# 1 - Analyse

## 1.1 Micro-ordinateurs

Dans le contexte du projet, il nous faut deux **micro-ordinateurs**, pour la gestion des deux capteurs de température et du capteur de géolocalisation. Ces micro-ordinateurs permettront de gérer le flux d’information reçu et l’enverra sur le serveur de l’entreprise.

Les Micro-ordinateurs devront respecter ces conditions :

* Doit posséder un **espace de stockage**.
* Doit pouvoir **se connecter en Wi-Fi**.
* Doit pouvoir **résister un minimum le froid**.
* Doit avoir des **pins des entrées et des sorties**.

Les deux propositions de micro-ordinateurs sont :

* **Raspberry PI3B+**
* **Arduino UNO Rev3**

Ces deux **micro-ordinateurs** sont connus dans leur milieu. Néanmoins, ils ne sont **pas faits pour les mêmes utilisations**, nous listerons leurs caractéristiques afin de choisir le plus **adapté**.

Vous trouverez ci-dessous le tableau comparatif entre le Raspberry PI3B+ et l’Arduino UNO Rev3 :

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Raspberry PI3B+** | **Arduino UNO Rev3** | **Meilleur choix** |
| Consommation électrique | 5V | 7-12V | Raspberry |
| Température de fonctionnement | 0-50°C | Non communiqué | Raspberry |
| Ports entrées/sorties | 40 pins | 12 pins | Raspberry |
| Langage de programmation | Python | C | Raspberry |
| Mémoire vive | 1GB | 2 kB | Raspberry |
| Emplacement carte SD | Oui | Non | Raspberry |
| Connexion Wi-Fi | Oui | Non | Raspberry |
| Prix | ≃ 40€ | ≃ 20€ | Arduino |

Notre choix final sera donc, le **Raspberry PI3B+**. Il nous faudra quatre Raspberry, un pour chaque véhicule de livraison, plus undans la chambre froide.

Pour le fonctionnement d’une carte Raspberry, il faut :

* Un **Système d’alimentation 5V** environ
* Une **carte micro SD** avec un **système d’exploitation flashé**

**Pour la simplification du projet, nous allons installer et programmer une seule carte (celles qui seront dans les véhicules). Ensuite, nous allons la cloner pour les deux autres véhicules.**

Sonde de Température

Nous avons besoin d’utiliser des **Sondes de température** afin de surveiller la Température d’une chambre froide ainsi qu'à l’intérieur des véhicules de livraison dans le but d’effectuer une traçabilité du froid des véhicules et de la chambre.

Les Sondes Que nous aurons à choisir devront respecter ces conditions :

* Doit pouvoir être **connectée à un Raspberry** directement ou **via GPIO.**
* Doit être la **moins encombrante** **possible** (plus petite, moins de câble).
* Doit être **peu onéreuse**.
* Doit pouvoir capter une plage de température comprise entre **0-5°C**

Les deux Sonde à nos dispositions sont les suivantes :

* **DS18B20**
* **DB130-10**

La **DS18B20** est une sonde assez connue dans le milieu des sondes de températures et est produite par l’entreprise **Maxim integrated**. En opposition peu d’informations sont à disposition sur la **DB130-10**, nous utiliserons les caractéristiques de celles produites par l’entreprise **DFROBOT** pour nos comparatifs.

Veuillez trouver ci-dessous un comparatif des caractéristiques de chacune des sondes ainsi que celle que nous avons choisi pour ce projet :

|  |  |  |  |
| --- | --- | --- | --- |
|  | **DS18B20** | **DB130-10** | **Meilleur Choix** |
| **Consommation électrique** | 3V à 5,5V | 5V | **DB130-10** |
| **Type d’interface** | Câble unique | Interface 2 câbles | **DB130-10** |
| **Plage de température** | -55°C à +125°C | -10°C à 80°C | **Indifférent** |
| **Taille** | 30mm x 6mm | 49mm x 14mm | **DS18B20** |
| **Longueur du Cable** | 91cm | 90 cm | **DB130-10** |
| **Prix** | 10€ | 27€ (29.50$) | **DS18B20** |

Notre Choix de Sonde final sera donc la **DB130-10**, il nous en faudra 4, une pour la chambre froide et une pour chaque Véhicule de Livraison.

Nous avons trouvé plus intéressant de **conserver la même sonde pour chaque cas d’utilisation.**

GPS

Nous serons amenées à récupérer des **informations GPS** pour reproduire le parcours d’un véhicule de livraison et récupérer toutes les informations souhaitées en fonction d’une position a un temps donné. Par conséquent des r**écepteurs GPS** sera nécessaire.

Les Récepteurs GPS Que nous aurons à choisir devront respecter ces conditions :

* Doit avoir une **faible consommation électrique.**
* Doit être la **moins encombrante possible** (plus petite).
* Doit être **peu onéreuse**.

Les deux Récepteurs GPS à nos dispositions sont les suivants :

* **BU-353**
* **Here 2**

Le récepteur GPS BU-353 est produit par **Globalsat**, une entreprise spécialisée dans les **GPS Tracker, récepteurs GPS...**

**Pixhawk 2** est un **projet** datant de **2008** qui vise à fournir du **matériel de contrôle de pilotage** (autopilote) **haut de gamme**, à **coûts bas ou raisonnables** à des communautés d'utilisateurs universitaires, de loisirs et industrielles.

Veuillez trouver ci-dessous un comparatif des caractéristiques de chacun des Récepteurs GPS ainsi que celui que nous avons choisi pour ce projet :

|  |  |  |  |
| --- | --- | --- | --- |
|  | **BU-353** | **SIM28** | **Meilleur Choix** |
| **Chipset GPS** | L1 SiRF Star III | UB-5010 | **SIM28** |
| **Interface** | USB | GPIO | **SIM28** |
| **Fréquence trames NMEA** | 1 Hz | Max 10 Hz | **SIM28** |
| **Récepteur 20 canaux** | L1, C/A code | L1, C/A, WAAS, EGNOS, MSAS, GAGAN | **SIM28** |
| **Dimensions** | 53mm\*19mm | 53mm\*47mm\*23mm | **SIM28** |
| **Consommation** | 80 mA | 35 mA | **SIM28** |
| **Précision** | 10m (2D RMS) | 5 m | **SIM28** |
| **Prix** | 39€ | 23€ | **SIM28** |

On choisira par conséquent Le **SIM28** car il est plus **économique** et **plus performant**.

Afficheur

Pour pouvoir afficher les données de la sonde chargée d’observer la température, nous aurons besoin d’un **afficheur** à l’entrée de la chambre froide nous donnant toutes les informations que l’on souhaite

Les différents **afficheurs** que nous aurons à choisir devront respecter ces conditions :

* Doit avoir une **faible consommation électrique.**
* Doit être la **moins encombrante possible** (plus petite).
* Doit être **peu onéreuse**.
* Doit être **tactile.**

Les deux afficheurs à nos dispositions sont les suivants :

* **Raspberry Pi 7’’ Touchscreen Display**
* **Raspad**

Le récepteur GPS BU-353 est produit par **Globalsat**, une entreprise spécialisée dans les **GPS Tracker, récepteurs Gps...**

**Pixhawk 2** est un **projet** datant de **2008** qui vise à fournir du **matériel de contrôle de pilotage** (autopilote) **haut de gamme**, à **coûts bas ou raisonnables** à des communautés d'utilisateurs universitaires, de loisirs et industrielles.

Veuillez trouver ci-dessous un comparatif des caractéristiques de chacun des afficheurs ainsi que celui que nous avons choisi pour ce projet :

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Touchscreen Display** | **Raspad** | **Meilleur Choix** |
| **Consommation** | Avec la Raspberry (3 à 5,5V) | 6000mAh | **Touchscreen Display** |
| **Dimension** | 194mm x 110mm x 20mm | 25,4 cm de diagonale | **Touchscreen Display** |
| **Tactile** | Oui | Oui | **Equivalent** |
| **Connectique** | DSI port | USB | **Touchscreen Display** |
| **Résolution** | 800\*480 pixels | 1280\*800 pixels | **Raspad** |
| **Fréquence** | 60Hz | 60Hz | **Equivalent** |
| **Prix** | 60€ | 200€ | **Touchscreen Display** |

On choisira par conséquent Le **Raspberry Pi 7’’ Touchscreen Display** car il est plus économique et **moins** **consommateur** en énergie. De plus sa **connectique** est plus adaptée dans notre contexte.

Langage De Programmation

Enfin la partie la plus importante à nos yeux, nous devons choisir un langage de programmation dans lequel

Les différents **afficheurs** que nous aurons à choisir devront respecter ces conditions :

* Doit être **facile** **d’utilisation**
* Doit être **dynamique (**pour le **temps réel)**

Les deux langages à notre disposition sont :

* **Python**
* **C++**

D’autres langages peuvent être utilisés mais ces deux-là sont les deux **meilleures options** sur un système comme une **Raspberry**

Veuillez trouver ci-dessous un comparatif des caractéristiques de chacun des afficheurs ainsi que celui que nous avons choisi pour ce projet :

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Python** | **C++** | **Meilleur Choix** |
| **Compilation** | Besoin d’un interpréteur | Pré-compilé avant exécution | **C++** |
| **Efficience** | Language Simplifié | Language traditionnel | **Python** |
| **Difficulté** | Facile | Difficile | **Python** |
| **Collection des déchets** | Oui | Non | **Python** |
| **Cross-platform** | Oui | Non | **Python** |
| **Langage Dynamique** | Oui | Non | **Python** |

On choisira par conséquent Le **Python**, car parmi les deux il est le seul langage dynamique, de plus il est beaucoup plus **simple** d’utilisation que le **C++.**

Alimentation de la Raspberry de livraison.

La Raspberry a besoin d’alimentation de 5V 2.5A en micro USB pour fonctionner.

Il faut que l’alimentation soit :

* Peu consommatrice en énergie (5V 2.5A).
* Que l’alimentation soit éteinte quand la voiture est à l’arrêt

Nous avons deux solutions qui sont à notre disposition :

* Un adaptateur allume-cigare à micro USB de 5V
* Régulateur 5V/12V

Veuillez trouver ci-dessous un comparatif des caractéristiques de chacun des afficheurs ainsi que celui que nous avons choisi pour ce projet :

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Adaptateur allume-cigare 5V** | **Régulateur 5V/12V** | **Meilleur Choix** |
| **Consommation** | Sortie : 5V/1A | Sortie : 5V/2.4A | **Régulateur 5V/12V** |
| **Longueur des câbles** | 1.20 m | 5 cm | **Régulateur 5V/12V** |
| **Difficulté de mise en place** | Plug-and-Play | Besoin d’une installation | **Adaptateur allume-cigare 5V** |
| **Courant qui passe plus quand la voiture et à l’arrêt** | Oui | Oui | **Equivalent** |
| **Protection de surintensité** | Non | Oui | **Régulateur 5V/12V** |
| **Prix** | 10.90€ | 13.90€ | **Adaptateur allume-cigare 5V** |

On choisira par conséquent **Régulateur 5V/12V**, car parmi les deux, c’est celui qui gardera un courant stable peu importe l’intensité de la voiture.

Symfony

Symfony est un framework français open source pour PHP. Les frameworks, en général, permettent un gain de productivité mais pas seulement. Ils permettent d'avoir une bonne organisation du code et on sait qu'un code bien organisé est plus maintenable et évolutif. Symfony est open source. On n'a donc pas de contraintes imposées et on peut développer des solutions propriétaires. Il est considéré comme un des framework PHP les plus puissants et les plus flexibles. Un des principaux inconvénients d'un framework est le temps d'apprentissage qui est bien plus élevé que pour un langage en lui-même. Symfony, plus particulièrement nécessite d'apprendre plusieurs autres technologies qui lui sont propres comme YAML ou Doctrine.

Une Doctrine ORM implémente 2 patterns objets pour mapper un objet PHP à des éléments d'un système de persistance :

* Le pattern "Data Mapper".
* Le pattern "Unit of Work".

Le Data Mapper est une couche qui synchronise la donnée stockée en base avec les objets PHP. En d'autres termes :

* Il peut insérer, mettre à jour des entrées en base de données à partir de données contenues dans les propriétés d'un objet ;
* Il peut supprimer des entrées en base de données si les "entités" liées sont identifiées pour être supprimées ;
* Il "hydrate" des objets en mémoire à partir d'informations contenues en base.

L'implémentation dans le projet Doctrine de ce Data Mapper s'appelle l'Entity Manager, les entités ne sont que de simples objets PHP mappés.

L’utilisation de Symfony, nous permettra de créer un serveur internet virtuel, pour notre phase de conception.

IDE

Nous avons deux choix pour l’IDE permettant de faire le site WEB en symfony. L’un c’est NetBeans, l’autre c’est Symfony. Voici les pour et les contre.

NetBeans est un environnement de développement intégré (EDI), placé en open source. En plus de Java, NetBeans permet la prise en charge native de divers langages tels le C, le C++, le JavaScript, le XML, le Groovy, le PHP et le HTML.

PhpStorm est un IDE PHP qui suit les dernières tendances PHP et langages Web, intègre une variété d'outils modernes et apporte encore plus d'extensibilité avec la prise en charge des principaux frameworks PHP.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Netbeans** | **PhpStorm** | **Meilleur Choix** |
| **Plugin Symfony4** | Non | Oui | PhpStorm |
| **Prix** | Gratuit | Gratuit (30 jours) | NetBeans |
| **Utilisation** | Modéré | Facile | PhpStorm |

Nous allons utiliser PhpStorm pour avoir le plugin de Symfony4 et car c’est le meilleur IDE pour faire du PHP.

L’analyse

1. Le diagramme de cas d’utilisation

Le diagramme de cas d’utilisation permet de montrer les différents types de scenarios que les acteurs peuvent rencontrer lors de l’utilisation d’un projet. Ces scenarios permettront de créer des diagrammes de séquences. Voici ci-dessous le diagramme de cas d’utilisations pour notre projet :



*Figure 1 Diagramme de cas d'utilisation du projet*

Les diagrammes de séquences

Les diagrammes de séquences sont la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique. Voici le premier diagramme Acquisition des données Température et GPS de la Raspberry de livraison :

Une image contenant texte, carte

Description générée automatiquement

*Figure 2 Diagramme de séquence pour l’acquisition*

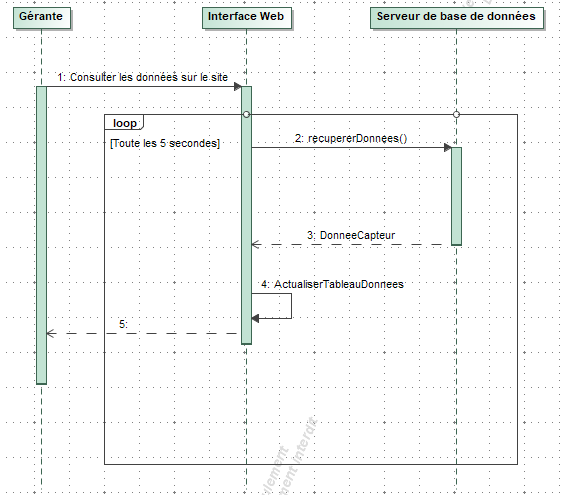
Voici le deuxième diagramme Synchronisation des données Température et GPS de la Raspberry de livraison :

Une image contenant texte, carte

Description générée automatiquement

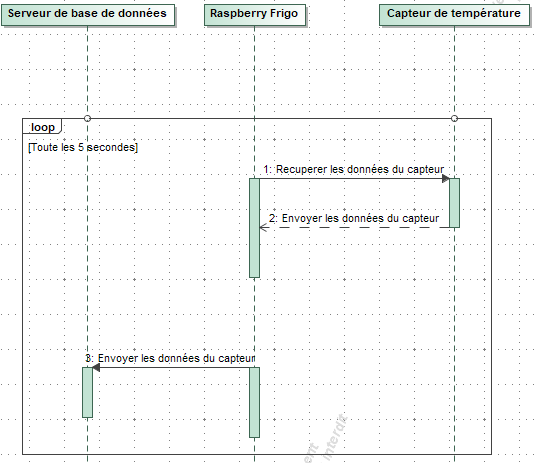
Figure 3 Diagramme de séquence pour la synchronisation

Voici le diagramme orienté pour le site WEB, ce diagramme ci-dessous permet de consulter les données.



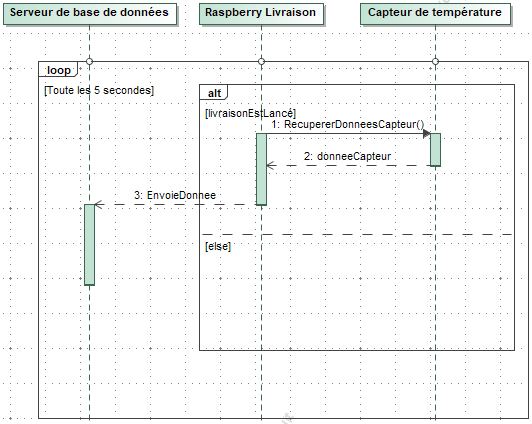
*Figure 4 Diagramme de séquence pour la consultation des données*

Le diagramme ci-dessous, nous montre comment nous récupérons les informations de température de la chambre froide.



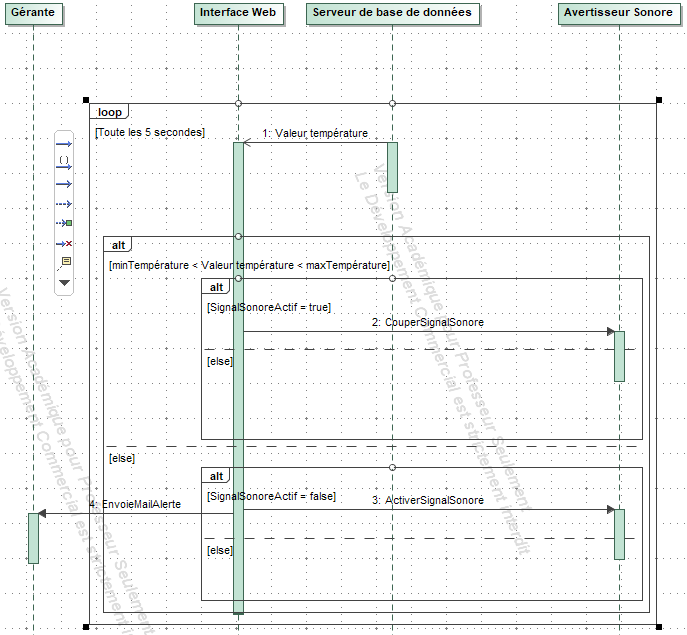
*Figure 5 Diagramme de séquence pour la chambre froide*

Le diagramme ci-dessous, nous montre comment nous récupérons les informations de température d’un véhicule frigorifique.



*Figure 6 Diagramme de séquence pour un véhicule livraison*

Notre dernier diagramme permettra d’alerter la gérante en cas de problèmes de température.



*Figure 7 Diagramme de séquence pour alerter la gérante*

Diagramme de classe

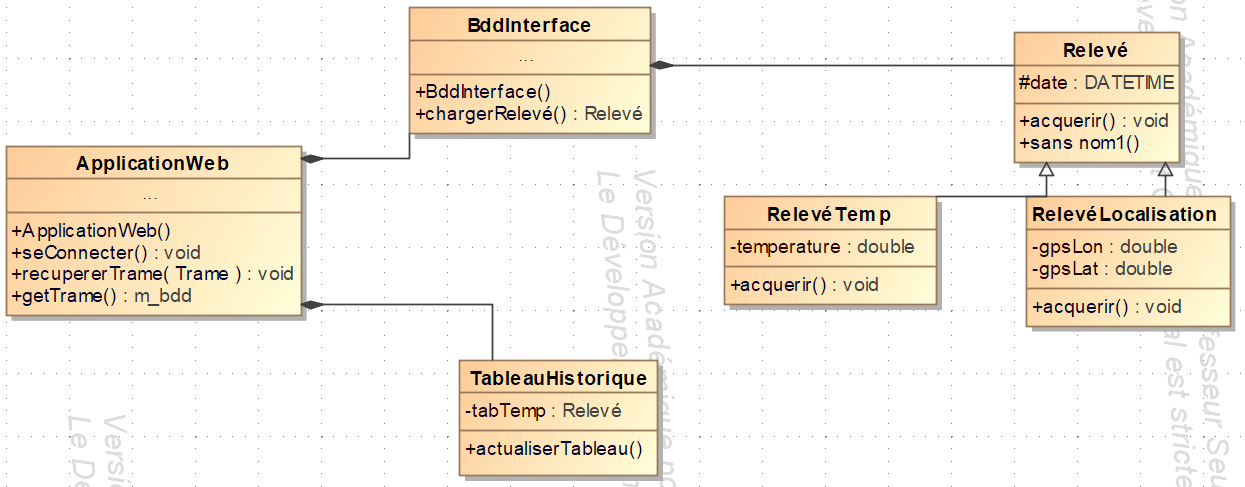
Le diagramme de classe permet de présenter les classes et les interfaces des systèmes ainsi que les différentes relations entre celles-ci. Le diagramme ci-dessous représente la partie des systèmes de relevé de température et de données GPS.

Une image contenant carte, texte

Description générée automatiquement

Figure 8 Diagramme de classe pour la Raspberry de livraison

Le diagramme ci-dessous, nous montre les classes à réaliser pour l’application WEB.



*Figure 9Diagramme de classe pour l'application Web*

La base de données sur MariaDB

La base de données permettra de stocker les informations des différentes Raspberry. Elle possèdera trois colonnes, l’une se nommera « ChambreFroide », l’une « Livraison » et la dernière sera « users ». Les deux premières colonnes récupèreront les informations du capteur de température lié à sa Raspberry. La particularité du deuxième est qu’il récupèrera en plus les informations de longitude et de latitude du capteur GPS lié à sa Raspberry. La dernière colonne permettra à la gérante du site de pouvoir consulter des informations, classées « confidentiel ». Il y aura donc le statut « admin » et le statut « user ». Un mail devra être adresser à la gérante, donc il faudra qu’elle renseigne son adresse mail dans la colonne adéquate.



*Figure 10 Diagramme de la base de données*

Interface WEB

Notre interface WEB permettra à la gérante de pouvoir visualiser les informations stockées dans la base de données. La gérante devra cependant se connecter avec ses identifiants afin d’accéder à ces informations. Le site aura une page « Suivie livraison », « Chambre Froide » et « Historique ». C’est ce même site qui enverra un courriel à la gérante, si un incident concernant la conservation des aliments intervient. Vous trouverez les prototypes de ces pages en annexe.

Charte graphique

Notre charte graphique sera composée des deux couleurs de l’entreprise. Le rouge pourpre (#a40233) et le jaune (#fbb021). Ces couleurs seront les couleurs de notre site et du graphique nous permettant de voir la température de la chambre froide.

Partie Physique

1. Changement de température (ENET)

L’une des parties physiques de notre projet se posera sur la thermodynamique du changement d’état de refroidissement. Le refroidissement est une perte de puissance colorifique. Elle peut se perdre de trois façons :

* La perte provenant du rayonnement du corps (exemple : fer rouge proche, nous chauffe par ses infra-rouges)
* La perte par conduction (exemple : la peau se chauffe si on la plonge dans de l’eau chaude)
* La perte par convection (exemple : un mouvement d'air chaud réchauffe la peau beaucoup mieux que de l’air stagnant à même température)

La façon qui nous intéresse est la troisième, la convection. Cette façon correspond à la manière avec laquelle, nos plats peuvent varier de température. Car pour être plus précis que l’exemple ci-dessus, voici sa définition. La convection se fait grâce au contact du fluide ambiant/avec le corps, ce qui arrache de petits éléments de matière plus important quand il y a fort brassage volumique du fluide.

Cette formule de perte s’écrit : **(Pc = h1.S.∆T)** où h1(W/m²-K) est le coefficient de transfert thermique, S (m²) est la surface et DT la différence de température.

Les capteurs de température (FLENET)

Une seconde partie qui nous intéressera à sein de notre projet sera le type de capteur thermique, en effet il serait intéressant de comprendre comment une information comme la température peut être traduit en données numériques, et plus précisément comment le capteur que l’on va utiliser fonctionne.

Il existe plusieurs types de capteurs de pouvant interpréter et transformer en message électrique la température :

* Thermocouple (consiste en deux tiges de métaux différents : ayant une résistance à la chaleur différente. Un thermocouple calcule numériquement grâce la différence de tension entre les deux barres.

En effet les électrons de par leur nature, se dirigent, dans un matériau, toujours vers l’extrémité la plus froide, sachant que les deux tiges de métal sont d’un métal différent, pas la même quantité d’électrons se retrouveront aux extrémités froides,

Par conséquent une différence de tension se crée et est facilement calculable grâce à un voltmètre.

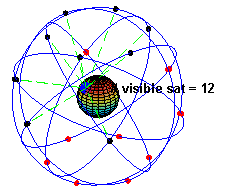
* RTD : capteur de température a résistance de platine (une résistance dans un circuit électrique varie avec la température, le platine est un matériel auquel la courbe de sa résistance associée à sa température est linéaire, il est donc très simpliste de trouver la résistance et d’en déduire sa température)
* Thermistor (2 types : NTC (negative temperature coefficient) et PTC (positive temperature coefficient) ou la résistance augmente ou descends en fonction de la température (similaire rtd)

Et enfin notre type de capteur : les semi-conducteurs comme le silicum, c’est un matériau qui n’est ni tout à fait un conducteur d’électricité, ni tout à fait un isolant. Il peut être soit l’un, soit l’autre selon diverses conditions.

Le caractère conducteur ou isolant prend sa source dans la structure même des atomes : chaque élément du tableau périodique possède un certain nombre d’électrons qui sont agencés autour d’un noyau. C’est cet agencement sous la forme de couches d’électrons, différent selon les éléments, qui est responsable de la conductivité électrique.

GPS (Boriboun)

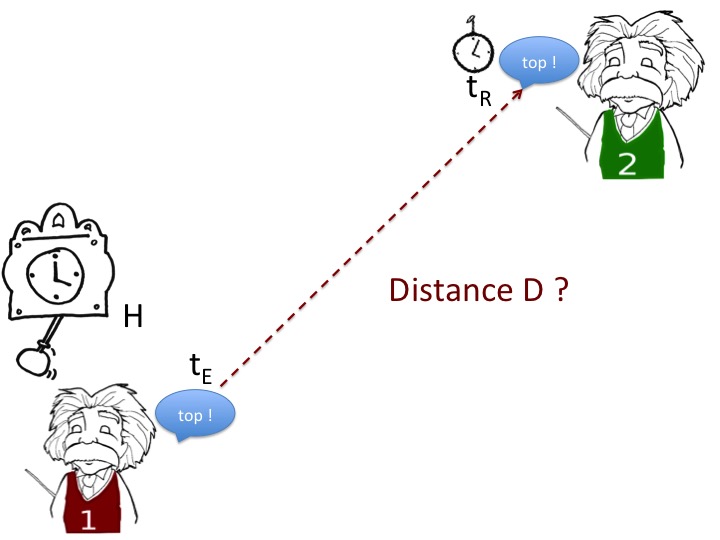
Le système GPS (Global Positioning System) est un procédé pour localisée un objet dans l’espace avec une constellation de 30 satellites en orbite dont 24 satellites actifs autour de la Terre.



Le principe du GPS est assez simple à visualisée :

Par Exemple, on souhaite localiser le presonnage numéro 2 et trouver sa distance D par rapport à l'horloge H.

La personne 1 qui est sous l'horloge H envoie des « top » toutes les minutes. Le signal est émis au temps tE et l'onde se propage pour arriver sur le personnage 2.



Le personnage 2 va recevoir le « top » au temps tR.

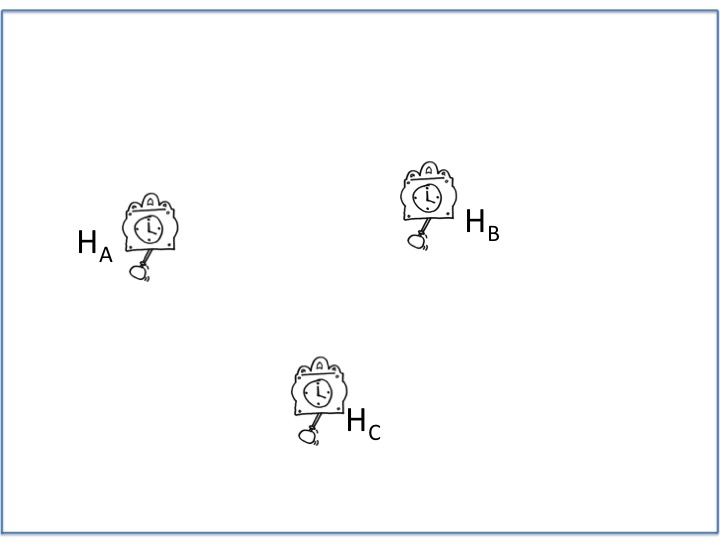
Comme il connait tE, il est capable de savoir à quelle distance D il est situé de l'horloge :

il mesure sur son propore chronomètre tR, il connait la vitesse du son dans l'air et peut ainsi remonter à la distance D qui le sépare de l'horloge.

D = c (tR - tE) = c ΔT

Le principe fondamental du GPS repose sur cet exemple : c'est un calcul de distance à l'aide de la mesure d'un signal dont on connait les dates d'émission et de réception et la vitesse de propagation.

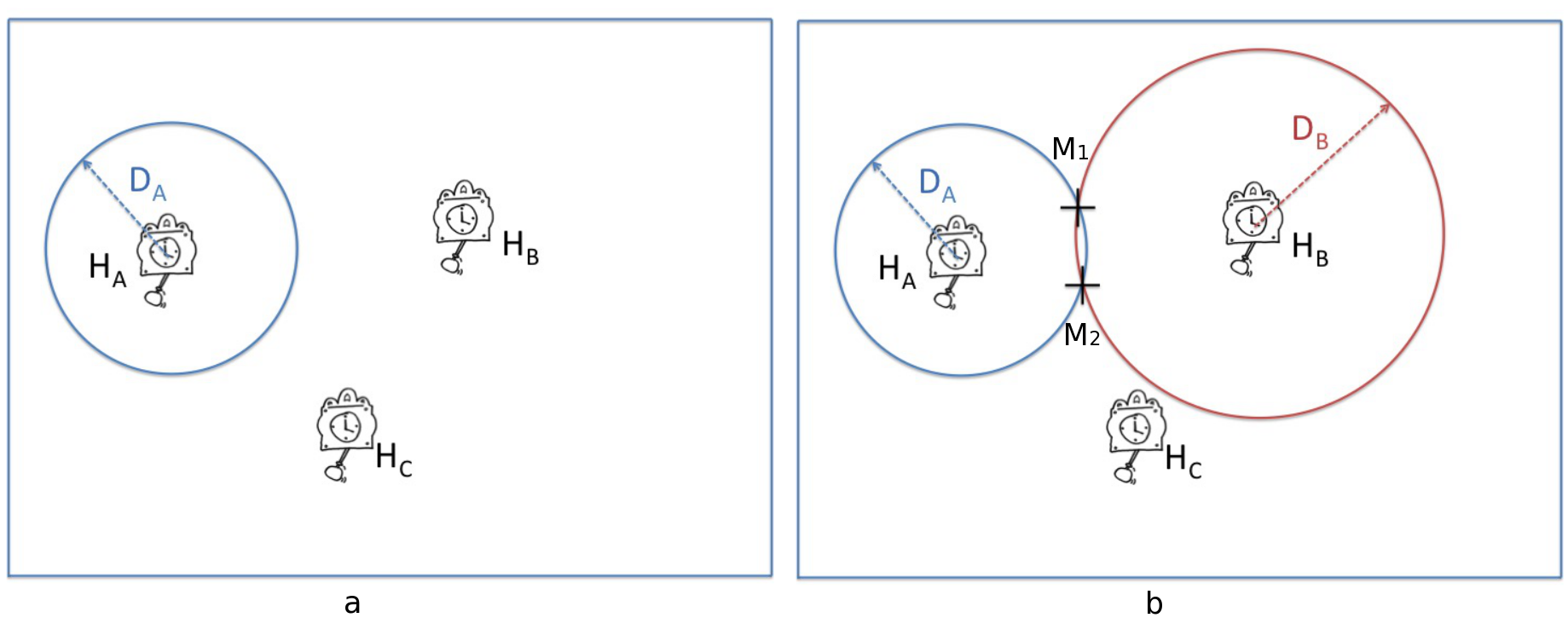
Maintenant, il faut savoir comment ce système fais pour localiser l’objet dans l’espace de manière précise, il faut juste trois satellites minimums.

[](http://culturesciencesphysique.ens-lyon.fr/images/articles/gps/fig2-gps1.jpg)

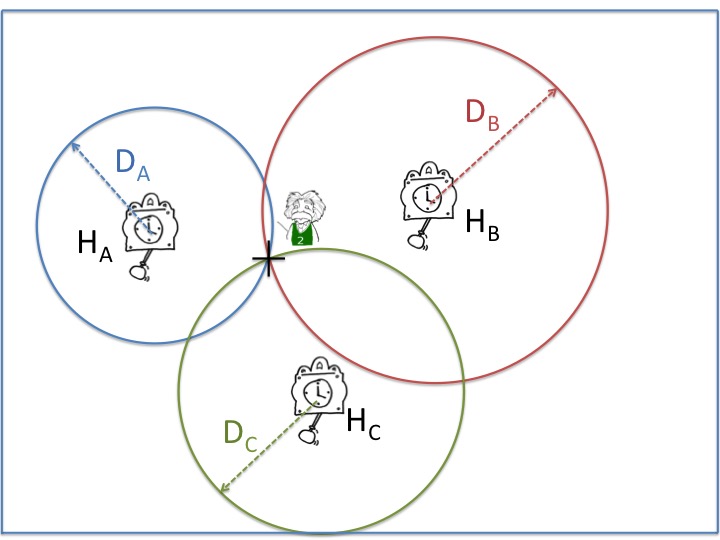
Dans cet exemple, trois horloges différentes HA, HB, HC émettent des sons différents afin de pouvoir les identifier au niveau de la réception. On pourra donc savoir à quelles distances le personnage se trouve des différentes horloges.

À l'aide de l'exemple précédent, on détermine la distance DA qui sépare le presonnage de l'horologe HA, figure 3 a. De la même manière le personnage se trouve aussi sur le cercle de rayon DB, figure 3 b.

Il existe deux points d'intersections. Le personnage est placé soit en M1 soit en M2.

[](http://culturesciencesphysique.ens-lyon.fr/images/articles/gps/fig3ab-gps1.png)

À l'aide de la troisième horloge on sait que le presonnage est aussi sur un cerle centré sur HC et de rayon DC. Cela permet de conclure que le personnage est localisé au point M2.

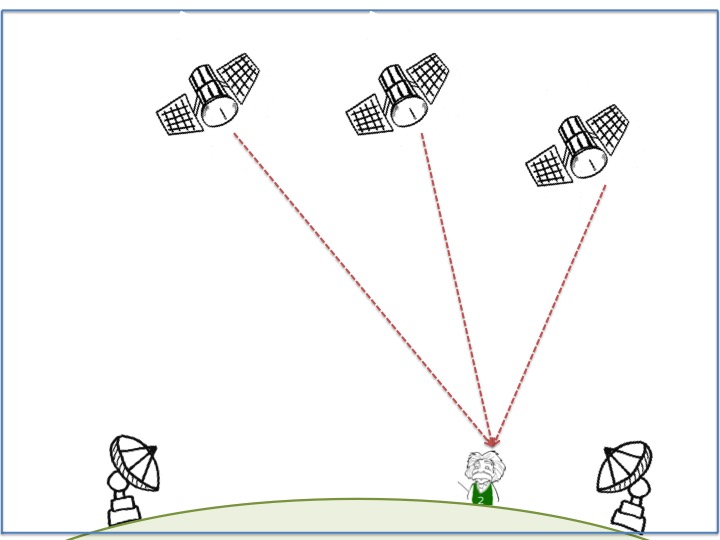
[](http://culturesciencesphysique.ens-lyon.fr/images/articles/gps/fig3-gps1.jpg)

Nous venons de voir le principe de localisation d'un point dans **un plan.** C'est très simple.

On notera que dans cet exemple, les points HA, HB, HC sont des points fixes. Par ailleurs il est nécessaire de connaître très précisément la vitesse de propagation des ondes, pour remonter au calcul des distances.

De plus, ici nous n'avons pas pris en compte les incertitudes de mesure qui se répercutent sur localisation d'une zone plutôt que d'un point.

L'utilisateur est, cette fois, dans **un espace à 3 dimensions** et souhaite se repérer. Il utilise un ensemble de satellites munis d'horloges atomiques synchronisées les unes avec les autres

[](http://culturesciencesphysique.ens-lyon.fr/images/articles/gps/fig4-gps1.jpg)

Maintenant, les points repères ne sont plus des points fixes. Il est alors nécessaire de connaître les positions exactes des satellites en chaque instant. Cette opération va être réalisée par des stations de contrôle régulièrement disposées sur Terre pour communiquer avec les satellites.

Chaque satellite transmet par ondes électromagnétiques son identification, afin de pouvoir déterminer de quel satellite provient chaque signal. Il envoie aussi des informations pour que l'on puisse connaître sa position et une convention de date d'émission de son signal.

À partir de l'intervalle de temps entre l'émission tE et la réception tR, il est possible de calculer les temps de parcours, les distances aux différents satellites et finalement de trouver sa position sur Terre.

Le principe de la localisation tridimensionnelle n'est pas beaucoup plus compliqué que pour la localisation bidimensionnelle.

Avec un satellite on accède à la distance D. Le personnage est situé sur une sphère de rayon D centrée sur le satellite

|  |
| --- |
|  |

Lorsqu'on dispose de deux satellites, les solutions de position sont sur le cercle d'intersection des deux sphères centrées sur les deux satellites

Avec maintenant trois satellites, il reste 2 points d'intersection. Pour lever l'indétermination, on aura besoin de quatre satellites.

On peut mettre en place un quatrième satellite pour le temps.