Chapter 1

Choix BDD

1.1 Base de données

1.1.1 Choix de la base de donnée

Nous avons tout d'abord dû choisir quelle technologie nous allions utilisé pour notre base de données. Nous connaissions en connaissions un nombre suffisant pour ne pas chercher autre chose et juste faire un choix.

Technologies que nous connaissions

Nom	Niveau de connaissance	Facilité d'intégration	License	
PostGreSQL	Inconnu	Oui	Oui	
MySQL	Bon	Oui	Oui	GPLv2
Oracle Database	Bon	Oui	Oui	
sqlite3	Bon	Non	Oui	

1.2 Communication avec l'application

1.2.1 Serveur

1.2.2 Protocole

Liste des messages possibles dans le sens clients → serveur

Commande	Utilisation
Subscribe:$< id >:< mdp >$	Permet de s'inscrire
Connect:$< id >:< mdp >$	Permet de se connecter à son compte
History:$< debut >:< fin >$	Permet de récupérer x trajets entre début et fin (en id)
Projects:$< debut >:< fin >$	Permet de récupérer les x projets entre début et fin
NewP:$< nom >:< x + y + z; ... >$	Ajoute un nouveau projet
NewJ:$< nom >:< x + y + z + t; ... >$	Ajoute un nouveau trajet
EditP:$< id >:< x + y + z; ... >$	Modifie un projet

Liste des messages possibles dans le sens serveur → client

Commande	Utilisation
Subscribed:$< id >$	Confirme l'inscription
Unsubscribed	Erreur lors de l'inscripton
Connected:$< id >$	Valide la connection à son compte
Unconnected	Erreur lors de la connection
Project:$< id >:< nom >:< x + y + z; ...; x + y + z >$	Envoie des informations sur un projet
Journey:$< id >:< nom >:< d >:< x + y + z + t... >$	Envoie des informations sur un projet

Chapter 2

Serveur

2.1 Architecture

2.1.1 Processus

Le serveur utilise une architecture avec plusieurs processus. Le processus principal attend une connection provenant d'un client (application). A chaque connection un nouveau processus est créé pour intéragir avec le client. Ce processus attend donc un message du client, exécute la commande SQL nécessaire pour obtenir une réponse et enfin, il répond au client en formattant les données. Un processus supplémentaire existe pour pouvoir travailler directement avec le serveur, sans passer par un client. Pour cela, l'entrée et la sortie standard sont utilisées et il faut donc un accès direct à la machine.

2.1.2 Classes

Le processus principal n'utilise qu'une seule classe *Serveur*, tout comme le processus de communication direct *LocalCommand*. Cependant le processus de communication avec le client est séparé en plusieurs objets : *Client*, *SQLHandler* et *CommunicationHandler*. Le premier est le processus en lui même et utilise les deux autres pour effectuer les tâches qui lui sont assignées. L'objet *SQLHandler* accède à la base de données et formatte les réponses. Enfin le *CommunicationHandler* gère la communication réseau avec le client, il permet la réception et l'envoie de chaîne de caractères. Enfin deux classes sont utilisées pour facilité le travail de conversion des formats pour les points : *Point3D* et *Point4D* qui correspondent respectivement à une coordonnée dans l'espace : (x, y, z) et à une coordonnée dans l'espace et le temps : (x, y, z, t).

2.1.3 Echange type

TODO : Diagramme de séquence

2.2 Implémentation

2.2.1 Langage

TODO : Pourquoi java ?

2.2.2 Fonctionnement des processus

TODO : Runnable, Threads

2.2.3 Patron de conception

TODO : Fabric/Builder sur le serveur pour les clients
TODO : Singleton Serveur + (LocalCommand ???)

2.2.4 Communication réseau

TODO : Expliquer Serveur Socket et Socket, le accept()
TODO : Insérer le protocole de communication écrit plus haut