TRACKING GPS

Session 2020

BTS SYSTEME NUMERIQUE OPTION INFORMATIQUE ET RESEAUX

TRACKING GPS

LA PROVIDENCE |146 boulevard St Quentin 80090 Amiens

**Table des matières .**

**PARTIE 1 : Présentation commune du projet 3**

Présentation du sujet 3

Le but du projet 3

Le principe de réalisation du projet 4

*Synoptique simplifié du système* 4

*Synoptique simplifié du boitier de régulation autonome* 5

Analyse fonctionnelle du système 6

*Diagramme de cas d’utilisation* 6

*Diagramme d’exigences* 7

*Diagramme de séquence* 7

*Diagramme de Classe* 8

*MCD* 10

Organisation du projet 11

*GANTT prévisionnel* 12

*GANTT réel* 13

*Répartitions des tâches* 14

Organisation de l’équipe 15

*Compte rendu d’activité (CRA)* 15

*Cahier de bord* 17

*GitHub et versionning* 17

*Démarrage de projet et classe de simulation* 18

*Logiciel d’analyse et de développement* 19

*Maquette et prototype* 20

Choix technique et étude technique 21

*Choix de la carte contrôleur pour le système embarqué* 21

*Choix des capteurs et module* 21 *Choix des moyens de communication* 22

*Liaison Système Embarquée 🡨🡪 TCP* 22

*Liaison TCP 🡨🡪 BDD / Liaison TCP 🡨🡪 IHM Web* 22

Recette 23

Test d’intégration du prototype 24

Avancement et conclusion 26

**PARTIE 2 : Partie individuelle MARTIN Vincent 27**

Diagramme de cas d’utilisation détaillé 27

Analyse Physique 28

*Définition du système de navigation GPS* 28

*Principe de la triangulation* 29

Module de test réception de la trame système embarqué 30

*Diagramme de séquence de réception de la trame* 30

*Scénarios : réception de la trame* 33

Module de test découpé trame système embarqué 34

*Diagramme de séquence découper trame* 34

*Scénarios : réception de la trame*

Module de test envoyer à la BDD trame système embarqué

*Diagramme de séquence envoyer trame à la BDD*

*Scénarios : réception de la trame*

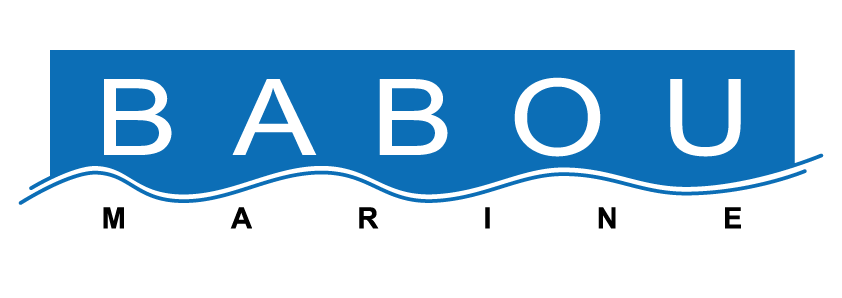
Module de test découpé trame système embarqué 34

*Diagramme de séquence découper trame* 34

*Scénarios : réception de la trame*

PARTIE 1 : Présentation commune du projet .

Présentation du sujet .

**

Le client est un gestionnaire d’une agence proposant la location de bateaux à moteur sans permis. Babou marine est implanté à Cahors dans la commune de Long depuis 1928. Cette entreprise au niveau national est spécialisée dans les croisières fluviales dans toute la France grâce à ses nombreuses agences.

Voici un extrait du cahier des charges du projet :

*« Actuellement, il n’y a pas de système de géolocalisation en temps réel ni de système embarqué permettant d’envoyer la vitesse et la profondeur instantanée et ainsi prévenir en cas d’anomalie. Pour localiser leurs bateaux, la société fait appel à la bonne fois des plaisanciers et aux différents checkpoint que doivent effectuer les bateaux sur les différents fleuves (passage d’écluse par exemple) ».*

Notre projet sera donc de proposer un prototype de système de tracking GPS national communiquant par réseau mobile permettant de savoir en temps réel où sont situés les bateaux sans permis en cours d’utilisation dans toute la France ainsi que leur profondeur, vitesse et niveau de batterie restante.

Le but du projet .

Le but du projet est de répondre à 5 points importants du cahier des charges :

1. Les systèmes embarqués doivent être auto alimentés.
2. Les systèmes embarqués permettent d’afficher et d’envoyer via un réseau mobile en temps réel le positionnement GPS, la profondeur (en mètre), la vitesse (en km/h) et le niveau de batterie restante (en %).
3. Les systèmes embarqués doivent détecter automatiquement ou recevoir une anomalie.
4. Centraliser toutes les données concernant les bateaux sur un site web.
5. Historier toutes les données concernant les bateaux sur un site web.

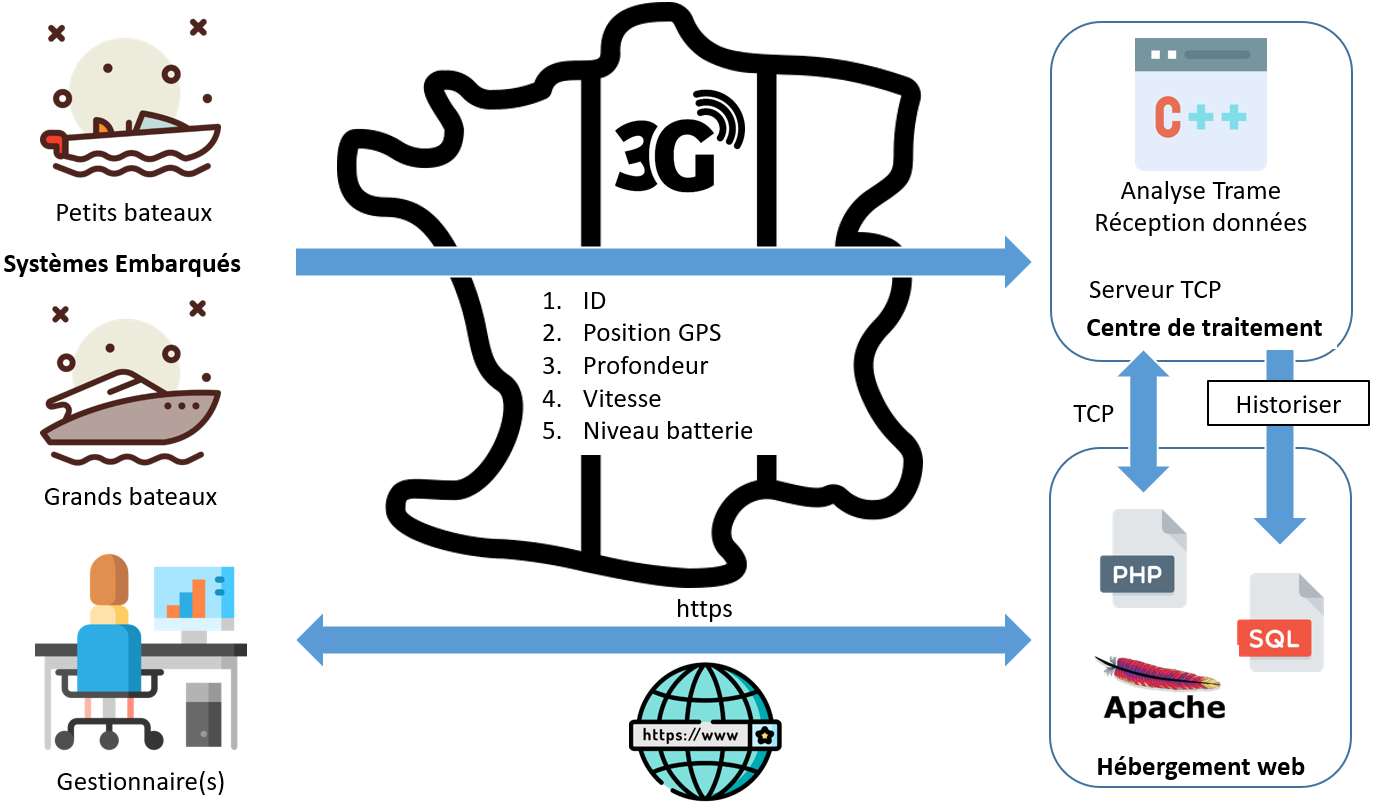
Cela va permettre aux agences de gestion de location dans les différents fleuves de France de mieux informer en temps réel les plaisanciers des informations importantes sur le trafic ou les dangers que risque un débutant sur son parcours (courant, écluse, cul-de-sac, niveau d’eau, zone interdite, etc.) à l’aide d’un site web centralisant et historisant toutes les informations (positionnement GPS, vitesse, profondeur, niveau de batterie) concernant les bateaux de l’agence sur une carte mise à jour en continu.

Le principe de réalisation du projet .

Pour répondre à ces problématiques. Nous allons réaliser un prototype de système embarqué autonome qui pourra sans risque être placé dans un bateau. Ce dernier permettra au bateau d’envoyer toutes ses informations en temps réel.

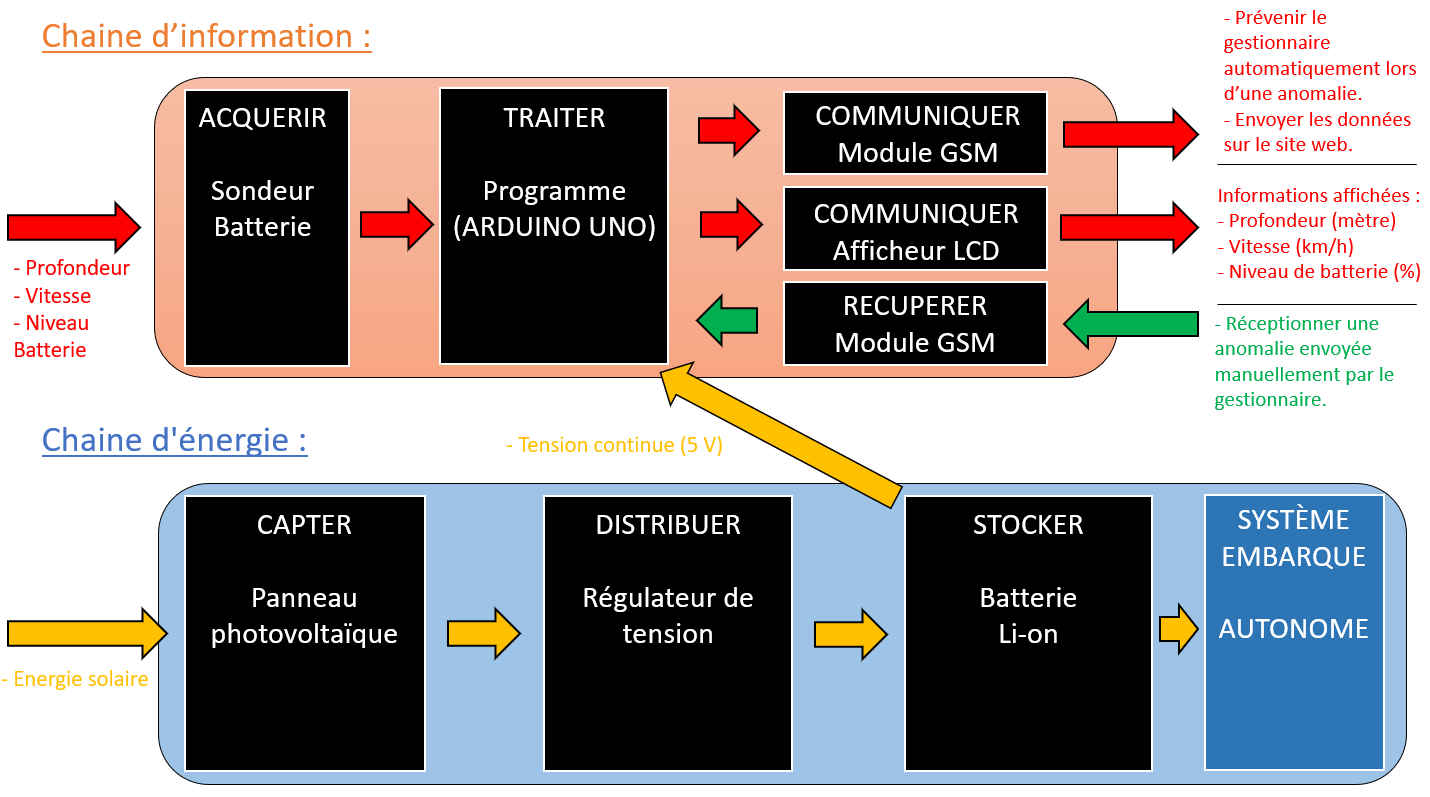
Il sera connecté à un réseau GSM pour centraliser tous les bateaux en cours d’utilisation possédant ce système. Ainsi, il sera facile à distance de connaître l’état des bateaux et de recevoir automatiquement ou d’envoyer une anomalie si l’on souhaite être informé ou prévenu d’un danger.

Synoptique simplifié du système



Le synoptique ci-dessus simplifie la vision de la demande du client. Nous pouvons voir qu’il existe 2 catégories de bateaux qui embarquent un système embarqué qui est chargé de transférer vers un centre de traitement 5 types de données qui sont reçues, analysées puis envoyées vers un hébergement web qui va afficher et historier les données pour que toutes les données soient consultables sur un site web accessible par les gestionnaires en temps réel.

Synoptique simplifié du système embarqué autonome



Lors de la première étude sur le système embarqué nous avons tout d’abord décidé des composants dont nous aurons besoin afin de gérer un seul bateau :

* Un sondeur passe-coque pour connaître la profondeur et la vitesse du bateau.
* Une carte programmable afin de commander les différents composants.
* Un module GSM pour envoyer et recevoir les données via un réseau mobile.
* Un module GPS pour connaître le positionnement GPS du bateau.
* Un afficheur OLED pour que le plaisancier puisse voir les informations en temps réel.
* Un panneau photovoltaïque pour fournir une énergie solaire et rendre le système autonome.
* Un régulateur de tension pour fournir une tension continue de 5 V au contrôleur et aux composants.
* Une batterie Li-ion pour que le système embarqué soit autonome sur une plus longue durée.

Durant la phase de développement, il sera éventuellement possible d’ajouter des modules selon les contraintes rencontrées non prévues en phase d’analyse.

Analyse fonctionnelle du système .

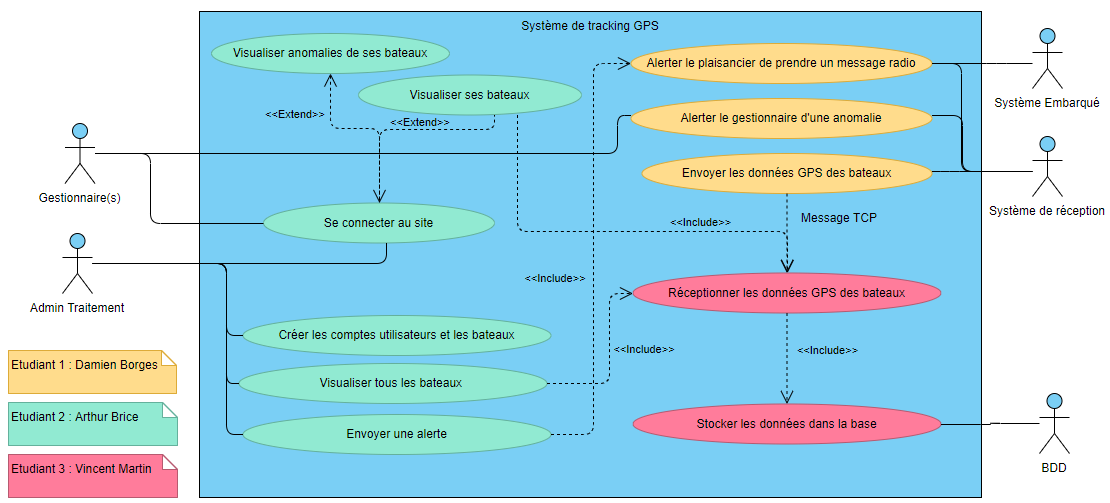
Dans cette partie, nous allons présenter l’analyse qui précède la phase de conception. Durant cette phase du projet, les échanges avec le client sont réguliers.

Certains points pourront donc être amenés à changer pour sécuriser davantage le système ou contourner des contraintes non prévues durant l’analyse.

Diagramme de cas d’utilisation

Pour la réalisation de tous les diagrammes, nous avons utilisé l’application web suivante :

[*https://online.visual-paradigm.com*](https://online.visual-paradigm.com).

*Les différentes bulles représentent les fonctionnalités que propose le système.*

On peut voir qu’il y a deux types d’utilisateurs les gestionnaires qui ne peuvent être en charge que de leur(s) bateau(x) et les administrateurs qui peuvent voir tous les bateaux disponibles et gérer tous les comptes des gestionnaires (modification mot de passe ou identifiant, création de nouveaux comptes, suppression de comptes).

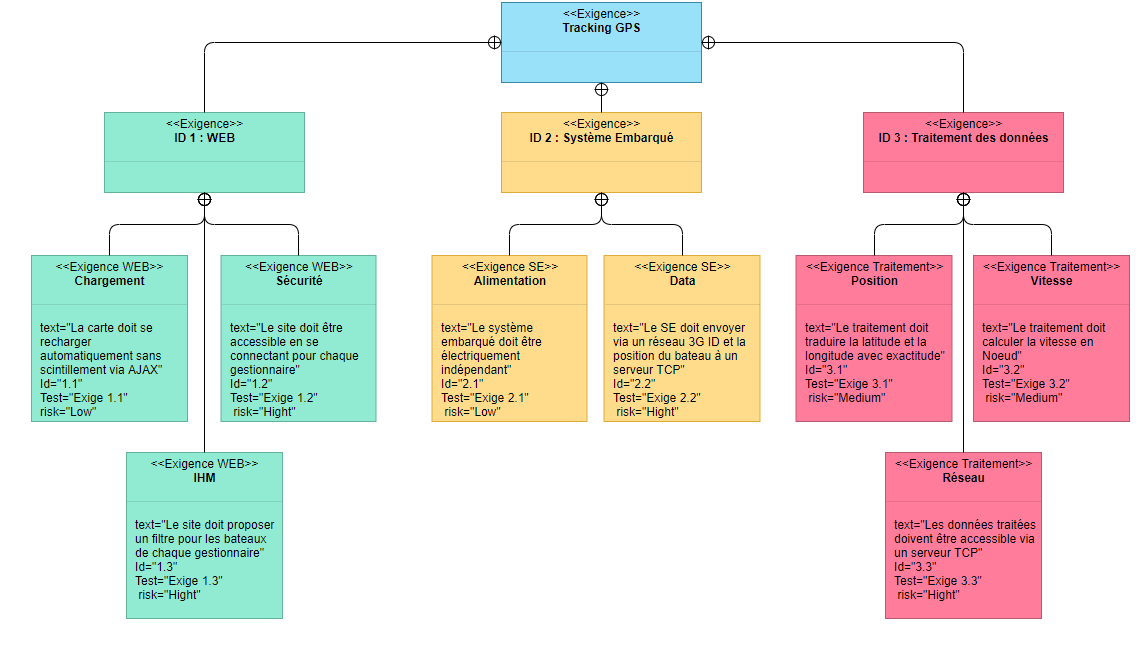
Le gestionnaire peut agir sur le système seulement de deux manières. La première est l’envoi d’une anomalie en sélectionnant un de ses bateaux. Le second est de pouvoir via une IHM visualiser l’état de tous ses bateaux sur une carte.

L’administrateur peut agir sur le système de trois façons exclusivement. La première est l’envoi d’une anomalie en sélectionnant n’importe quel bateau sur une carte. Le second est de pouvoir via une IHM visionner l’état de tous les bateaux sur une carte. Le troisième est d’avoir la possibilité de modifier, supprimer ou créer un compte gestionnaire.

En ce qui concerne le système embarqué, il est capable de recevoir, de détecter et de prévenir le gestionnaire et le plaisancier automatiquement en cas d’anomalie. Enfin, il transfère en temps réel toutes les données qu’il détecte à l’aide de ses capteurs.

Diagramme d’exigence

Le diagramme des exigences qui suit va nous indiquer plus précisément avec des règles le fonctionnement attendu du système.



Durant la phase de programmation, il faudra bien faire attention à bien respecter les exigences et procéder à un test de conformité pour chacune d’entre elles.

Diagramme de séquence

Les diagrammes de séquences sont la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation Unified Modeling Language (UML).

Le diagramme de séquence permet de montrer les interactions d'objets dans le cadre d'un scénario d'un Diagramme des cas d’utilisation. Dans un souci de simplification, on représente l'acteur principal à gauche du diagramme, et les acteurs secondaires éventuels à droite du système. Le but étant de décrire comment se déroulent les actions entre les acteurs ou objets.

La dimension verticale du diagramme représente le temps, permettant de visualiser l'enchaînement des actions dans le temps, et de spécifier la naissance et la mort d'objets. Les périodes d'activité des objets sont symbolisées par des rectangles, et ces objets dialoguent à l'aide de messages.

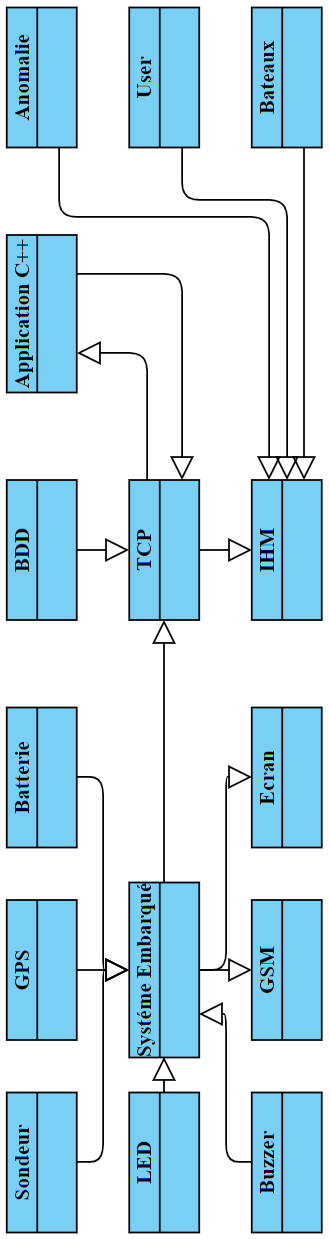
Diagramme de Classe

Nous allons commencer par présenter le diagramme de classe. Les classes ont été sélectionnées en décomposant toutes les entités du projet.

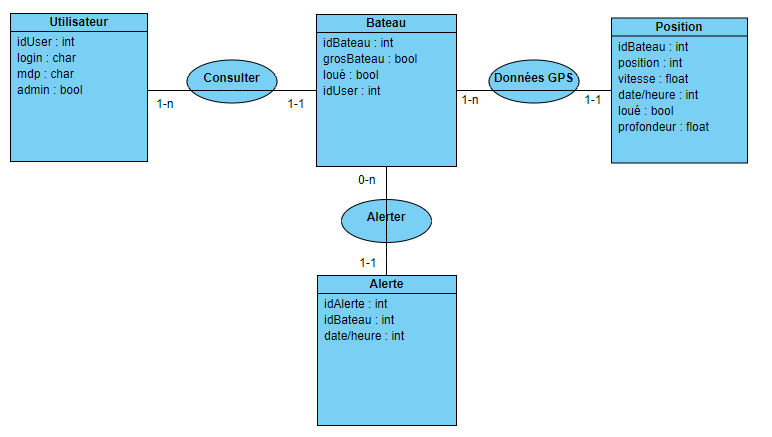
Une fois les classes identifiées, nous allons présenter le diagramme de séquence de chaque fonctionnalité du système (use case, diagramme de cas d’utilisation).

Comme nous sommes dans la partie commune, nous n’allons pas présenter les méthodes et propriété de ses classes. Ceci afin d’avoir une vision globale du système a programmé plus clair.

|  |  |
| --- | --- |
|  | Pour la partie Système Embarqué : la class principale est *Système Embarqué*.  Elle sera composée de 7 class :  - les class *LED* et *Buzzer* servent à visualiser une anomalie.  - la class *Ecran* permet de voir les différents messages reçus.  - la class *Batterie* permet de renseigner sur l’état de la batterie du système embarqué.  - la class *Sondeur* renseigne sur la profondeur.  - la class *GPS* renseigne sur la position.  - la class *GSM* envoie les données au centre. |
| C:\Users\Vincent\Desktop\g.PNG | Pour la partie IHM Central : la classe principale est *IHM*.  Elle sera composée de 3 class :  - une class *Anomalie* pour recevoir et envoyer un défaut au système embarqué.  -une class *User* qui va gérer la partie administration du central.  - une class *Bateaux* qui permet de voir l’historique de celui-ci et de savoir à qui il appartient. |
| C:\Users\Vincent\Desktop\test.PNG | Pour la partie C++ Centre de Traitement : la classe principale est *TCP*.  Elle sera composée de 4 class :  - une class *Système Embarqué* pour la réception de la trame.  - une class *Application C++* pour convertir les données et les envoyer au site web et à la BDD via TCP/IP.  - une class *BDD* pour garder en historisation les données des différents bateaux.  - une class *IHM WEB* pour envoyer en direct la position des bateaux sur une carte dynamique. |



Modèle conceptuel des données (MCD)



Les différents rectangles représentent les tables de la base de données accompagnées de leurs paramètres.

On peut voir que notre base de données est composée de 4 tables avec une utilisée pour les utilisateurs, les bateaux, les données des bateaux et les alertes.

Cette composition va nous permettre de mettre en place toutes les fonctionnalités demandées (connexion / inscription utilisateur, historisation des bateaux, etc.).

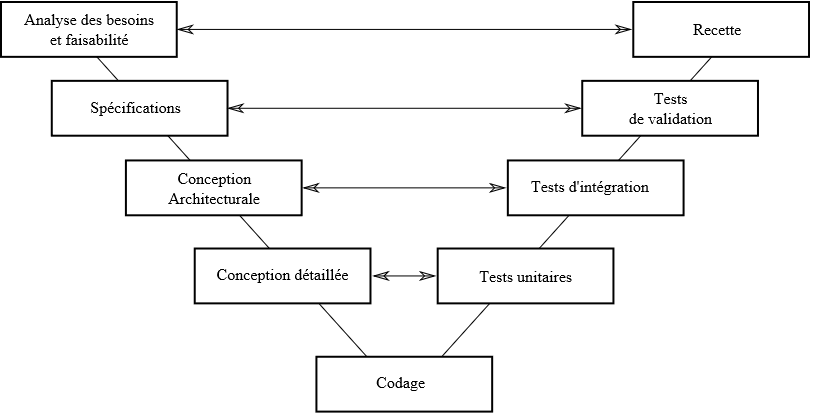
Organisation du projet .

Maintenant que le sujet est correctement appréhendé, il est nécessaire d’organiser le temps de travail jusqu’à la date butoir.

L’objectif est d’avoir un plan d’action établie afin de savoir si le projet commence à prendre du retard ou non. Pour la réalisation de ce projet, nous nous sommes organisés ainsi :

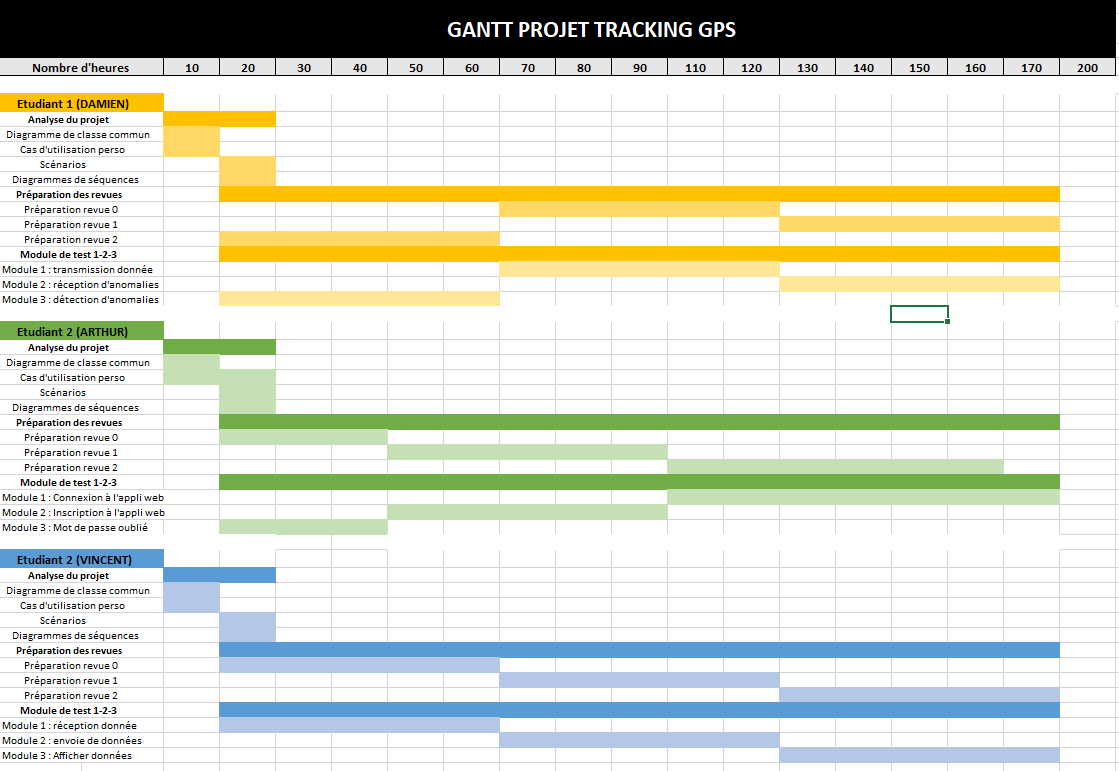
* Analyse.
* Préparation.
* Conception.
* Module de test.
* Intégration.

Ce découpage correspond au modèle de projet en cycle V :

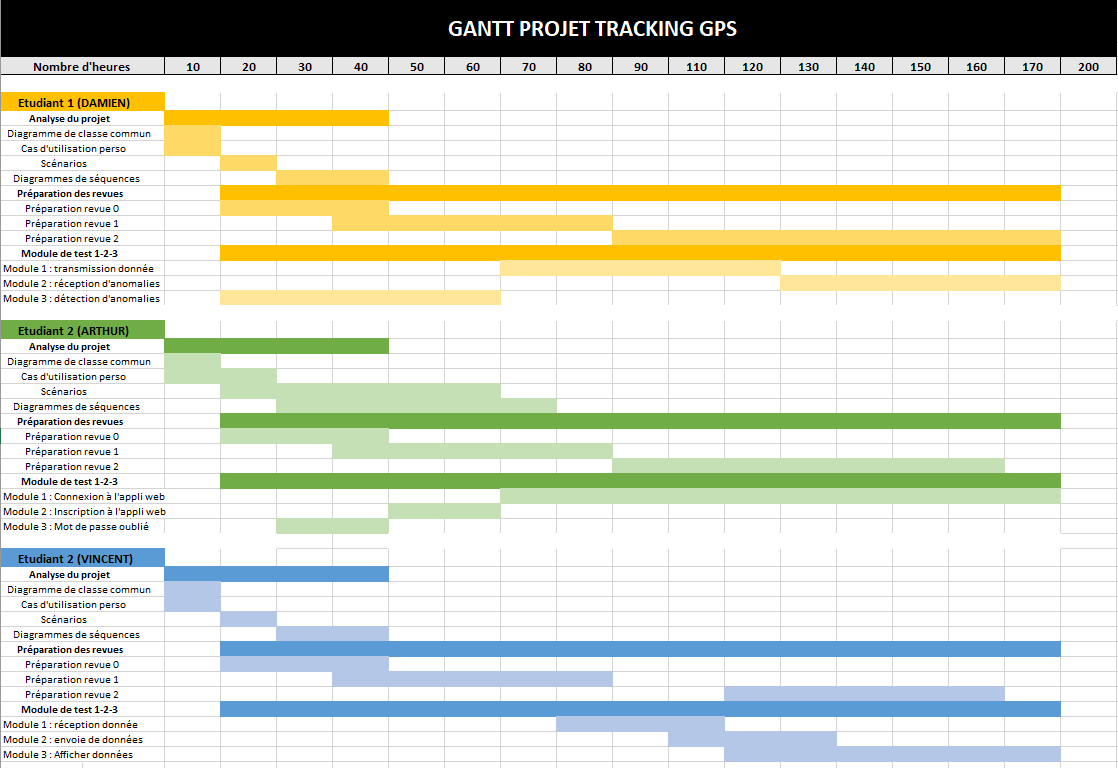


Étant un groupe de 4 étudiants sur ce projet, nous devons travailler en équipe. Nous allons donc détailler notre planning prévisionnel ainsi que le final pour voir comment nous avons réellement travaillé par rapport à nos prédictions.

GANTT prévisionnel



GANTT réel



Répartition des tâches

Voici la répartition des tâches qui nous est imposée par le sujet de BTS SN :

**Étudiant 1 : BORGES Damien en charge du système embarqué autonome**

* Analyse du projet et de sa partie.
* Étude comparative des composantes du système embarqué.
* Mise en place d’une application de simulation d’envoi de données pour l’étudiant 3.
* Choix des composants du système embarqué.
* Réalisation d’un prototype fonctionnel.
* Utilisation d’une classe C++ pour l’envoi des données.

**Étudiant 2 : ARTHUR Brice en charge de l’IHM web de supervision dynamique**

* Analyse du projet et de sa partie.
* Création du site web de supervision avec accès protégé par mot de passe.
* Gestion du Back Office (ajout, modification, suppression des utilisateurs et bateaux).
* Étude d’une API de cartographie web open source.
* Mise en place d’un serveur TCP pour récupérer les informations des bateaux.
* Affichage des bateaux du système en temps réel sur une page de supervision.
* Création de 3 classes PHP (User et BDD et TCP).
* Création d’une page anomalie (déplacement des bateaux sans être loué + localisation hors zone).

**Étudiant 3 : MARTIN Vincent en charge du centre de traitement**

* Analyse du projet et de sa partie.
* Utilisation d’un système de réception des informations mobiles.
* Application C++ qui récupère les informations des systèmes embarqués.
* Traitement des informations (découpage de trame, analyse de vitesse…).
* Réalisation d’un service C++ Linux qui envoie en TCP les informations au site web.
* Mise en place d’une fonctionnalité d’historisation des données des bateaux.
* Création des classes C++ (BDD, TCP, Système embarqué).
* Dois proposer en premier un simulateur TCP d’envoi de données pour l’étudiant 2.

Organisation de l’équipe .

Pour communiquer en dehors des heures de cours, nous utilisons l’application :

Messenger : cette application nous permet de communiquer ensemble grâce à une création de groupe afin de partager nos idées et de savoir où nous en sommes dans le projet.

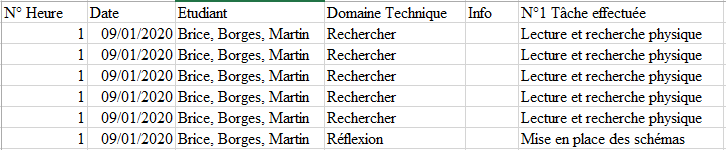
Git Hub : qui nous permet de mettre les différentes parties de notre projet, code, diagramme et dossier en créant différents dossier afin d’avoir une trace et de pourvoir récupérer ce que les autres font pour avancer sur nos parties, par exemple le code.

Drop Box : c’est ici que nous sauvegardons nos différents dossiers qui ne sont plus d’actualité et les dossiers récent afin d’avoir une trace de l’évolution de notre projet.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Git Hub | Messenger | Drop Box |

Compte rendu d’activité (CRA)

Pour avoir, un suivit de notre activité, nous avons réalisé un CRA horaire sous Excel.



Aujourd’hui nous pouvons remarquer que nous n’avons pu respecter notre CRA que nous avions espéré.

Cela dû au fait du climat actuel, face au coronavirus. Nous n’avons pu nous revoir pour mettre en commun nos travaux, et remédier aux différents problèmes rencontrés.

Chaque partie individuelle a pu être finie a presque 100 % de ce que nous avons espéré, elles fonctionnent indépendamment les unes des autres, mais nous ne pourrons jamais finir 100 % notre projet.

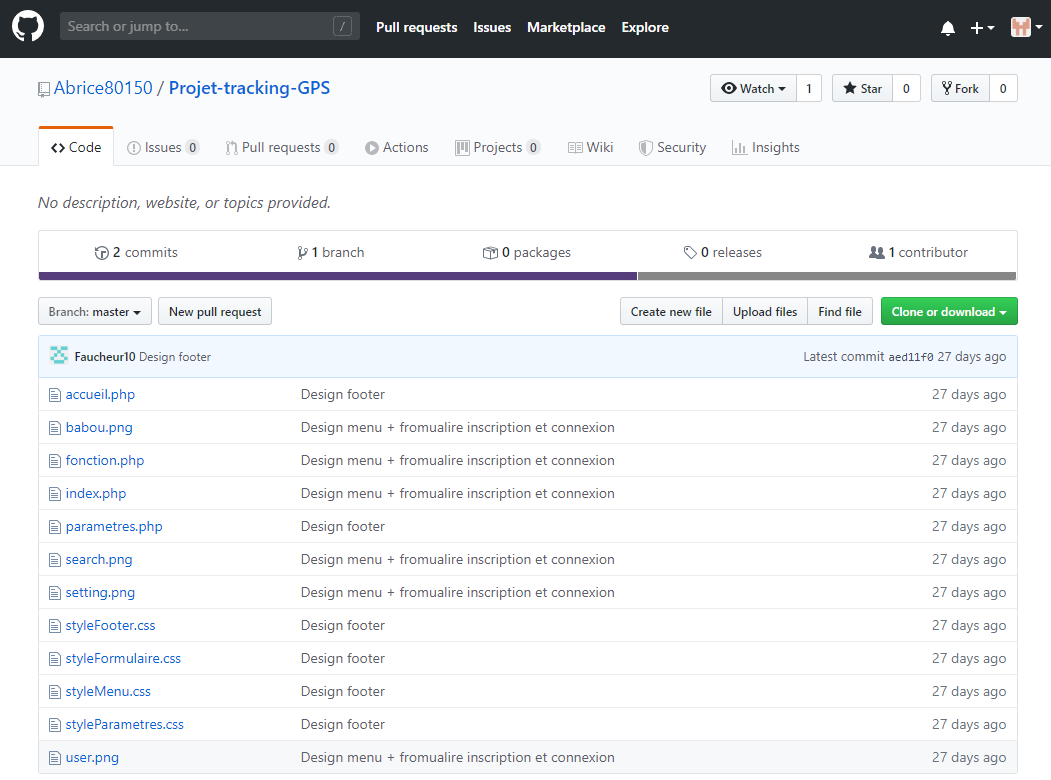
De ce fait, le CRA ne peut être réellement suivi par rapport à nos attentes.

|  |
| --- |
|  |
|  |
|  |

Cahier de bord

Tous les jours, nous tenons à jour un cahier de bord dans lequel sont détaillés les différents travaux réalisés durant les heures de projet. Ce cahier de bord permet de faire une synthèse de travaux qui sera présentée dans les parties individuelles du rapport de projet.

Git Hub et versionning



Pour faciliter le travail en collaboration, nous avons utilisé le logiciel de versionning Git ainsi que la plateforme d’hébergement [*https://Git-hub.com*](https://Git-hub.com).

Sur nos PC de développement, nous avons notre propre version de code source avec nos différents « Commit » une fois qu’une fonctionnalité est opérationnelle, nous la publions sur le site d’hébergement « Push » pour que tous les membres du projet puissent avoir accès à cette nouvelle fonctionnalité.

En arrivant sur l’hébergeur, il y a toujours la version la plus à jour de notre projet. Il est très facile de récupérer une version antérieure en cas de problème. En début de projet, nous avons créé toutes nos classes et méthodes qui ont été utilisées dans nos diagrammes de séquence.

Démarrage de projet et classe de simulation

Durant l’analyse toutes les méthodes des classes ne sont pas implémentées, elles le seront progressivement à mesure que le projet avance.

Les méthodes sont donc vides, mais retournent une valeur attendue simulée. Ainsi, un développeur peut utiliser une classe non implémentée en mode simulation. Les méthodes seront par la suite implémentées et « comité » sur le projet sans impacter éventuellement celui qui l’utilise.

Voici un exemple de classe simulée :

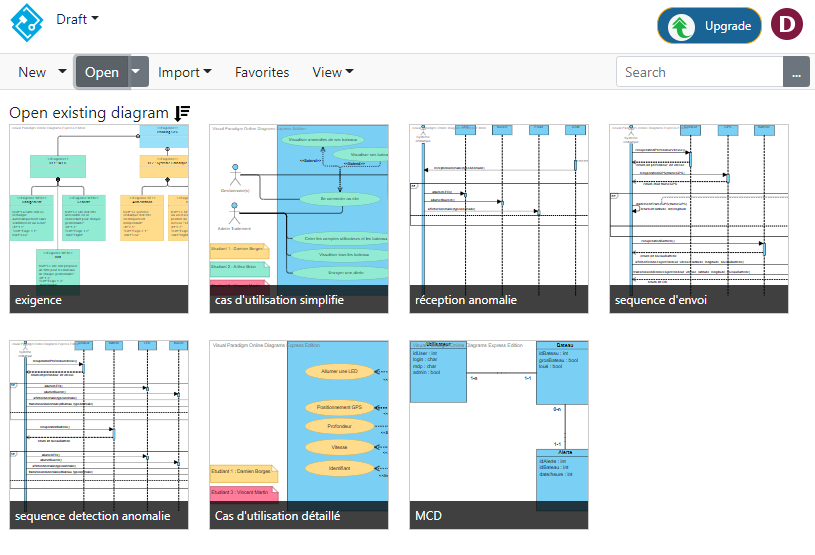
|  |  |
| --- | --- |
|  | Ceci est une partie de la class BDD.  Elle permet de se connecter à la base de données du site, d’enregistrer les différentes informations, de se créer un compte et de se connecter à son propre compte.  Elle est appelée par exemple dans la page index.php. |

|  |
| --- |
|  |
| Ceci est l’endroit où la class BDD est appelé afin de pouvoir se connecter à son compte et à la Base de Données. |

Logiciel d’analyse et de développement

Pour réaliser tous les diagrammes, nous avons utilisé l’outil en ligne suivant :

[*https://online.visual-paradigm.com*](https://online.visual-paradigm.com).



Cette application en ligne permet de sauvegarder nos diagrammes sur n’importe quelle machine en mode édition.

L’avantage, c’est que nous n’avons pas besoin d’installer une grosse application, l’espace est gratuit et suffisant pour nos besoins.

Maquette et prototype

Pour le prototype, nous avons utilisé une carte programmable C++ Arduino.

La passerelle TCP est une application C++ développée pour s’exécuter sur un processeur embarqué.

On utilise des VM à l’aide de Virtual Box pour nos serveurs web, de base de données et l’application C++.

Système embarqué autonome

|  |  |
| --- | --- |
| C:\Users\Damien Borges\Downloads\IMG_20200312_084729.jpg | C:\Users\Damien Borges\Downloads\IMG_20200312_084837.jpg |
| *Lorsqu’aucune anomalie n’est détectée* | *Lorsqu’une anomalie est détectée* |

Pour le prototype actuel, le led et le buzzer font office d’alarme et les différents capteurs (profondeur, vitesse, niveau de batterie) sont simulés par 3 potentiomètres.

Lorsqu’un potentiomètre est tourné, l’affichage de sa valeur change en temps réel sur l’écran.

L’alarme se met en marche lorsqu’une des valeurs a dépassé son seuil, et cela, jusqu’à qu’aucune anomalie ne soit détectée.

Partie Centre de Traitement (Communication TCP/IP)

Du côté de la partie du Centre De traitement, le récepteur est une application codée en C++ qui se trouve sur une VM.

Celle-ci permet de faire la passerelle entre le système embarqué et la partie IHM qui réunit la BDD et le Site WEB.

Partie Centre de Traitement (Communication TCP/IP)

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 simulé pour le prototype sur une VM de type Debian. Avec Apache et Maria DB de configurer.

Choix technique et étude physique .

Dans cette partie, nous allons uniquement présenter les solutions qui ont été retenues. Les études techniques plus approfondies présentant d’autres solutions seront détaillées dans les parties individuelles.

Choix de la carte pour le système embarqué

|  |  |
| --- | --- |
| Résultat de recherche d'images pour "arduino uno" | Pour le prototype, nous choisissons une carte programmable Arduino UNO.  Son prix (19,50 € TTC), ses nombreuses entrées/sorties de type numérique/analogique et sa facilité de programmation pourront grandement faciliter la phase de prototypage. |

Choix des capteurs et modules

|  |  |  |
| --- | --- | --- |
| **Aperçu** | **Description** | **Prix** |
|  | Shield GPS/GPRS/GSM TEL0097 :  **Module SIM808 disposant des fonctions GSM, GPRS et GPS** | 43,80 € |
| MCL053SRT - LED, Rouge, Traversant, T-1 3/4 (5mm), 20 mA, 1.85 V, 645 nm | LED rouge :  **MCL05SRT** | Gratuit |
| KPEG-350 - Transducteur, Piezo, Buzzer, Buzzer, Continue, 3 V, 28 V, 7 mA, 85 dB | Buzzer :  **KPEG-350** | Gratuit |
| Résultat de recherche d'images pour "ecran OLED 0.96 seeed" | Afficheur OLED 1,3' :  **Module afficheur monochrome OLED 1,3'' I2C TF051, 128 x 64 bits** | 10,90 € |
| Résultat de recherche d'images pour "breadboard" | Platine d’essai :  **Platine d’essai 400 points** | Gratuit |
| Résultat de recherche d'images pour "module de gestion de charge solaire Arduino" | Module de gestion de charge :  **Module de gestion de charge solaire Arduino** | 14,28 € |
|  | Accu Li-Ion :  **Accu Li-Ion 3,7V 1050 mAh PR474446** | 10,90 € |
| https://france-chauffage-solaire.fr/753-large_default/module-photovoltaique-panneau-solaire-250-wc-pv.jpg | Panneau solaire :  **Module photovoltaïque solaire** | Gratuit |
|  | **TOTAL :** | **79,88 €** |

Choix des moyens de communication

Les liaisons de communication qui sont à étudier dans notre projet sont les suivantes :

* Liaison système embarqué 🡨🡪 serveur TCP.
* Liaison serveur TCP 🡨🡪 BDD.
* Liaison serveur TCP 🡨🡪 IHM web.

Liaison Système Embarquée 🡨🡪 TCP



Centre de Traitement

Système Embarqué

Pour que le système embarqué puisse envoyer les données GPS au serveur TCP, il faut que :

- Le serveur peut recevoir les informations depuis le Système Embarqué, qu’ils puissent s’écouter et se parler.

- Que les deux systèmes soient tout le temps alimentés, pour éviter les pertes de données.

- Que la couverture réseau soit valable sur tout le territoire.

Nous avons donc retenue le réseau 3 G pour faire la communication entre les deux systèmes.

Du fait qu’il couvre presque tout le territoire français, qu’il soit assez simple d’utilisation et qu’il ne demande pas beaucoup de matériel pour l’utiliser.

Liaison TCP 🡨🡪 BDD / Liaison TCP 🡨🡪 IHM Web

IHH WEB

BDD

Centre de Traitement

Pour la communication entre l’application C++ et la BDD, nous devons utiliser la liaison TCP/IP, car l’application C++ et la BDD ne se trouvent pas au même endroit de même pour l’IHM Web. En effet, l’installation doit se faire sur tout le territoire français.

Plusieurs technologies filaires existent : Ethernet, le coaxial, l’USB, la fibre optique. Nous avons retenu l’Ethernet. Disponible sur presque 100 % du territoire français adapté aux cartes Arduino et Raspberry.

Ethernet est sécurisé si nous utilisons les bons protocoles de ce fait, nous enverrons les données en protocole TCP/IP afin de les sécuriser.

Recette .

Pour valider entièrement la recette client, nous devons reprendre toutes les fonctionnalités système attendues. Elles seront validées par un test d’intégration dans chaque partie individuelle, nous avons une recette des fonctionnalités détaillées qui seront validées par des tests unitaires.

|  |  |  |
| --- | --- | --- |
| **Fonctionnalités du système**  **+ Nom du test pour la validation** | **ETAT**  **(OK/NOK)** | **Commentaires** |
| Transmission des données par le système embarqué  *+testEnvoieDonnées* | OK |  |
| Réception d’une anomalie par le système embarqué  *+testRéceptionAnomalie* | OK |  |
| Détection automatique d’une anomalie par le système embarqué  *+testDétectionAnomalie* | OK |  |
| Connexion d’un utilisateur sur le site  *+testConnection* |  |  |
| Inscription d’un utilisateur sur le site  *+testInscription* |  |  |
| Affichage dynamique des bateaux sur une carte en temps réel  *+testAfficahge* | NOK | Impossible d’afficher en direct la position, aucune donnée n’est envoyée au site Web. |
| Affichage de l’historique des bateaux  *+testHistoriqueBateaux* | NOK | Impossible d’afficher l’historique des bateaux vu qu’aucune donnée est présent dans la BDD. |
| Modification par un administrateur d’un identifiant ou d’un mot de passe gestionnaire  *+testModification* |  |  |
| Création par un administrateur d’un nouveau gestionnaire  *+testCréation* |  |  |
| Suppression par un administrateur d’un gestionnaire  *+testSuppression* |  |  |
| Gestionnaire envoi manuellement depuis le site une anomalie à un bateau  *+tetsEnvoieAnomalie* |  |  |
| Récupérer les données des bateaux  *+tetsRéceptionDonnées* | OK |  |
| Découpage des trames  *+testDécoupageTrames* | OK |  |
| Envoyer les données en TCP au site  *+testEnvoieDonnéesTCP* | NOK | Les problèmes liés au code et au climat ne permet pas de réaliser à 100% ce test. |
| Enregistrement des données en BDD  *+testEnregistrerDonnéesBDD* | NOK | Problème de création dans la BDD qui ne permet pas la réception des données. |

Tests d’intégration du prototype .

Voici le cahier de test d’intégration qui sera validé dans chaque partie individuelle par des tests unitaires.

|  |  |  |
| --- | --- | --- |
| **Nom du test** | **Détail du test** | **Résultat du test** |
| Transmission données | Lorsque vous êtes sur la page d’accueil cliquer sur l’onglet historique puis actualiser la page et vérifier que les données soient historiées pour valider la bonne transmission de celles-ci. |  |
| Réception anomalie | Lorsque vous êtes connecté cliquer sur l’onglet anomalie dans le menu et envoyer une anomalie à un bateau. Dès que l’action est confirmée retourner sur la page d’accueil et cliquer sur l’onglet historique dans le menu puis actualiser cette page et vérifier que l’anomalie envoyée soit historiée pour valider la bonne réception de celle-ci. |  |
| Détection anomalie | - Baisser manuellement la profondeur à moins de 2 m et vérifier si le système embarqué détecte une anomalie de profondeur.  - Baisser manuellement la vitesse en dessous de 12km/h soit 6,5 nœuds et vérifier si le système embarqué détecte une anomalie de vitesse.  - Débrancher la batterie pour simuler un niveau en dessous de 10 % et vérifier si le système embarqué détecte une anomalie de batterie. |  |
| Connexion | - Entrer dans le formulaire de connexion un identifiant et un mot de passe juste et vérifier si cela vous donne accès au site.  - Entrer dans le formulaire de connexion de faux identifiant et mot de passe et vérifier si cela ne vous donne pas accès au site. |  |
| Inscription | Accéder au formulaire d’inscription et effectuer une inscription en suivant les démarches. Dès que l’inscription est confirmée, connectez-vous à l’aide de votre nouveau compte avec le formulaire de connexion et vérifier si vous avez accès au site. |  |
| Afficher  bateaux | Lorsque vous êtes sur la page d’accueil du site vérifié si les positions des bateaux sur la carte s’actualisent en temps réel. |  |
| Afficher historique bateaux | Lorsque vous êtes sur la page d’accueil cliquer sur l’onglet historique dans le menu et vérifier la présence d’un tableau réunissant toutes les données des bateaux en fonction d’une date/heure. |  |
| Modifier gestionnaire | Lorsque vous êtes sur la page d’accueil en tant qu’administrateur cliqué sur l’onglet back office dans le menu et modifier l’identifiant et le mot de passe d’un compte gestionnaire existant en remplissant le formulaire modification. Dès que l’action est confirmée vérifier la modification des informations dans le tableau récapitulatif des gestionnaires. |  |
| Créer gestionnaire | Lorsque vous êtes sur la page d’accueil en tant qu’administrateur cliqué sur l’onglet back office dans le menu et créer un nouveau compte gestionnaire en choisissant un identifiant et un mot de passe en remplissant le formulaire création. Dès que l’action est confirmée vérifier la création du nouveau compte dans le tableau récapitulatif des gestionnaires. |  |
| Supprimer gestionnaire | Lorsque vous êtes sur la page d’accueil en tant qu’administrateur cliquer sur l’onglet back office dans le menu et supprimer un compte gestionnaire existant en remplissant le formulaire suppression. Dès que l’action est confirmée vérifier la suppression du compte dans le tableau récapitulatif des gestionnaires. |  |
| Envoyer anomalie | Lorsque vous êtes sur la page d’accueil en tant qu’administrateur cliquer sur l’onglet anomalie dans le menu et sélectionner un bateau et le type d’anomalie a envoyé en remplissant le formulaire envoi anomalie. Dès que l’action est confirmée vérifier dans l’historique si le bateau en question a bien enregistré l’anomalie. |  |
| Réception Données | Démarrer le logiciel Embarcadero. Une fois, cela fait choisir le programme GPS. Cliquer sur le bouton exécuté le programme. Une page va s’ouvrir cliquer sur ouvrir port COM3, les trames reçues par l’application vont arriver. Elles seront cryptées, mais vous les verraient arriver en direct. |  |
| Découpage Trames | L’application C++ une fois lancée réceptionne les trames et les découpes automatiquement en faisant les conversions nécessaires. Les différentes données seront affichées dans un mémo. |  |
| Envoie Données TCP | L’application C++ envoie elle-même les données une fois convertir au site EB via une liaison TCP/IP avec les différentes données de la VM où se situe le site WEB. Ainsi, nous pouvons avoir la position des bateaux en direct. |  |
| Envoie Données BDD | Sur la page ouverte après avoir exécuté le programme, nous avons une page qui s’est ouverte dessus est écris purge BDD, l’application envoie directement les différentes informations de la trame en BDD comme cela, nous aurons toujours un historique des bateaux |  |

Avancement et conclusion .

Nous pouvons constater actuellement que l’analyse nous a retardés dans l’avancement du projet cependant cela nous a permis de mieux nous rendre compte de la complexité du projet.

Nous savons désormais que nous partons dans la bonne direction.

La préparation du projet afin de faciliter le travail en équipe a également pris du temps, mais nous commençons à bien maîtriser le versionning ainsi que le travail collaboratif.

Les délais d’attente pour recevoir les composants nous empêchent d’avancer sur certains points d’où l’obligation pour le moment de simuler.

Néanmoins la chaîne d’information circule correctement de bout en bout. La préparation de travail en groupe nous a permis de nous rendre compte des points sensibles à appréhender pour travailler dans de bonnes conditions (partages des classes, versionning et partage de l’analyse).

Ces choix ont été réalisés dans le but de contourner les contraintes rencontrées tout en respectant au mieux la demande du client.

Du au problème mondial du coronavirus, nous n’avons pas pu avancer comme nous le souhaitions avec les autres membres de ce projet, malgré ceci, nous avons essayé au maximum de faire fonctionner nos différents parties.

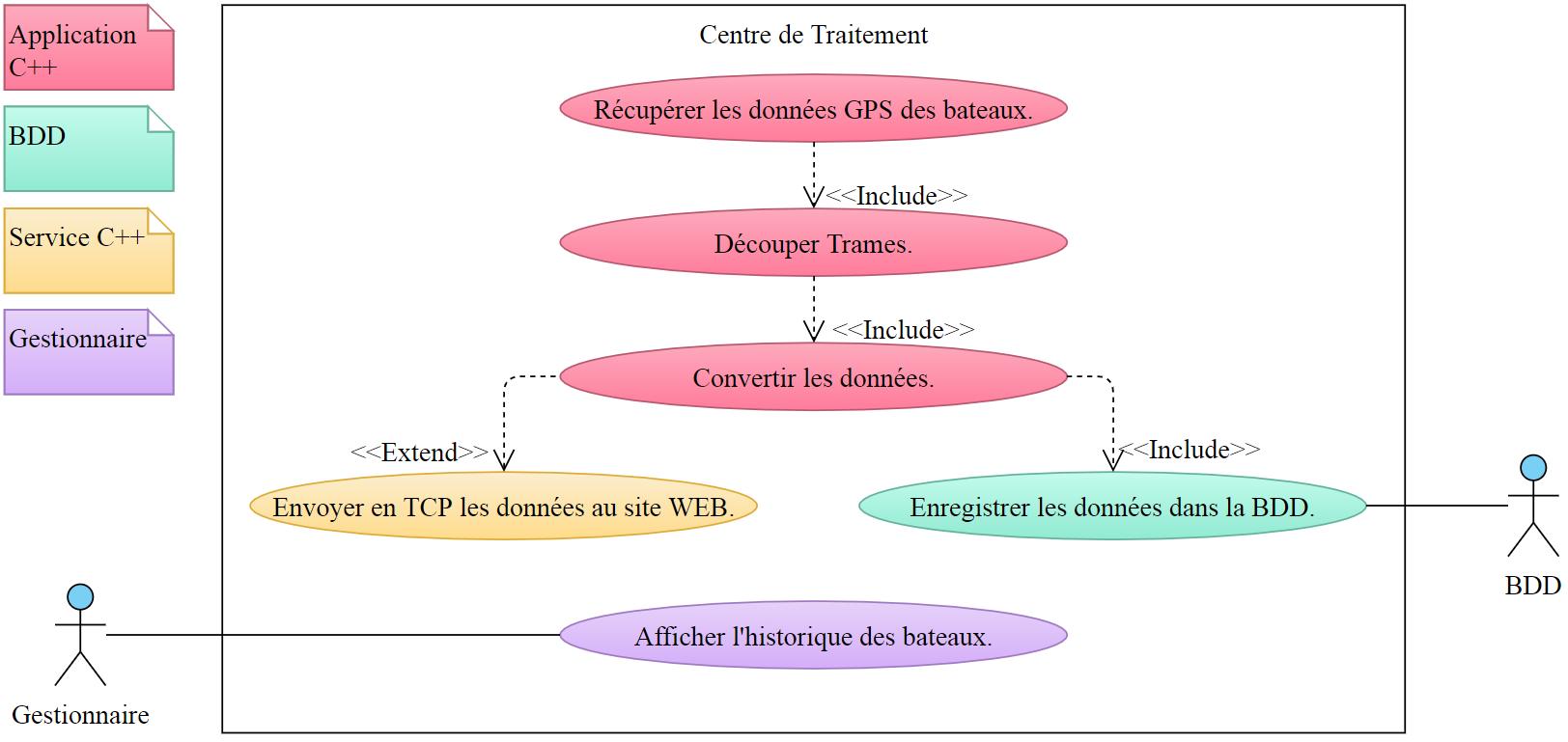
De ce fait, nos parties sont opérationnelles presque à 100 %, il faudrait que nous puisions les mettre en commun afin de régler les différents problèmes liés au code et à l'appel de données, pour que le projet soit mené à bien.

Nous pensons avoir mené à bien notre projet malgré le fait que nous sommes partis dans la mauvaise direction, que le matériel arrivant en retard et aussi du fait que nous avons dû continuer le projet à domicile.

Nous avons essayé de faire de notre maximum pour mener à bien ce projet, nous sommes satisfaits de notre travail effectué et du rapport que nous fournissons.

PARTIE 2 : Partie Individuel Martin Vincent .

Diagramme de cas d’utilisation détaillé .



*Les différentes bulles représentent les fonctionnalités que propose le système.*

Nous pouvons voir ici que lorsque le système embarqué envoie les différentes données du bateau par une liaison 3 G (donc TCP/IP), l’application C++ qui se trouve au centre de contrôle récupérer les données GPS des bateaux en vérifiant leur intégralité.

Une fois les données GPS récupérés et celles-ci complètes l’application les découpe (trames) afin d’en décrypter les différentes parties que nous impose le cahier des charges, puis elle les convertis afin que l’administrateur et le gestionnaire puissent les lire.

Par la suite, l’application C++ envoie en liaison TCP/IP les données à la Base De Données afin d’avoir un historique des déplacements des bateaux et envoie de même les données au site WEB afin d’avoir un suivit en direct des bateaux.

Ainsi, l’administrateur et le gestionnaire peuvent voir et afficher l’historique des bateaux à une date ultérieure.

Analyse Physique .

Définition du système de navigation GPS

Le GPS pour *Global Positioning System* est un système de positionnement par satellite.

À l'origine, le GPS a été conçu par le département de la Défense des États-Unis puis il a été mis à la disposition des applications civiles.

Le système repose sur 24 satellites qui tournent en permanence autour de la terre sur 6 orbites différentes. Ces satellites transmettent régulièrement leur position en orbite ainsi que des signaux horaires.

Au niveau terrestre, le récepteur GPS (petit boîtier électronique) échange des informations avec au minimum trois de ces satellites. Il mesure le temps de propagation des ondes et calcule la distance qui le sépare des satellites, ce qui lui permet de déterminer son positionnement spatial (longitude, latitude, altitude).

Ainsi, grâce au système GPS, vous pouvez connaître votre position n'importe où à la surface de la terre, en mer ou dans l'air. Le GPS permet également d'indiquer la vitesse de déplacement d'une personne ou d'un objet à chaque instant.

*I) L'antenne ou récepteur GPS :*

Il s'agit d'un petit boîtier contenant une puce GPS ou chipset. Le rôle de l'antenne est de réceptionner les signaux en provenance des 24 satellites GPS. C'est ce qui permet à l'appareil de se positionner en permanence.

La deuxième caractéristique d'un récepteur GPS après la puce, concerne son interface. On distingue plusieurs types de récepteur GPS suivant la connectivité avec le terminal mobile : (USB; Compact Flash /SD; Bluetooth; PS/2; GPS Marine).

*II) Le terminal mobile :*

Par terminal, on entend tout ce qui est PDA, téléphone mobile, Smart Phone ou PC portable. Le rôle du terminal est de traiter les signaux captés et transmis par le récepteur GPS puis de les restituer sous forme d'image en indiquant notamment la position du GPS sur une carte.

*III) GPS autonome :*

Lorsque l'appareil GPS intègre dans un même boîtier, le récepteur, le terminal et le logiciel, on parle alors de GPS autonome. Le principal intérêt de ce type de GPS réside dans sa simplicité d'utilisation et leur fiabilité.

*IV) Conclusion :*

Pour résumé, un appareil GPS est constitué d'un récepteur, d'un terminal (PDA, Smart Phone, téléphone mobile) et d'un logiciel de navigation (cartographie).

Principe de la triangulation

Le principe, découvert par Thalès (625 - 547 av. J - C) pour mesurer la distance qui séparait un bateau en mer de la côte, consiste à mesurer les angles entre deux points de référence dont on connaît la position et la distance qui les séparent, et le point dont on souhaite évaluer la distance.

On parle de triangulation puisque l'ensemble de ces angles et distances forme un triangle.

La triangulation permet grâce à trois satellites fixes qui connaissent leur position exacte de déterminer la position d’un point précis sur Terre. Chaque satellite du réseau GPS possède une horloge atomique d’une grande précision.

*Principe :*

À un instant *t* un satellite émet son signal vers le récepteur GPS qui le reçoit à un instant *t1*.

Le GPS calcule ensuite grâce à une opération simple *tl – t = T* qui est le temps qu’a mis le signal pour arriver.

Connaissant la vitesse du signal (300 000 km/h) le GPS, grâce à la relation *distance (d)= temps \* vitesse* ceci détermine une sphère de centre le satellite et de rayon *d* qui sont ses positions susceptibles.

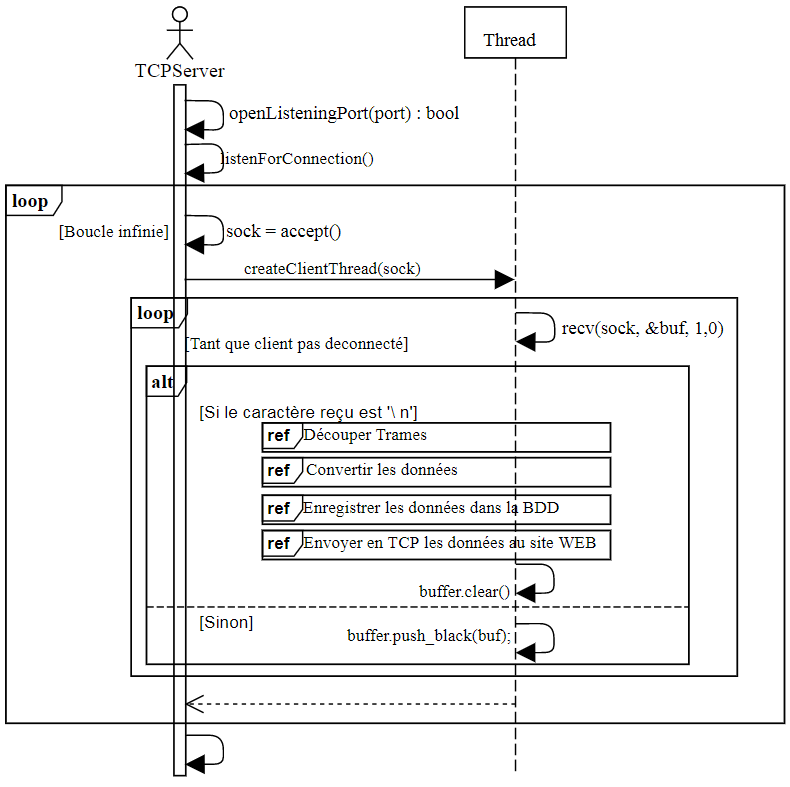
En renouvelant ce mécanisme avec un deuxième satellite, il détermine une deuxième sphère qui coupe la 1re en un plan sur lequel est situé le GPS.

Pour terminer, un troisième satellite est nécessaire afin de trouver sa position exacte sur ce plan. En effet, lorsque la troisième sphère coupe le plan, elle donne deux points.

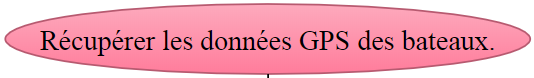
Dans le cas où l’utilisateur se situe à la surface de la Terre seul un des 2 points est cohérent. Ainsi, on peut déduire sa position exacte en éliminant le point donnant un résultat incohérent.

Module de test réception de la trame système embarqué .

Diagramme de séquence de réception de la trame

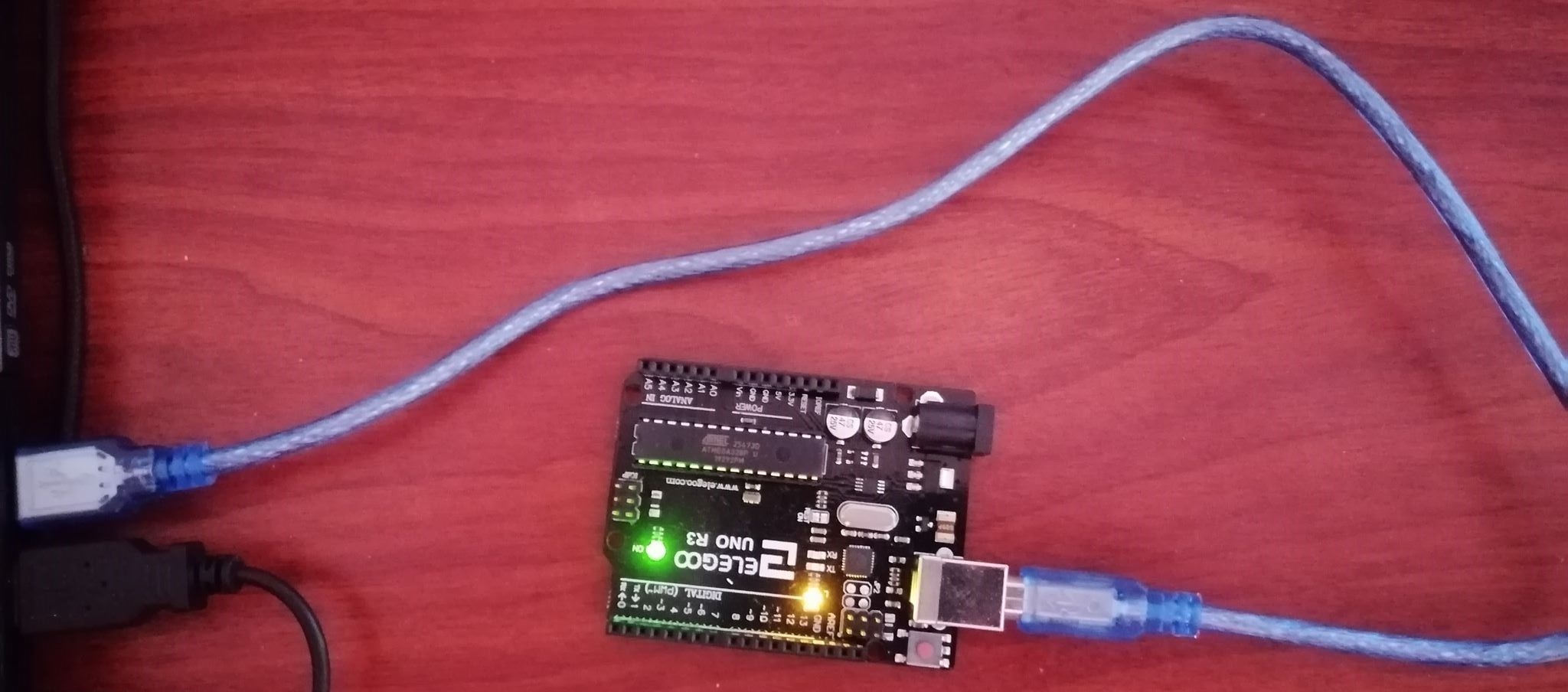


Ce module permet de mettre en place la fonctionnalité suivante :



En effet, le système embarqué envoie la trame des donnés GPS du bateau par liaison 3G grâce à un module GSM au centre de traitement, afin que l’application C++ puisse la récupérer.

Voici le câblage réalisé afin de simuler le système embarqué qui envoie les données GPS donc la trame à l’application C++ par liaison 3G :



Nous sommes dans l’obligation d’utiliser une carte ELEGOO UNO R3 afin de pouvoir émuler la réception de la trame GPS du système embraqué.

Pour cela, nous allons utiliser aussi l’application Arduino, ceci va permettre d’envoyer des informations sur la carte ELEGOO UNO R3, une fois les données envoyées à cette carte nous lançons l’application C++.

Une fois l’application C++ lancée, nous la connectons au port COM3 de la carte ELEGOO. La carte va envoyer à l’application C++ le code de qu’elle a gardée en mémoire quand nous lui avons envoyé la trame via l’application Arduino.

Pour que ceci fonctionne bien, il ne faut pas oublier de fermer l’application Arduino afin qu’il n’y ait pas de confrontation entre l’application Arduino et C++, car une seule application peut se connecter au port COM3 de la carte et non deux à la fois.

Ci-dessous une photo de la trame écrite sous Arduino et envoyée à la carte ELEGOO UNO R3 avec le port COM3 comme passerelle :

|  |  |
| --- | --- |
| Capture | Captyure |

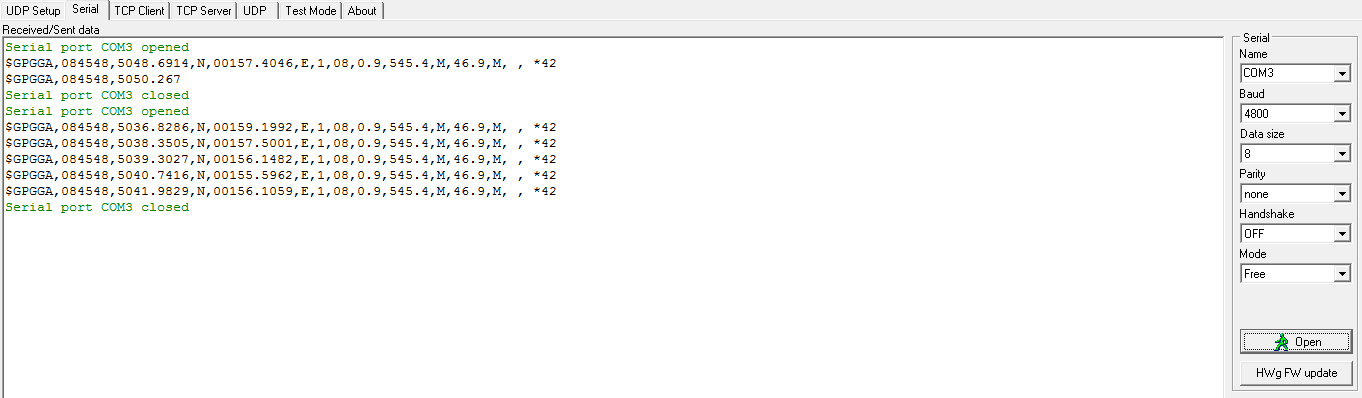
Une fois le code écrit nous le téléversons vers la carte ELEGOO UNO R3 si le téléversement se passe bien la carte vas clignoter pour nous prévenir de cela et l’application vas nous donner ce type de message :

vds

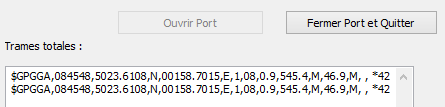
Afin de vérifier que les manipulations que nous avons effectuées fonctionnent nous allons d’abord tester avec Hercules la réception des données avant de faire ceci avec l’application C++.

Pour cela une fois l’application Arduino fermée et Hercules lancé nous sélectionnons la case Serial et nous remplissons les différentes informations que nous avons par rapport au code au-dessus.

Une fois, cela fait nous ouvrons le port et si tout fonctionne nous avons la réception de la trame, photo ci-dessous :



Vu que notre code fonctionne nous allons faire de même avec notre application C++, ce qui va nous donner le résultat suivant en photo :



Lorsque le module de test est en fonctionnement nous pouvons visualiser les différentes trames arrivées une fois le port COM3 ouvert.

Dans la trame comme sur la photo au-dessus nous avons comme informations la vitesse, la latitude, la longitude.

Lors de la réception, l’application C++ vérifie si la trame est complète si celle-ci ne l’est pas alors elle attend que le système embraqué lui envoie de nouveau une trame afin de pourvoir réaliser les différentes étapes pour avoir les différentes données en claires.

Scénarios : réception de la trame

L’application C++ reçoit une trame :

1- Le système attend la réception d’une trame.

2- Le système affiche la trame reçu.

Précondition :

- L’application doit être allumée.

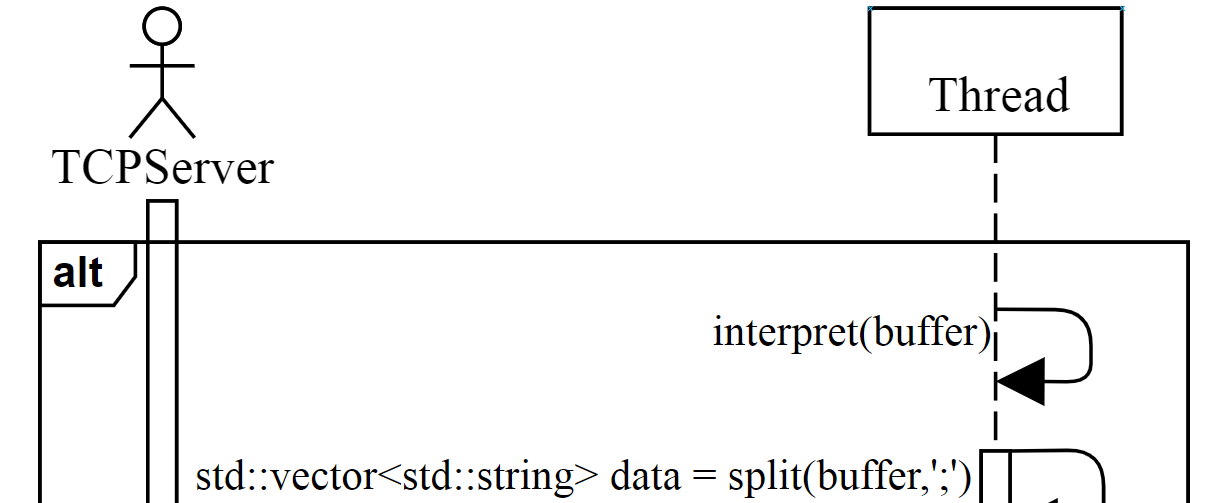
- L’application doit pourvoir communiqué avec le système embarqué via le réseau 3G.

(Si la trame n’est pas complète.).

- L’application attend une nouvelle trame jusqu’à ce que celle reçu soit complète.

Module de test découpé trame système embarqué .

Diagramme de séquence découper trame



Ce module permet de mettre en place la fonctionnalité suivante :

C:\Users\Vincent\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Découper_Trames.png

En effet une fois la trame réception depuis le système embarqué l’application C++ la découpe afin d’avoir les différentes parties bien distinctes les unes des autres

Nous sommes dans l’obligation d’utiliser une carte ELEGOO UNO R3 afin de pouvoir émuler la réception de la trame GPS du système embraqué.

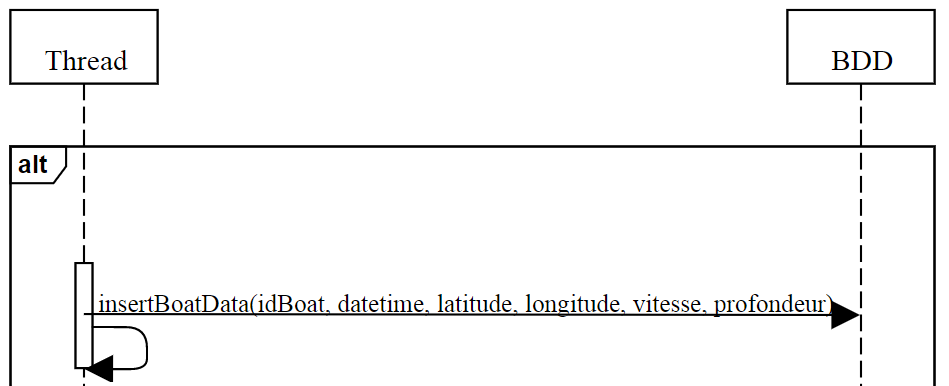
Pour cela nous allons utiliser aussi l’application Arduino, ceci vas permettre d’envoyer des informations sur la carte ELEGOO UNO R3, une fois les données envoyer à cette carte nous lançons l’application C++.

Une fois l’application C++ lancé nous la connectons au port COM3 de la carte ELEGOO, la carte va envoyer à l’application C++ qu’elle a gardé en mémoire quand nous lui avons envoyé la trame via l’application Arduino.

Pour que ceci fonctionne bien il ne faut pas oublier de fermé l’application Arduino afin qu’il n’y est pas de confrontation entre l’application Arduino et C++ car une seule application peut se connecter au port COM3 et non deux à la fois.

Module de test envoyer à la BDD trame système embarqué .

Diagramme de séquence envoyer trame à la BDD



Module de test envoyer au Site WEB trame système embarqué .

Diagramme de séquence envoyer trame au Site WEB

