction.....	2
PARTI 1 : Présentation commune du projet (20-30 pages).....	3
Présentation du Sujet.....	3
Le but du projet.....	3
Le Principe de réalisation du projet.....	4
<i>Synoptique simplifié du système.....</i>	<i>4</i>
<i>Synoptique simplifié du boîtier de régulation autonome.....</i>	<i>4</i>
Analyse Fonctionnelle du système.....	5
<i>Diagramme de cas d'utilisation simplifié.....</i>	<i>5</i>
<i>Diagramme d'exigence.....</i>	<i>5</i>
<i>Diagramme de classe.....</i>	<i>6</i>
<i>Diagramme de séquence.....</i>	<i>7</i>
Organisation du projet.....	8
<i>GANTT Prévisionnel.....</i>	<i>8</i>
<i>Gantt Réel.....</i>	<i>9</i>
<i>Répartition des tâches.....</i>	<i>9</i>
Organisation de l'équipe.....	10
<i>Compte rendu d'activité (CRA).....</i>	<i>10</i>
<i>Cahier de bord.....</i>	<i>10</i>
<i>Git Hub et Versionning.....</i>	<i>11</i>
<i>Démarrage projet et classe de simulation.....</i>	<i>11</i>
<i>Logiciel d'analyse et de développement.....</i>	<i>12</i>
<i>Maquettage et Prototype.....</i>	<i>13</i>
Choix technique et Etude physique.....	14
<i>Choix de la carte contrôleur pour le boîtier connecté.....</i>	<i>14</i>
<i>Choix des capteurs et module.....</i>	<i>14</i>
Recette.....	14
Tests d'intégration du prototype.....	16
Avancement et Conclusion.....	16
PARTI 2 : Partie Individuel Julien Langlacé (20-30 pages).....	17
Fonctionnalité 1.....	17
<i>1 Module sous-test coté Raspberry.....</i>	<i>17</i>
<i>2 Module sous-test coté Arduino.....</i>	<i>19</i>
<i>3 Module test Complet communication RS433.....</i>	<i>20</i>
<i>Diagramme de séquence Consigne de chauffe Arduino.....</i>	<i>21</i>

Introduction

La rédaction de ce dossier de projet va permettre aux élèves de BTS SN de s'inspirer de ce dernier pour rédiger un plan pour leur dossier projet. Le contenu de ce projet va permettre de mieux comprendre la méthodologie et les attendus pour réaliser un projet from scrtach (de A à Z). Le sujet de se projet est un sujet libre qui regroupe les grands axe du référentiel de BTS SN. Il y aura donc de l'analyse SysML-UML , du développement Web et de bas niveau C++. De la communication matériel, du réseau et des études physiques. Il y aura aussi des explications sur l'organisation projet (Suivi de projet , versionning etc..).

Le but de la lecture de ce dossier est donc de bien comprendre comment rédiger un dossier projet mais aussi d'apprendre à appréhender au mieux un nouveau projet pour le mener à bien et réussi l'examen du BTS.

Un projet est un travail d'équipe. Il y aura donc une partie Commune présentant le projet. Et une partie personnelle concernant l'investissement personnel de l'étudiant.

Voici le contenue de la partie commune

- introduction, situation du projet dans son contexte industriel
- dossier de spécifications
- dossier d'étude préliminaire et plan de tests des performances au regard du cahier des charges. Suivant la nature du projet et ses points d'entrée, certains éléments de ce dossier peuvent être présents dans les parties personnelles.
- éléments nécessaires à la recette de la maquette ou du prototype final
- résultats des essais de la maquette ou du prototype final
- conclusion par rapport au cahier des charges fourni par le donneur d'ordre : test intégration, procédure et résultats de la recete.

PARTI 1 : Présentation commune du projet (20-30 pages)

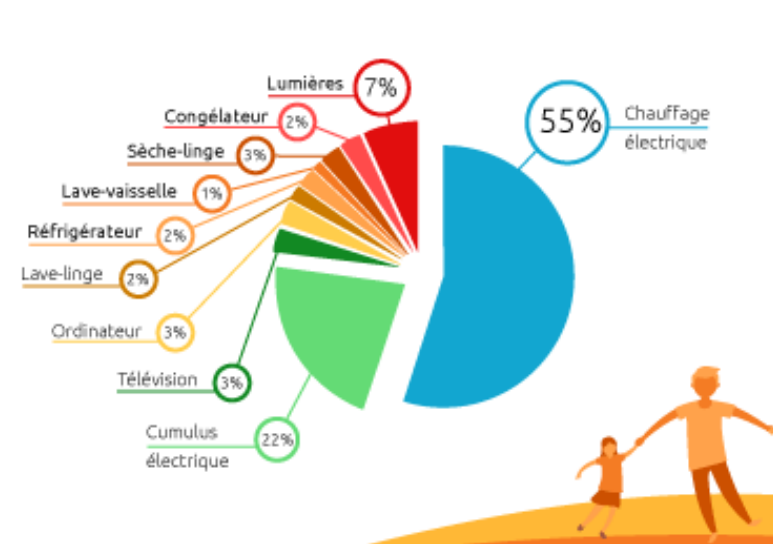
Présentation du Sujet

[IMAGE DE VOTRE CLIENT] Présentez ici votre projet dans les grandes lignes (entre 10 et 15 lignes)

Le but du projet

Le but du projet est donc de répondre à deux points importants du cahier des charges :

Parts moyennes des équipements pour un logement dans la consommation d'électricité



-1 Fonction 1

-2 Fonction 2

Dans cette partie vous expliquez le but du projet c'est à dire pourquoi le client a besoin de votre système

Le Principe de réalisation du projet

Pour répondre à ces problématiques (au but) . vous allez réaliser une maquette . dans la partie principe vous expliquerez comment vous allez faire pour répondre aux problématiques (BUT) de votre projet.

Synoptique simplifié du système.

Le synoptique simplifie la vision de la demande du client. Expliquer votre synoptique avec quelques ligne (une dizaine)

Synoptique simplifié du boitier de régulation autonome

Il peut y avoir plusieurs synoptique.

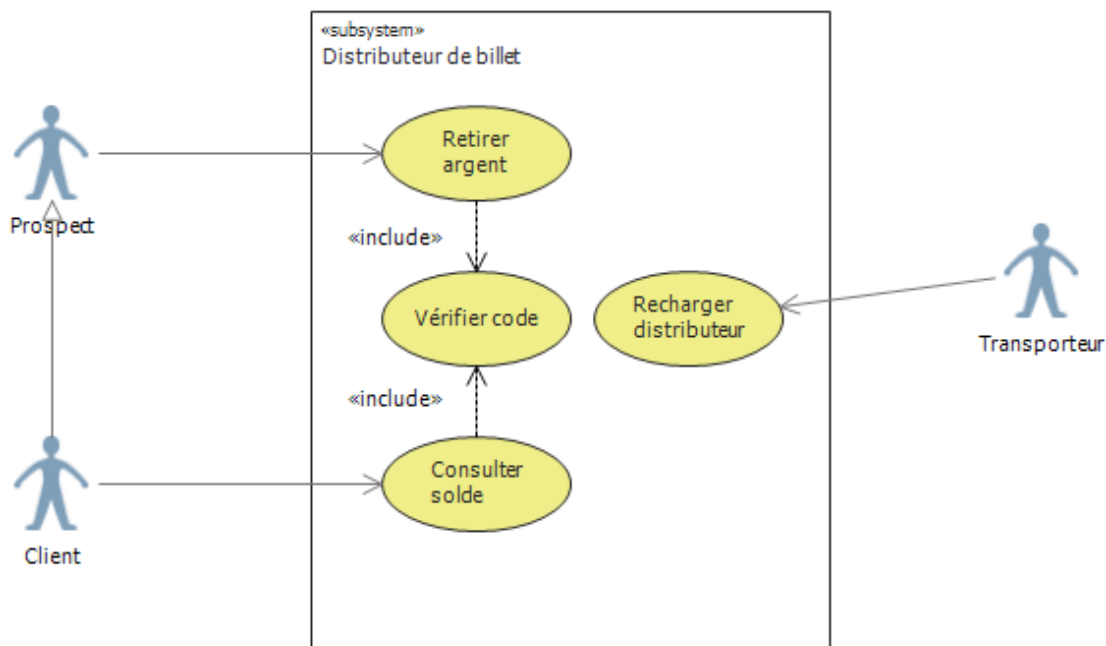
Analyse Fonctionnelle du système

Dans cette partie vous présenter l'analyse avant la phase de conception. Durant la phase du projet et les échanges avec le client. Certain point pourront être amené à changer pour sécuriser le système ou contourner des contraintes non prévu durant l'analyse.

Diagramme de cas d'utilisation simplifié

Pour réaliser ce diagramme j'ai utilisé l'application web :

<https://online.visual-paradigm.com>

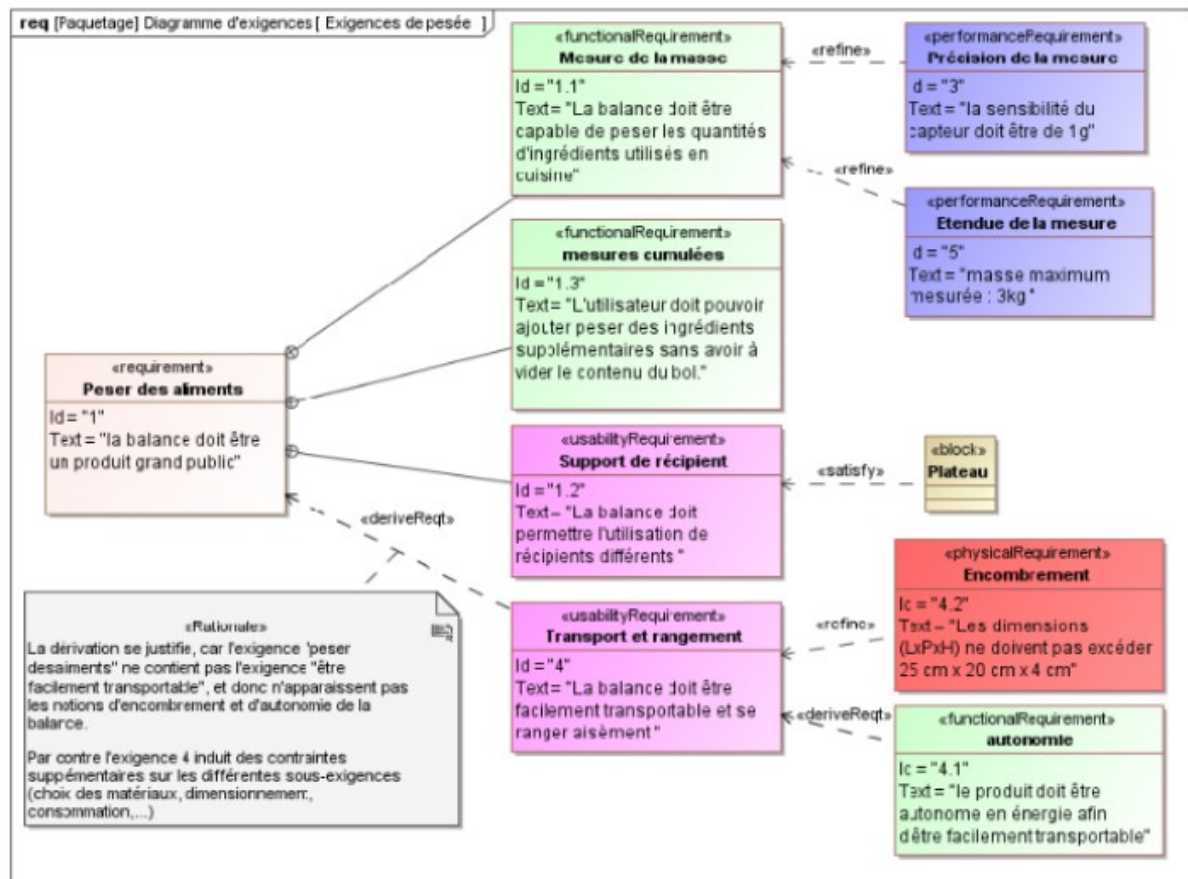


les bulles jaunes représentent les fonctionnalités que propose le système.

Vous expliquerez en quelques lignes les interactions entre les acteurs et le système

Ce diagramme doit être un diagramme du projet simplifié, il sera détaillé dans vos parties perso.

Diagramme d'exigence

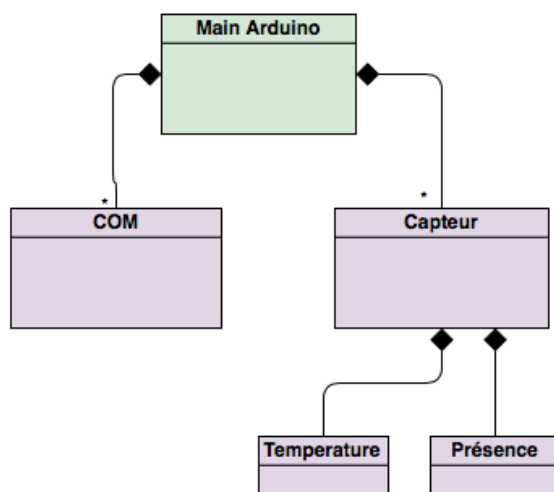


Durant la phase de programmation il faudra bien prendre en compte ces exigences et procéder à un test de conformité de chacune d'entre elle.

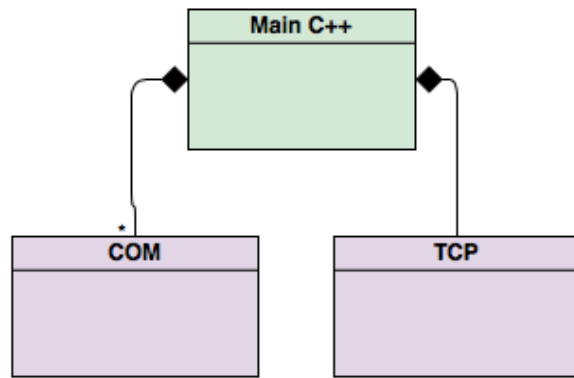
Diagramme de classe

Commencer par présenter le diagramme de classe. Les classe on été sélectionné en décomposant toutes les entités du projet. Une fois les classes identifiées. Le diagramme de séquence de chaque fonctionnalité du système. (use case , diagramme de cas d'utilisation) sera expliqué dans les partie perso

Vous ne présenter pas les méthodes et propriété de ses classes. Ceci afin d'avoir une vision global du système à programmé plus claire.



Vous pouvez faire plusieurs sous partie : vous expliquerez en quelques lignes les relations et liens entre les classes



Pour la partie 2: Vous pouvez découper votre diagramme de classe en petit groupe s'il n'y a pas de lien entre les 2

Diagramme de séquence

Les diagrammes de séquences seront présenté par chacun des étudiants en charges de ces derniers dans la partie perso

1. *Journal of the American Medical Association*, 1997; 278: 1019-1024.



1. *Journal of the American Medical Association*, 2000; 283: 2686-2692.

Le Gantt est décomposé en semaine de travail. Les semaines 1 à 9 sont des semaines entre 8h et 10h de Projet. Les semaines 10 à 14 sont des semaines à temps plein. Avant les revue 1 et 2 un travail de relecture et de mise a jours des diagrammes fonctionnelles sont nécessaire. Les Zones grises correspondent au temps de travail pendant les vacances pour rattraper le retard.

A la rédaction de ce projet le Gantt réel est le suivant.

Gantt Réel

Semaine :	1	2	3		4	5	6	7	8		9	10	11	12	13	14
Analyse du sujet																
Préparation Projet																
Use Case																
Classe																
MCD																
Séquence																
Exigence																
Déploiement																
Recherche Matériel																
Modules de Test																
Développement																
Revue 0																
Revue 1																
Revue 2																
Intégration / test																

L'analyse à pris plus de temps 3 semaines de plus et le début du développement spécifique aurait du commencer il y a 2 semaine.

Il est possible de superposé les deux avec un template excel adapté.

Répartition des tâches.

Voici la répartition des tâches. Cette dernière nous est imposée par le sujet de BTS SN.

Etudiant 1 :

Etudiant 2 :

Etudiant 3 :

Organisation de l'équipe.

Pour communiquer entre vous, vous utilisez la platform GitHub pour vos fichiers de code et la platform Discord Pour Communiquer entre vous à distance hors des heures projets en laboratoire. Nous avez une copie de nos fichiers sur un cloud ?

Compte rendu d'activité (CRA)

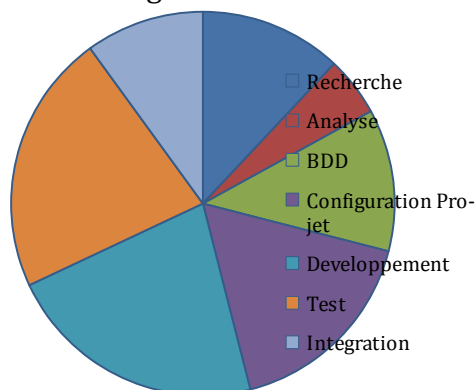
Pour avoir un suivi de votre activité avons devez utiliser un CRA Horaire sous excel

N° Heure	Date	Professeur	Domaine Technique	Info	N°1 Tâche effectuée
1	19/01	Langlacé	Recherche		Lecture du sujet
2	24/01				

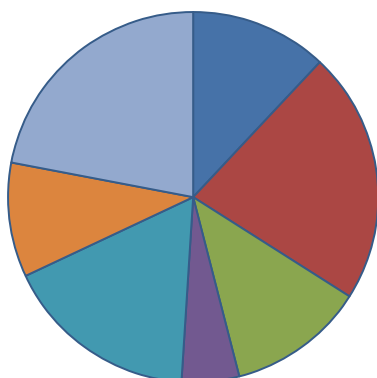
Mettre seulement un extrait ici pour expliquer votre suivi. Vous aurez une version détaillé dans votre partie perso

Chaque heure nous historions les tâches de travail effectuées.

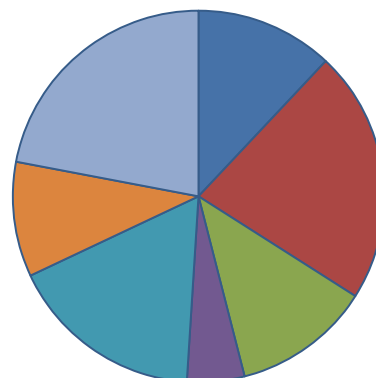
Etudiant 1 : M. Langlacé



Etudiant 2 : M. Gremont



Etudiant 3 : M. Grout



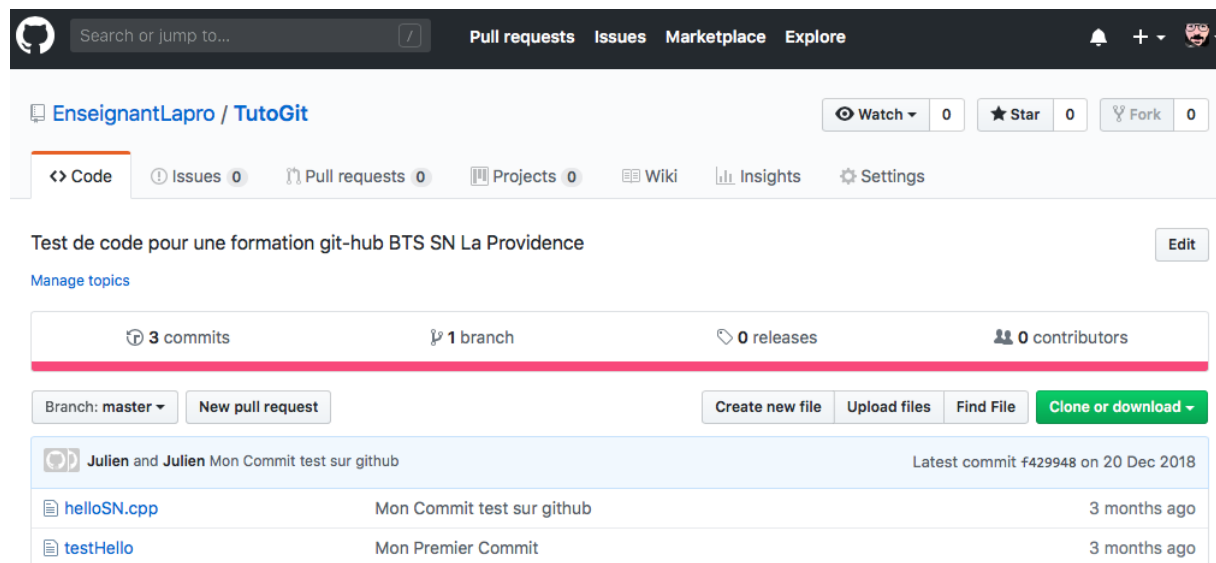
Cahier de bord

Chaque jours vous devez tenir à jour un cahier de bord, avec les différents travaux réaliser durant les heures de projet. Ce cahier de bord permet de faire une synthèse de travaux qui seront présenté dans les parties individuelles de ce rapport de projet.

Git Hub et Versionning

Expliquez ici comment vous versionnez

Pour travailler en collaboration vous avez utilisé le logiciel de versionning GIT. Ainsi que la Plateforme d'hébergement Git-hub.com. Sur vos PC de développement vous avons une version de notre code source avec nos différents "Commit" une fois qu'une fonctionnalité est opérationnel nous la poussons sur le site hébergement "Push" pour que tous les membres du projet puisse avoir la fonctionnalité.



Screen de IHM de versionning Git Hub

A chaque instant sur l'hébergeur il y a la version la plus à jours de notre projet. Et il est très facile de récupérer une version antérieure en cas de soucis. En début de projet nous avons créé toutes nos classes et toutes les méthodes utilisées dans nos diagrammes de séquence.

Démarrage projet et classe de simulation

Expliquer comment vous avez amorcé votre projet au niveau code

En début de projet toutes les méthodes des classes ne sont pas implémentés, elle le sont au fur et à mesure de l'avancement du projet. Les méthodes sont donc vide mais retourne une valeur attendu simulé. Ainsi si un développeur utilise la classe BDD non implémenté il peut quand même l'utiliser en mode simulation. Cette dernière sera un moment implémenté et "commité" sur le projet sans impacter celui qui l'utilise.

Voici un exemple de classe simulé (php)

Class User {	Utilisation dans index.php
<pre>\$Nom ; \$Prenom ; SeConnecter() { Return true ;</pre>	<pre>\$user = new User ; //on va vérifier que le user est connecté if (\$user->SeConnecter()) { //affichage partie admin } else {</pre>

<pre>} }</pre>	<pre>echo « access refusé » ; }</pre>
--------------------	---

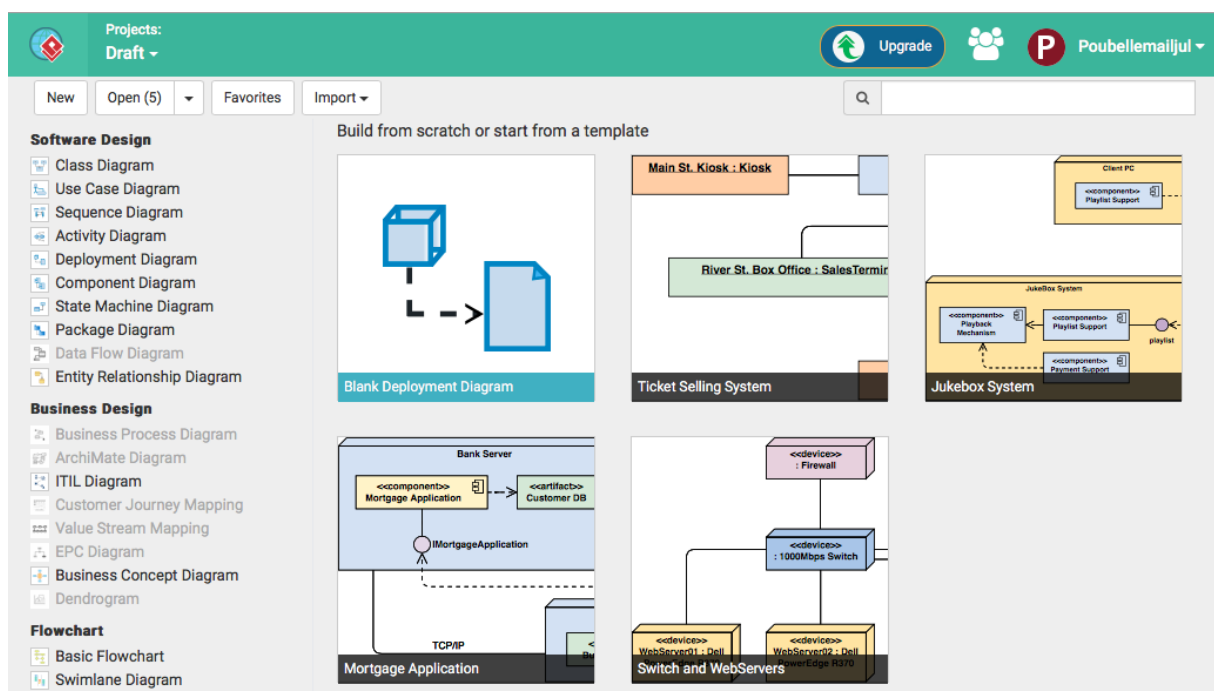
On remarque ici que la méthode `SeConnecter()` n'est pas implémenté il n'y a pas encore l'algo de connexion mais celle ci est utilisable des le début de projet car elle retourne une valeur simulé ici « true »

Logiciel d'analyse et de développement

Expliquez les logiques que vous utilisez

Pour réaliser nos diagrammes nous avons utilisé l'outil en ligne :

<https://online.visual-paradigm.com>



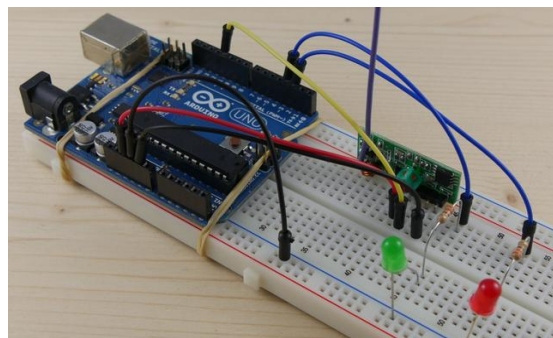
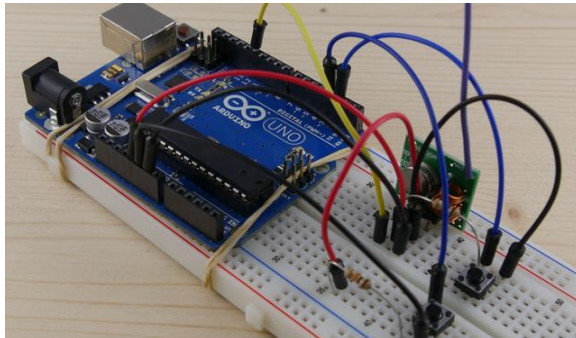
Cette application en ligne permet de retrouver nos diagrammes sur n'importe quelle machine en mode édition. Pas besoin d'installer d'application lourde. L'espace gratuit est suffisant pour nos besoins projet.

Maquettage et Prototype

Ici sera détaillé le prototype de votre projet (pas besoin pour la revu 0)

Pour le prototype nous avons utilisé une carte programmable c++ Arduino. La passerelle TCP est une application C++ développée pour tourner sur un processeur embarqué ARM grâce à la carte Raspberry. On utilise des VM avec VirtualBox pour nos serveurs applicatifs de type LAMP.

Partie boîtier connecté autonome (Capteurs)



Vous expliquerez votre montage en quelques lignes (10 – 15)

Partie autre (Communication TCP)

Il peut y avoir d'autres prototypes

On utilise un serveur Apache/MySQL avec du développement PHP et son module TCP.



Le serveur MySQL et le serveur Web sont pour le moment simulés pour le prototype sur une VM de type Debian. Avec Apache et MariaDB configurés.

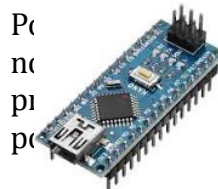
Choix technique et Etude physique.

Pas besoin de cette partie dans la revue 0

Dans cette partie nous allons présenter uniquement les solutions retenues. Les études techniques des autres solutions seront détaillées dans les parties individuelles.

Choix de la carte contrôleur pour le boîtier connecté



Pour nous choisissons une carte de programmable Arduino sont prix ses entrée sortie et sa facilité de programmation pourra facilité la phase de programmation. Suite il sera plus judicieux de programmer la puce ATmega328p seule volume dans le boîtier. (3€)



Vous devez expliquer vos choix




Choix des capteurs et module


Pour le prototype nous avons opté pour des composants chinois de très faible coût. Chaque capteur pourra être amélioré en gamme pour une version de production.

	Capteur de température : LM35DZ TO-92 LM35 Precision Centigrade Temperature Sensor	1,45€
	Module RF 433mhz : XD-RF-5V 433Mhz RF Decoder Transmitter With Receiver Module Kit For Arduino ARM MCU Wireless	3,52€
Etc	Etc etc	Etc

Recette.

Pour valider la recette client nous reprenons toutes les fonctionnalités attendues du système. Elles seront validées par un test d'intégration. Dans la partie individuelle de chacun nous avons une recette des fonctionnalités détaillées qui seront validées par des tests unitaires.

Fonctionnalités du système + Nom du test pour validation	ETAT (OK ou NOK)	Commentaires
Paramétrer une consigne de chauffe +testConsigne		
Afficher l'état des radiateurs via IHM +testIHM		Avancement du projet ne permet pas de réaliser ce test en intégralité.
Détecter la présence humaine d'une pièce +testPrésence		

Etc + test		
------------	---	--



Tests d'intégration du prototype.

Pas besoin dans la revue 0

Cahier de test d'intégration.

Les tests unitaires seront validés dans les parties individuelles de chacun.

Nom du test	Détail du test	Résultat du test
testConsigne	Saisie d'une consigne depuis l'IHM central pour un radiateur Cliquer sur , Valider Ect ect	La température de la pièce doit atteindre la nouvelle consigne
testIHM	Aller sur IHM et voir l'état des radiateurs (connecté ou déconnecté + consigne en cours)	Les radiateurs change d'état lorsqu'on déconnecte un boîtier
testPrésence	Se placer devant le capteur. Bouger Sauter ect	Une led indique que la présence est en cours.
ect		

Avancement et Conclusion.

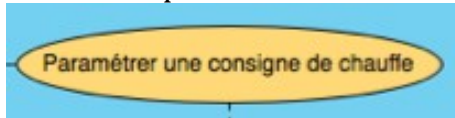
Ici vous détailler les points faibles les points fort de votre gestion les problèmes rencontrés et résolus

PARTI 2 : Partie Individuel Julien Langlacé (20-30 pages)

Dans la partie perso tout est découpé par fonctionnalité (pas besoin dans la revue 0)

Fonctionnalité 1

Mettre une photo de la bulle

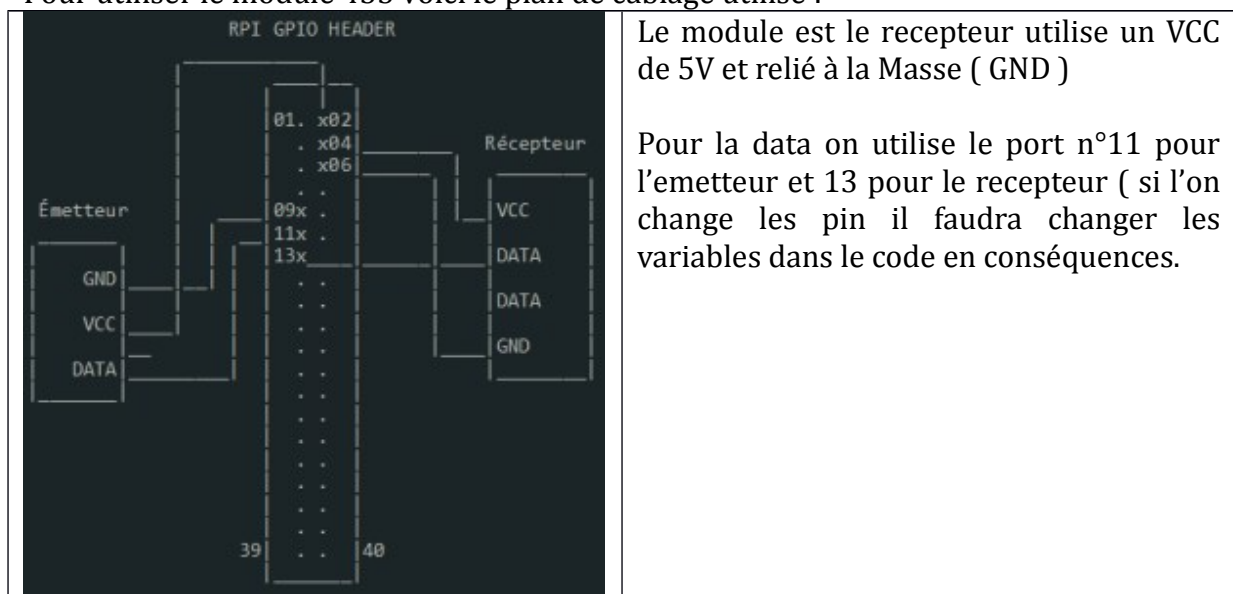


expliquez le scénario de la fonctionnalité

les sous-parties sont les modules de test pour valider cette fonctionnalité
vous devez le détaillé avec un Code , un diagramme de séquence , des captures d'écran de test , des explication exemple :

1 Module sous-test coté Raspberry

Pour utiliser le module 433 voici le plan de câblage utilisé :



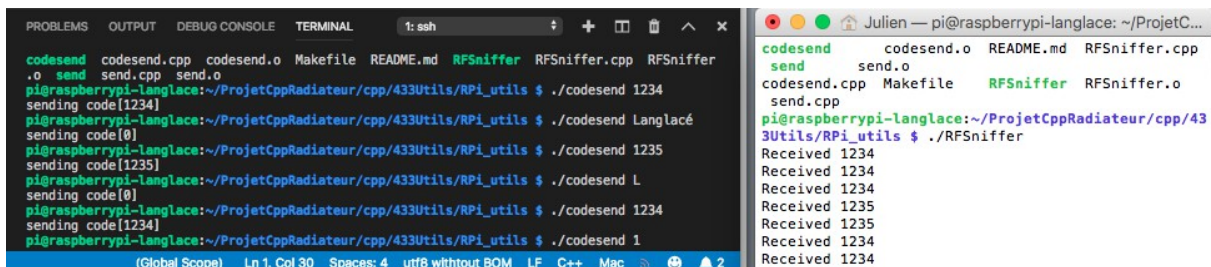
Nous ne devons pas réécrire le code pour gérer ces modules j'utilise alors une bibliothèque en C++ trouvé sur GitHub

- Installation de la gestion des GPIO raspberry (en mode root à la racine)
[shell]
git clone git://git.drogon.net/wiringPi
cd wiringPi
./build
[/shell]
- installation de la librairie 433 (dans mon projet)
[shell]

```
git clone git://github.com/ninjablocks/433Utils.git
cd 433Utils/RPi_utils
git submodule init
git submodule update
make
[shell]
```

```
Arduino_sketches CHIP_utils LICENSE rc-switch README.md RPi_utils
pi@raspberrypi-langlace:~/ProjetCppRadiateur/cpp/433Utils $ cd RPi_utils/
pi@raspberrypi-langlace:~/ProjetCppRadiateur/cpp/433Utils/RPi_utils $ ls
codesend.cpp Makefile README.md RFSniffer.cpp send.cpp
pi@raspberrypi-langlace:~/ProjetCppRadiateur/cpp/433Utils/RPi_utils $ make
g++ -DRPI -c -o ../rc-switch/RCSwitch.o ../rc-switch/RCSwitch.cpp
g++ -DRPI -c -o send.o send.cpp
g++ -DRPI ../rc-switch/RCSwitch.o send.o -o send -lwiringPi -lwiringPiDev -lcrypt
g++ -DRPI -c -o codesend.o codesend.cpp
g++ -DRPI ../rc-switch/RCSwitch.o codesend.o -o codesend -lwiringPi -lwiringPiDev -lcrypt
g++ -DRPI -c -o RFSniffer.o RFSniffer.cpp
g++ -DRPI ../rc-switch/RCSwitch.o RFSniffer.o -o RFSniffer -lwiringPi -lwiringPiDev -lcrypt
pi@raspberrypi-langlace:~/ProjetCppRadiateur/cpp/433Utils/RPi_utils $
```

Test des outils du module



The screenshot shows two terminal windows. The left window, titled 'ssh', shows the execution of the 'codesend' program with various arguments, including '1234', 'Langlacé', '1235', 'L', '1234', and '1'. The right window, titled 'Julien — pi@raspberrypi-langlace: ~/ProjetC...', shows the execution of the 'RFSniffer' program, which displays 'Received' messages for the same values: '1234', '1234', '1234', '1235', '1234', and '1234'.

(sending code 1234 à gauche et Received 1234 à droite)

Pour faire ce test j'ai utilisé 2 outils : ./RFSniffer et ./CodeSend dans 2 consoles différentes (en noir ./codesend et en blanc le ./RFSniffer). Le test fonctionne j'ai bien une information qui part du module d'émission vers la réception.

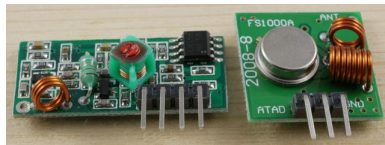
Problèmes

Le problème N°1 c'est que ce test n'est pas encore intégré dans une mon appli C++

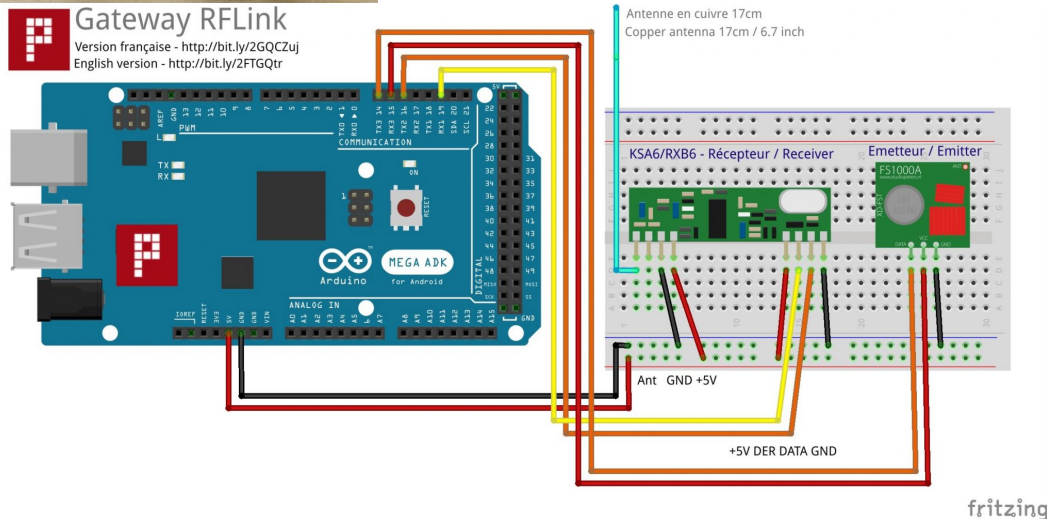
Le problème N°2 c'est que le protocole reconnu par la librairie de test ne comprends que les chiffres jusque 999999.

2 Module sous-test coté Arduino

j'utilise les mêmes modules pour être sûr de la compatibilité :

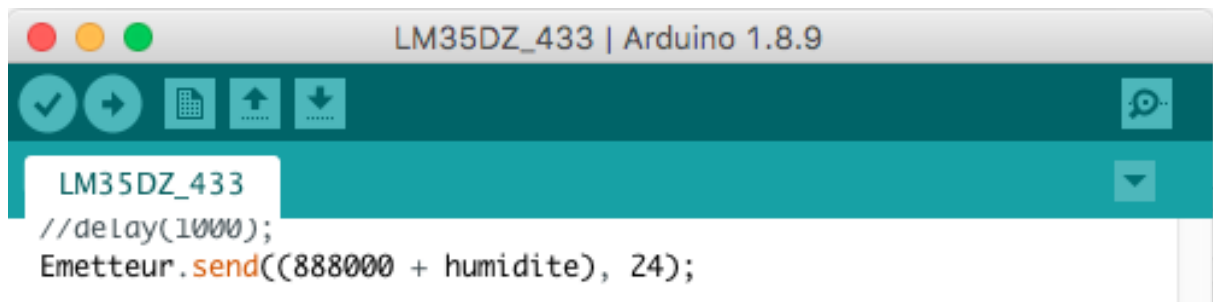


Voici le schéma de câblage coté arduino.
J'utilise le VCC 5v et le GND de l'arduino Nano
J'utilise l'entrée Numérique D2 et D10 du nano



Pour tester ce montage j'utilise la classe
`#include <RCSwitch.h>` que j'instance en objet Emetteur
`RCSwitch Emetteur = RCSwitch();`

J'utilise la méthode `send` de l'objet Emetteur de type `RCSwitch()`

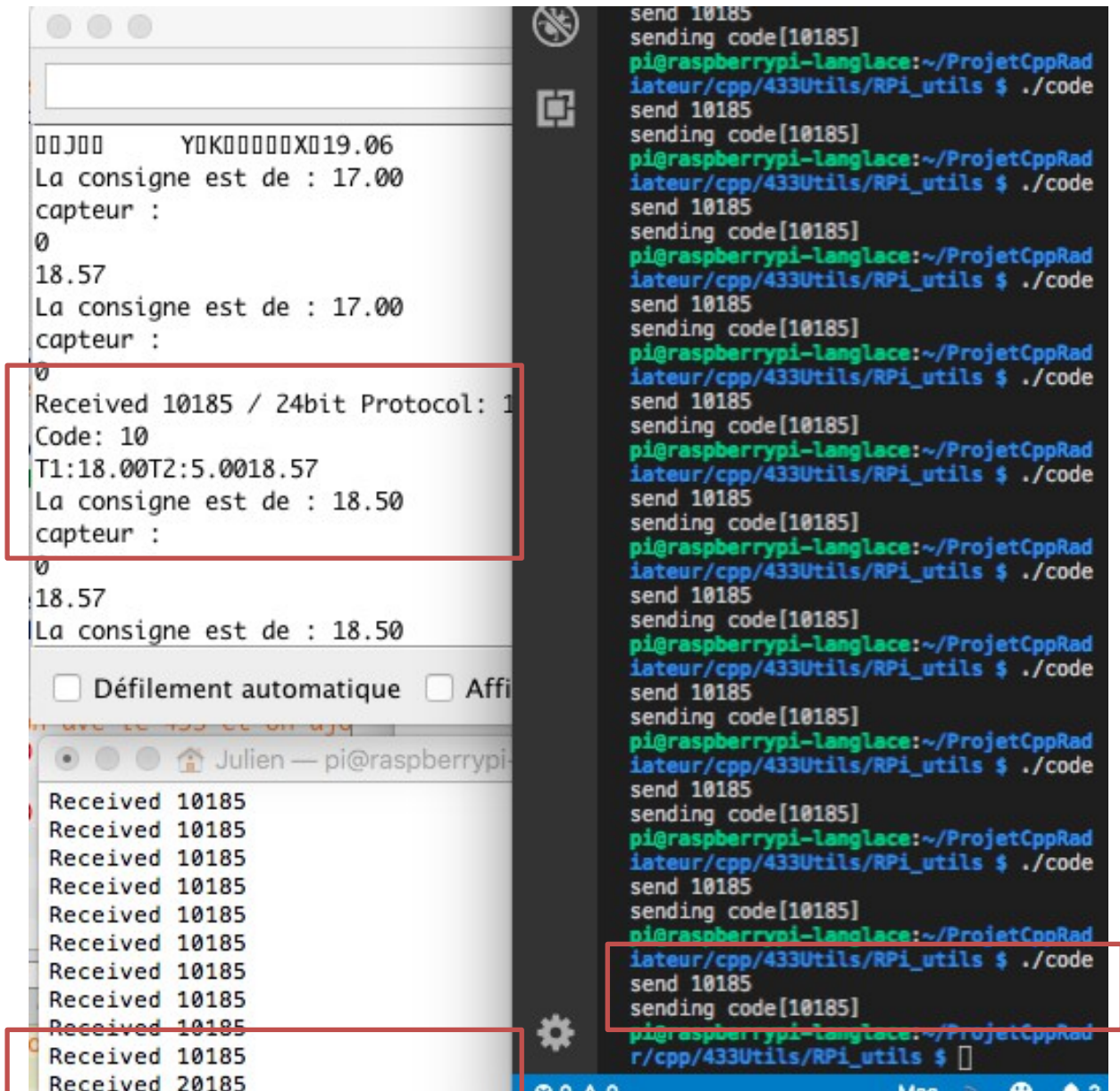


Lorsque j'exécute le code sur l'arduino et que je regarde le côté RFSniffer sur ma raspberry je retrouve mon message ici 888000 (humidité = 0)

```
pi@raspberrypi-langlace:~/ProjetCppRadiateur/cpp/433Utils/RPi_utils $ ./RFSniffer
Received 999000
Received 888000
Received 888000
Received 888000
```

3 Module test Complet communication RS433

En modifiant le code coté arduino et raspberry je vais tester une communication en bouche : le Raspberry envoie le code 10+La température de consigne sur 3 chiffres 10375 c'est 37,5° (protocole provisoire) l'arduino s'il reçoit le code 10 Met à jour sa consigne et renvoie un message de confirmation 20 + la consigne saisie.

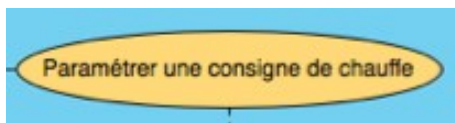


- A droite le module raspberry (noir) envoi le code 10185 (protocole 10+ Consigne sur 3 chiffre 2 entier et le 3^{ième} est le chiffre après la virgule.
- En haut à gauche la console arduino : la consigne était de 17 on reçoit le code 10 et on met à jour la consigne ici 185 => 18,5°
- En bas à gauche le module de reception raspberry il reçoit bien le code 20+ Consigne pour indiquer que la consigne à bien été prise en compte par arduino.

Le module fonctionne mais il reste de sérieux problème de sécurité. Le protocole doit être amélioré par exemple avec un identifiant.

Problèmes

Problème 1 : Arduino reçoit bien le code 10+ température (en haut sur la capture) mais le sniffer raspberry ne reçoit pas tout le temps la. Le protocole et l'algo doivent donc être améliorés avec des délais et des réponses de contrôle. En effet l'arduino étant une boucle qui tourne plus ou moins vite il suffit que l'émetteur et le récepteur entre raspberry et arduino soit un moment désynchronisé pour rater une réception.



Est donc partiellement validé par ce module de test.

Il reste le code C++ raspberry rattaché à un client PHP pour saisir la consigne via une IHM web.

Voici le diagramme de séquence final pour la programmation de cette fonctionnalité :

Diagramme de séquence Consigne de chauffe Arduino

Ici vous devez mettre le diagramme de séquence pour la fonctionnalité étudiée

Reproduisez cette mise en page pour chaque module et chaque fonctionnalité (1 fonctionnalité = 1 ou N module de test)