**Cahier des charges**

**Serveur d’applications**

A. Fourgs



# 1 Le projet

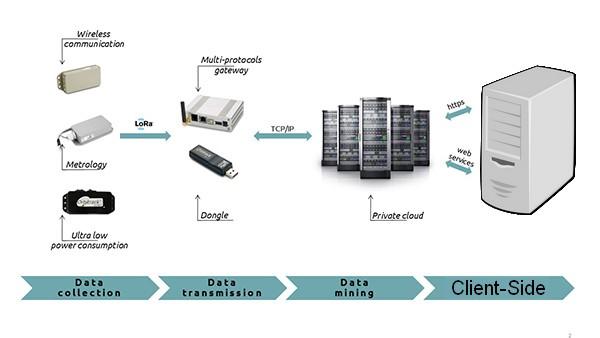
## 1.1 Contexte



Ce projet s’inscrit dans le cadre d’un stage de fin de licence d’informatique suivi à l’Université de Cergy Pontoise. Ce stage a lieu chez *Actility*, une jeune entreprise française piliers dans le domaine des objets connectés et de l’alliance *LoRa*.

## 1.2 Objet

Le but de ce projet est de mettre en place un serveur d’applications côté client.



Le rôle du serveur sera de recevoir, sauvegarder, traiter, éventuellement déchiffrer, et afficher les données transmises par Actility. Tout sera géré automatiquement par le serveur, le client n’aura aucune manipulation à effectuer. Les données seront affichées sous forme de graphes.

## 1.3 Organisation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Semaine\Jour | Lundi | Mardi | Mercredi | Jeudi | Vendredi |
| 05/09/2016 | Design fonction de décodage des PayLoads | Design fonction de décodage des PayLoads | Design fonction de décodage des PayLoads | Design fonction de décodage des PayLoads | Design fonction de décodage des PayLoads |
| 05/16/2016 | Mise en place plateforme AS | Mise en place plateforme AS | Développement Listener | Développement Listener | Développement Listener |
| 05/23/2016 | Mise en place décodeur | Mise en place décodeur | Mise en place décodeur | Mise en place décodeur | Mise en place décodeur |
| 05/30/2016 | Mise en place Base de Données | Configuration Base de Données | Connecteur Base de Données | Implémentation Base de Données | Implémentation Base de Données |
| 06/06/2016 | Écriture d’un cahier de tests | Écriture d’un cahier de tests | Mise en place de l’affichage | Mise en place de l’affichage | Mise en place de l’affichage |
| 06/13/2016 | Test bout en bout | Test bout en bout | Test bout en bout | Test bout en bout | Test bout en bout |

## 1.4 Environnement de travail

Nous utiliserons un ordinateur que nous transformerons en serveur sous Unix (Ubuntu).

Le serveur ainsi que tous les outils de parsing et de décodage seront programmés dans le langage *Python* :

* Nous utiliserons le framework *Bottle* pour mettre en place le serveur.
* La bibliothèque *lxml* sera utilisée pour parser les fichiers .xml.

La bibliothèque *SQLite* sera utilisée pour mettre en place et administrer la base de données locale.

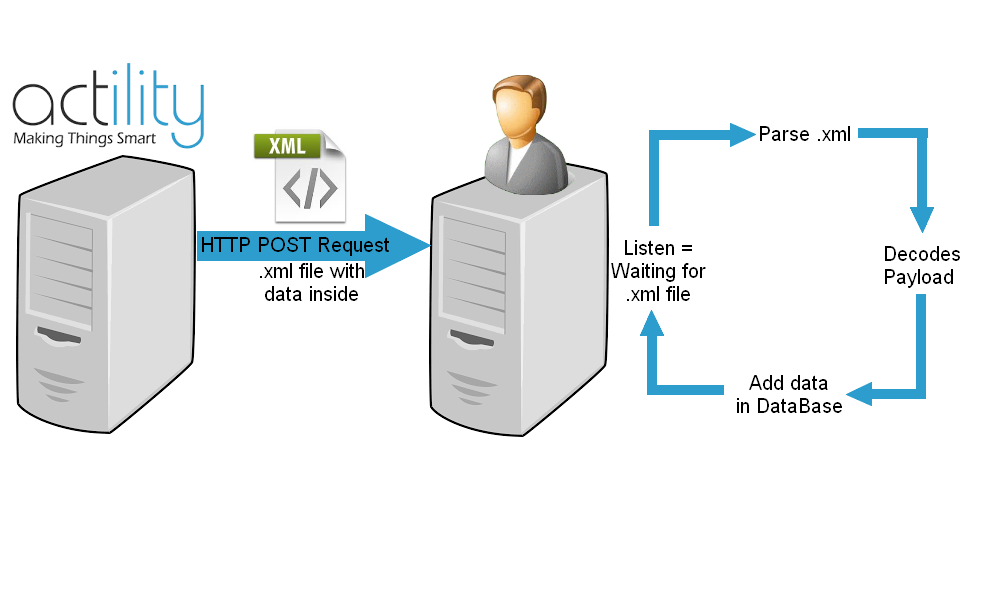
Enfin, pour l’affichage des données, nous programmerons les différentes pages web à l’aide des langages HTML, CSS, PHP et JavaScript, on utilisera plus particulièrement la bibliothèque *amCharts* pour les graphes.

Le développement du travail se fera principalement sur *Atom* sous l’environnement Ubuntu.

Le code sera versionné et partagé à l’aide de Git.

# 2 Objectifs

## 2.1 Fonctionnement du système



## 2.2 Méthodes envisagées

Pour la réception des données le serveur aura un “listener” capable de traiter toute demande du type “POST”.

En cas de soucis lors du traitement des données, nous garderons les données dites brutes afin de pas en perdre, c’est-à-dire que chaque fichier .xml récupéré à l’aide du listener sera sauvegardé tels quels.

Pour le traitement des données, il se fait en deux étapes :

* Décodage XML.
* Décodage Payload.

Dans un premier temps, le serveur va parser le fichier .xml pour en extraire les différentes données dont le PayLoad. Par la suite, le PayLoad sera lui aussi décoder. Enfin, lorsque toutes les données seront isolées à l’aide des différents décodeurs, nous les sauvegarderons dans une base de données sur le serveur client.

Si le client le souhaite, il peut chiffrer les données envoyées par les capteurs afin que ces dernières ne soient pas lisibles durant leur trajet jusqu’au serveur final. Dans notre cas, les données ne seront pas chiffrées, nous n’aurons donc pas besoin de les déchiffrer à leurs arrivés, mais si ça avait été le cas, il aurait été nécessaire de les déchiffrer.

En ce qui concerne le dernier point, l’utilisateur aura la possibilité d’afficher les données sous forme de graphes, le serveur fera donc office de serveur web pour cette partie.

Afin de faciliter la lecture des graphes, nous mettrons en place un graphe dynamique capable d’évoluer en fonction des différentes actions de l’utilisateur, par exemple :  
- On pourra changer l’échelle des axes.  
- On pourra agrandir certaines parties du graphe.  
- Par simple passage à l’aide de la souris sur les points, les données s’afficheront.

# 3 Livraisons attendues

## 3.1 Programmes

Les différents programmes qui seront livrés seront :

* Programme du serveur.
* Programme du moteur qui contient les décodeurs.
* Programme de la Base de Données.
* Programme de génération des graphes.

## 3.2 Documents

Un rapport sera rendu. De plus, les programmes seront documentés en anglais.