Cahier de Spécifications

Projet : RFAI7	Visualisation et comparaison de connectomes à l'aide de graphes	
Emetteur:	Jean-Baptiste HUYGHE	**MOA:** Jean-Yves RAMEL
Date d'émission:	18/11/2020	

Validation			
Nom	**Date**	**Valide (O/N)**	**Commentaires**

Historique des modifications		
Version	**Date**	**Description de la modification**
0	18/11/2020	Version initiale
1	25/11/2020	Suite et application des conseils de Nicolas Ragot

Table des Matières:

Introduction

Le présent document est un cahier des spécifications, il s'agit d'un document qui résume les principaux éléments d'un projet et qui définit ses objectifs. Le cahier des spécification servira de référence pour toute la suite du projet.

1. Contexte de la réalisation

1.1. Contexte

Ce projet est un projet informatique qui regroupe deux domaines, le domaine du médical et celui de l'informatique. Plus précisément il s'agit d'un projet qui relie les graphes et les connectomes dans un projet informatique. Un connectome est une représentation des échanges entre les neurones réunis dans différentes zones du cerveau. De par leur nature, les connectomes peuvent être représentés par des graphes. Il s'agit d'un terrain peu exploré et mon tuteur et client monsieur Jean-Yves RAMEL souhaite étudier. Après des échanges, le Centre Hospitalier Régional Universitaire de Tours a montré son intérêt pour ces recherches et collabore en fournissant notamment des fichiers contenant des connectomes. Le projet présenté ici à deux but à terme, permettre au CHRU de Tours d'analyser et d'étudier des connectomes à partir de fichier dans un logiciel et de l'autre de tester et proposer une suite d'outils (une API) open source, utilisable par des des chercheurs et informaticiens, pour étudier les utilisations possibles des graphes leurs algorithmes sur les connectomes.

1.2. Objectifs

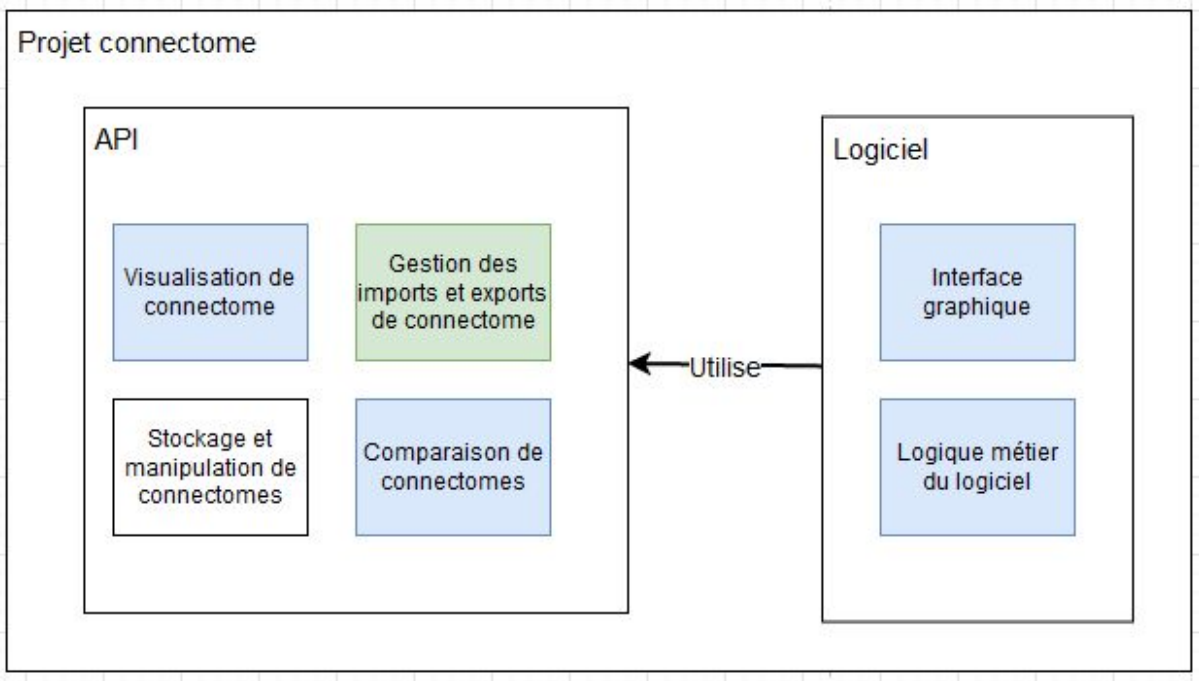
Dans ce contexte, monsieur Jean-Yves RAMEL à lancé un projet l'année dernière pour répondre à ce double objectif. Celui-ci à eu lieu dans le cadre d'un Projet de Recherche et Développement et à été réalisé par monsieur Clément CONDETTE. Le projet a bien avancé mais n'est pas terminé, il permet à l'heure actuelle d'importer des connectomes à partir de plusieurs types de fichiers, de choisir de calculer certaines informations et de visualiser les connectomes compatibles en 3 dimensions. L'objectif sera pour moi de reprendre l'existant, de l'analyser, de l'améliorer et d'ajouter de nouvelles fonctionnalités. Le but de ce document sera de vous présenter en détail ce qui devra être fait.

2. Description générale

2.1. Environnement du projet

Le projet est la reprise et la continuité d'un projet réalisé l'année dernière en Projet de Recherche et Développement par Clément Condette. Il s'agit d'un projet réalisé en python sur l'IDE PyCharm. En plus des fonctionnalités de base du langage, utilise des librairies Python extérieur com NetworkX. Le projet à été réalisé sur le système d'exploitation Windows et devra y rester compatible car les utilisateurs de la partie logiciel graphique l'utilise. Le projet doit également être compatible avec GNU/Linux qui pourra être utilisé par les utilisateurs de l'API.

Le projet existant est divisé en différentes parties qui travaillent ensemble, elles sont bien divisées dont je conserverai la structure actuelle. Cependant, pour répondre au besoin, je vais devoir agir sur quelques-unes d'entre elles. Dans le plan ci-dessous, les cases bleues correspondent aux parties que je devrais modifier pour répondre aux besoins. La case verte correspond aux fonctionnalités facultatives.



(Structure simplifiée du projet)

2.2. Caractéristiques des utilisateurs

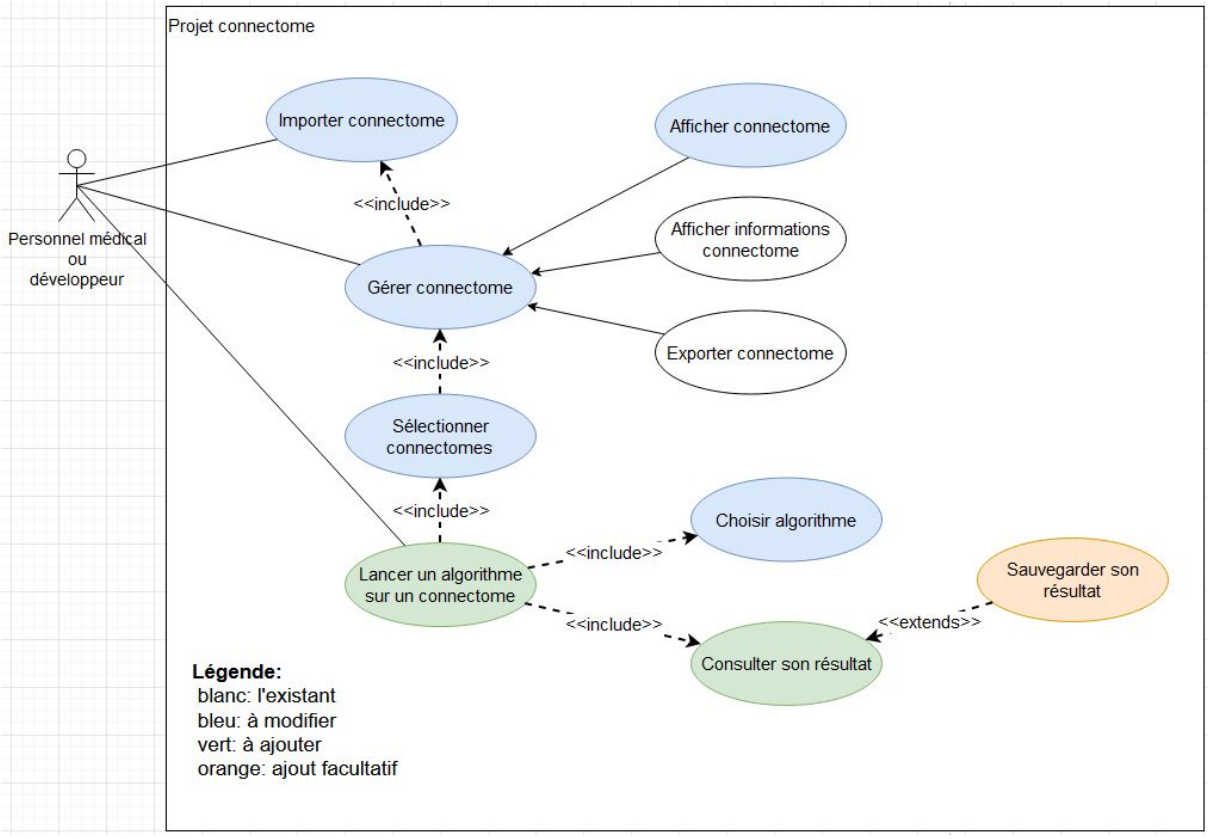
Comme il y a deux parties à ce projet, le logiciel et l'API, il y deux profils d'utilisateurs bien distincts.

Le premier profil d'utilisateurs est celui de ceux qui utiliserons la partie logiciel doté d'une interface graphique. Il s'agit de personnel médical, cela peut comprendre des médecins, des chercheurs sur les connectomes, des infirmières,... Ce sont des personnes qui ont une connaissance poussée du domaine et des termes médicaux, mais pas de connaissances particulières en informatique. Cela implique que le logiciel pourra contenir des termes médicaux mais que l'interface graphique devra être simple à prendre à main, intuitive et ergonomique.

Le second profil d'utilisateurs est celui de ceux qui utiliserons l'API. Ce profil rassemble des chercheurs, développeurs,... qui possèdent une connaissance avancée en informatique. L'API doit pouvoir être comprise et intégrée facilement à d'autres projets, pour cela le projet devra être bien commenté et documenté.

2.3. Fonctionnalités du système

Cette partie présente dans les grandes lignes les fonctionnalités qui existaient déjà, celles que je vais devoir modifier ou ajouter. Pour cela, voici un diagramme de cas d'utilisation qui les résume.



(Cas des cas d'utilisation du système)

2.4. Structure générale du système

Le projet est donc composé de deux parties, la partie API et la partie logiciel graphique.

La partie API n'a pas d'interaction avec d'autres composants si ce n'est la librairie NetworkX pour manipuler des graphes. L'API est en elle même décomposée en 3 sous parties: une s'occuper des imports et exports de fichiers de connectomes, une autre du stockage et de la manipulation de connectomes, et enfin une dernière qui permet d'appliquer des algorithmes de comparaison de connectomes.

La partie logicielle graphique a des interactions avec l'API puisqu'elle en utilisera les fonctionnalités. La partie logicielle est donc presque exclusivement composée de composants pour l'affichage. A savoir les parties Vue et contrôleur du modèle MVC.

Il existe un diagramme de classe de l'existant que vous pouvez trouvez en annexe. Cette conception est assez modulaire donc je pourrai facilement intervenir dessus en modifiant uniquement la partie concernée.

3. Description des interfaces externes du logiciel

3.1. Interfaces Homme / machine

Dans la partie logiciel graphique, l'ergonomie est importante pour que la prise en main soit simple et intuitive. En conséquence, les mêmes types fonctionnalités doivent se trouver proche. Elle doit respecter les conventions des logiciels usuels (selection, menus,...). Les procédures et tâches doivent être guidées, pour l'import des fichiers notamment. Les messages d'erreurs doivent être simples à comprendre.

La partie API doit reprendre le même principe concernant la prise en main, les messages d'erreurs.

Vous trouverez en annexe les propositions des maquettes pour les interfaces graphiques attendues.

3.2. Interfaces logiciel / logiciel

Le projet aura besoin de droits d'utilisateur suffisants sur le système d'exploitation sur lequel il est exécuté. Cela comprend le pouvoir lire les fichiers à importer ainsi que les droits d'écriture à l'endroit désiré pour l'export.

Ce n'est pas tout à fait une interface logiciel / logiciel puisque cela se fait au niveau de l'implémentation, mais le logiciel graphique utilisera l'API, qui elle-même utilisera des librairies.

4. Spécifications fonctionnelles

4.1. Définition de la fonction 1

Identification de la fonction 1: importConnectomeData
Fonction permettant d'importer les données concernant un connectome dans l'application.

Priorité: primordiale
Entrée :

- Une chaîne de caractères représentant le titre du connectome.
- Un fichier texte contenant l'atlas utilisé pour construire le connectome.
- Une liste de fichiers texte contenant une matrice d'adjacence du connectome.
- Une liste de chaîne de caractères représentant le nom de chaque matrice d'adjacence.

Sortie :

- Un connectome sous forme de graphe.

Cette fonction va stocker l'atlas et les matrices d'adjacence dans des variables pour pouvoir construire un graphe. L'utilisateur va pouvoir lors de l'import définir un nom pour le connectome qui servira pour les affichages et la construction du graphe. Les fichiers d'entrées doivent contenir à minima les données pour construire un graphe binaire et non orienté, avec des labels pour les sommets. L'utilisateur pourra à l'import choisir un titre pour le graphe modélisé à partir de ce connectome. Il pourra importer autant de matrice d'adjacence qu'il le souhaite et y donner un nom aux valeurs contenues.
Les matrices d'adjacence doivent être de même dimensions que l'atlas, sinon une erreur est retournée.

4.2. Définition de la fonction 2

Identification de la fonction 1: importConnectomeDataAndImages
Fonction permettant d'importer les données concernant un connectome dans l'application.

Priorité: facultative
Entrée :

- Une chaîne de caractères représentant le titre du connectome.
- Un fichier texte contenant l'atlas utilisé pour construire le connectome.
- Une liste de fichiers texte contenant une matrice d'adjacence du connectome.
- Une liste de chaîne de caractères représentant le nom de chaque matrice d'adjacence.
- Un fichier ".nii" contenant des images de coupes du connectome

Sortie :

- Un connectome sous forme de graphe.

Cette fonctionnalité est identique à la fonction 1 et reprend exactement les mêmes éléments mais doit permettre en plus d'importer un fichier d'iages de coupe 3D de connectome au format ".nii".

4.3. Définition de la fonction 3

Identification de la fonction 2: visualizeConnectome
Fonction permettant d'afficher un graphe issu d'un connectome en 3 dimensions

Priorité: primordiale
Entrée :

- Un connectome

Sortie :

- Pas de sortie, l'effet est uniquement graphique

Cette fonction va permettre à l'utilisateur de visualiser en 3D une représentation d'un connectome. Au minimum, l'utilisateur pourra tourner la représentation, changer le zoom de la visualisation. L'utilisateur pourra également sélectionner un sommet ou une arête du graphe du connectome. Cette fonction ne devra pas utiliser d'API utilisant de technologie web. La librairie utilisée devra permettre d'afficher des images issus de fichiers ".nii" même si cela n'est pas implémenté.

4.4. Définition de la fonction 4

Identification de la fonction 2: visualizeBrainConnectome
Fonction permettant d'afficher un graphe issu d'un connectome en 3 dimensions ainsi que ses coupes.

Priorité: facultative
Entrée :

- Un connectome

Sortie :

- Pas de sortie, l'effet est uniquement graphique

Cette fonctionnalité est identique à la fonction 4 qui reprend exactement les mêmes éléments mais qui doit permettre en plus d'afficher avec le connectome ses images (issus de son fichier de coupe 3D).

4.5. Définition de la fonction 5

Identification de la fonction : compareGraphs
Fonction permettant de comparer deux graphes ou plus.

Priorité: primordiale
Entrée :

- Une liste de graphe

Sortie :

- Une chaîne de caractères représentant le résultat de la comparaison

Cette fonction est une reprise de l'existant qui devra être adaptée et améliorée.
Elle permet à l'utilisateur de choisir au moins deux graphe de connectome et de les comparer avec un algorithme choisi. Les résultats de la comparaison sont consultables dans une zone de texte dédiée du logiciel, ou dans la console pour l'API.

4.6. Définition de la fonction 6

Identification de la fonction : graphEditDistance
Fonction permettant de mesurer la similarité entre deux graphes en utilisant leurs distance.

Priorité: primordiale

Entrée :

- Deux graphs

Sortie :

- Une chaîne de caractères représentant le résultat de la comparaison

Cette fonction permet à l'utilisateur de mesurer la similarité en utilisant la distance entre deux graphes. Cette fonction doit faire appel à la méthode algorithms.similarity.graph_edit_distance de la librairie Python NetworkX. Le résultat est affiché sous forme de chaîne de caractère dans un espace dédié du logiciel, ou dans la console pour l'API.

4.7. Définition de la fonction 7

Identification de la fonction : graphFindByPattern

Fonction permettant de trouver un motif particulier dans un graph donné.

Priorité: facultative

Entrée :

- Un graph
- Un motif

Sortie :

- Une chaîne de caractères représentant le résultat de la recherche

Cette fonction permet à l'utilisateur de chercher un motif particulier dans un graph du connectome sélectionné. Cela nécessite de pouvoir choisir un motif. Le résultat est affiché sous forme de chaîne de caractère dans un espace dédié du logiciel, ou dans la console pour l'API. Un variante de cette fonctionnalité alternative est d'afficher en plus le résultat en mettant en évidence le motif trouvé sur la visualisation 3D.

Cette fonction peut également contenir des sous fonctions, elles aussi facultatives, permettant la gestion des motifs particuliers. Cela comprend:

- l'ajout de motifs
- la suppression de motifs
- la sélection de sommets sur la visualisation pour en créer un motif
- l'import de motif au format ".graphml"
- l'export de motif au format ".graphml"

Les fonctionnalités d'ajout et de création de modif demande à l'utilisateur un nom pour le nouveau motif. Au niveau de l'implémentation côté API, un motif est un graph comme un connectome.

4.8. Définition de la fonction 8

Identification de la fonction : saveLog
Fonction permettant de sauvegarder les résultats d'un algorithme dans un fichier.

Priorité: facultative
Entrée :

- Une chaîne de caractères représentant la résultat d'un algorithme de comparaison

Sortie :

- Un fichier contenant la chaîne de caractères

Cette fonction permet à l'utilisateur de sauvegarder le résultat d'un algorithme présent dans la zone des logs dans un fichier. L'utilisateur se voit proposer une interface pour choisir où sera sauvegarder les logs ainsi que le nom qu'il souhaite donner au fichier.

4.9. Définition de la fonction 9

Identification de la fonction : selectConnectome
Fonction permettant de sélectionner un ou plusieurs connectomes.

Priorité: primordiale
Entrée :

- Une liste de connectome

Sortie :

- Une liste de connectome selectionnée

Cette fonction permet à l'utilisateur de choisir un ou plusieurs connectomes parmis ceux chargés dans le logiciel. Cette fonction est nécessaire pour pouvoir envoyer une liste de connectome à l'algorithme choisi.

5. Spécifications non fonctionnelles

5.1. Contraintes de développement et conception

Les choix de mon prédécesseur ainsi que les attentes de monsieur RAMEL m'imposent certaines contraintes de développement et de conception. La première concerne l'utilisation du langage de programmation qui sera le python et de l'API graphique base Tkinter et la librairie NetworkX. Le système d'exploitation n'est pas imposé mais le projet devra être compatible sur GNU/Linux et Windows. Le choix de la librairie de visualisation en 3 dimensions d'un connectome ne doit pas utiliser de technologie web.

Le système de version Git devra être utilisé et présenter tous les codes et documents attendus. Le dépôt GitHub devra être utilisé et accessible par monsieur Jean-Yves RAMEL Le rendu final se fera sur le dépôt GitHub.

5.2. Contraintes de fonctionnement et d'exploitation

5.2.1. Performances

Ce projet n'a pas de contraintes particulières concernant les performances ou les temps d'exécution si ce n'est s'exécuter dans un temps raisonnable pour que cela soit confortable pour l'utilisateur. Pour les algorithmes à implémenter, ils n'ont pas non plus d'attente concernant leurs performances / complexité mais il serait préférable d'essayer de prendre en compte cet élément.

5.2.2. Capacités

Il n'y a pas de contrainte de capacité particulière si ce n'est les capacités de stockage ou de calculs de la machine. Il faut cependant garder à l'esprit que plus le connectome à traiter sera gros, plus cela demandera de temps de calcul pour réaliser l'opération désirée.

5.2.3. Contrôlabilité

L'exécution des algorithmes doit afficher un retour dans un espace dédié de l'interface graphique, dans la console pour l'API .

5.2.4. Sécurité

Le projet ne doit pas répondre à un niveau de confidentialité. Les données les plus sensibles sont les informations stockées dans les fichiers de connectomes et dont la responsabilité revient à l'utilisateur.

5.2.5. Intégrité

Il n'y a pas de contrainte d'intégrité, le programme charge en mémoire vive les données importées et travaille toujours sur cette mémoire. Toutes fermeture (volontaire ou non) du logiciel ou de l'API sans effectuer d'export ne sauvegarde aucunes données.

5.3. Maintenance et évolution du système

Les questions de maintenance et d'évolutivité sont très importantes pour ce projet et ce pour plusieurs raisons. La première est que le but final de la partie API est de devenir open source. La seconde est que ce projet pourra être repris par quelqu'un d'autre pour y effectuer différentes actions: la maintenance, l'amélioration et l'ajout de nouvelles fonctionnalités.

En conséquence, le code et le projet devront être bien documentés et testés afin de faciliter toute maintenance ou évolution.

En plus du code source du projet, les tests, la documentation, le cahier des spécifications, et un rapport devront être fournis.

Annexes

Diagramme de classe de l’existant

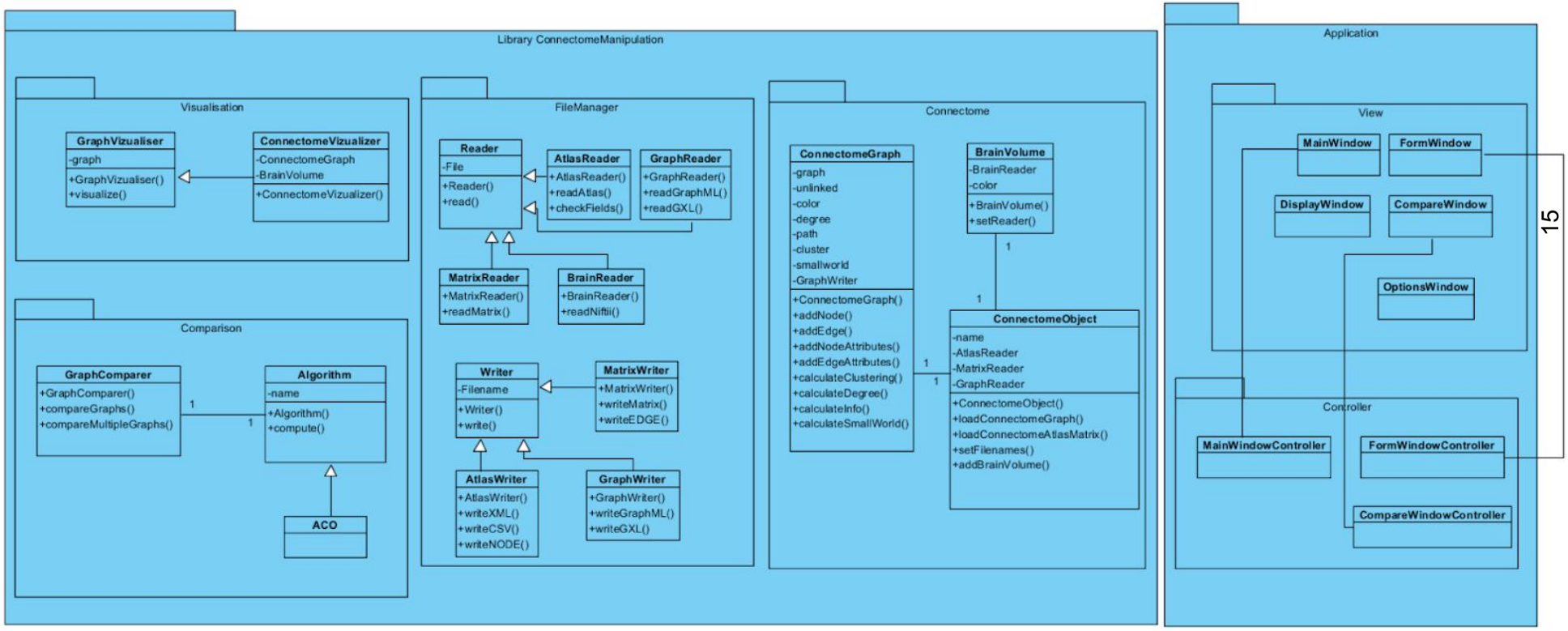
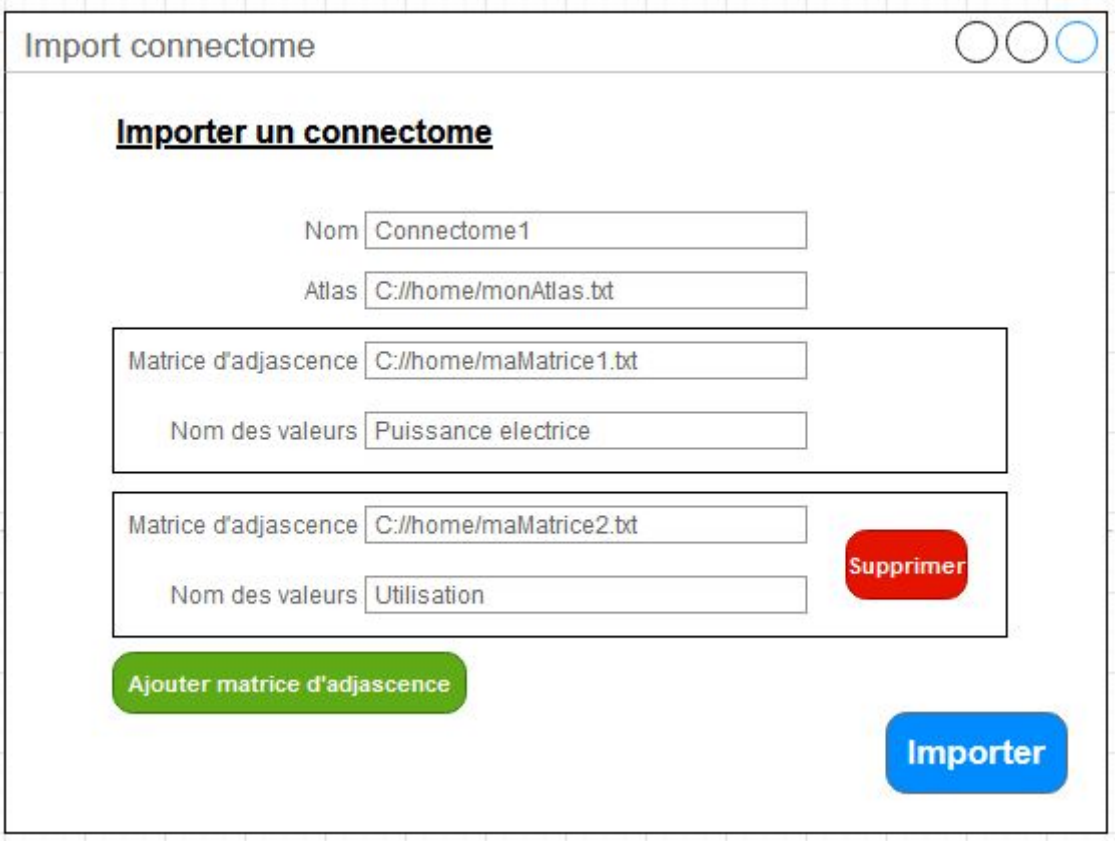


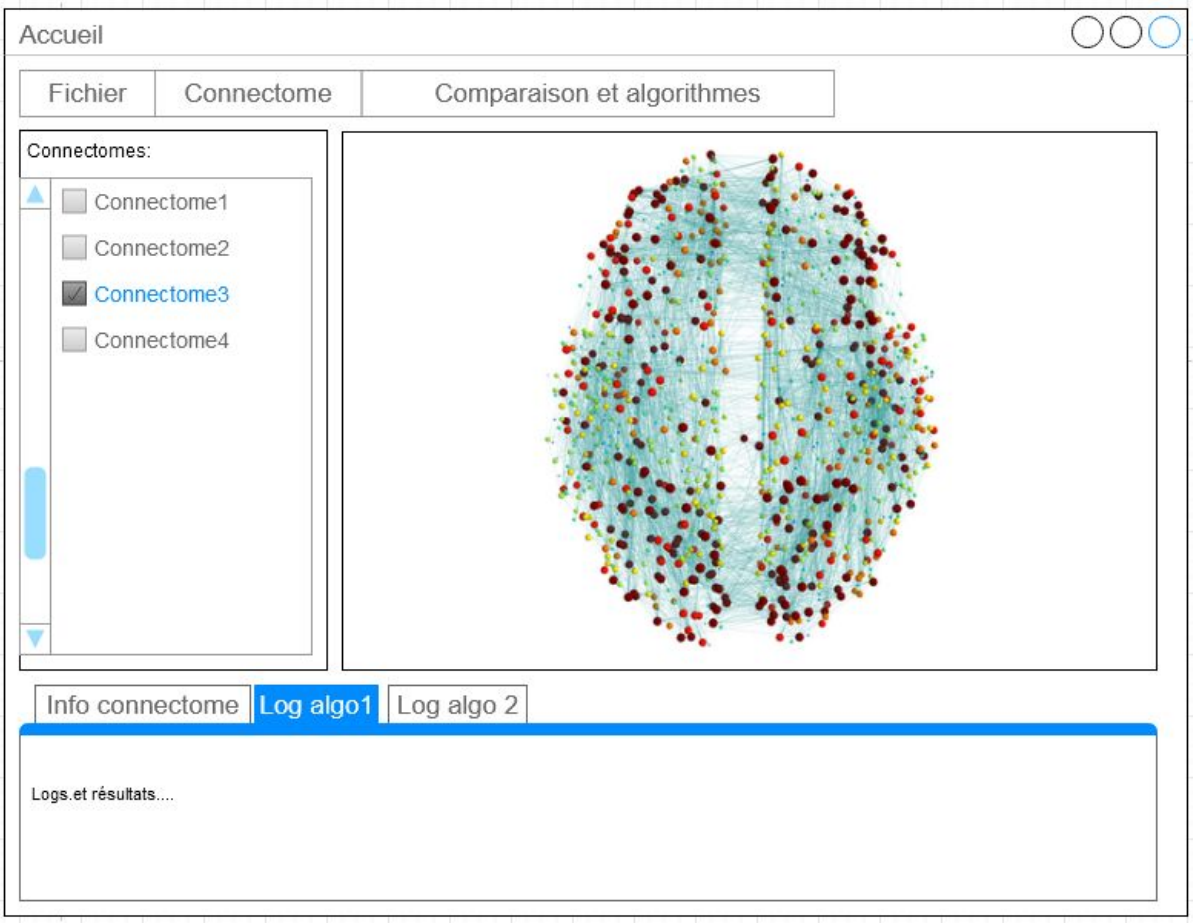
Figure 1 – *Diagramme de classe final du projet de recherche et développement*

Maquette des interfaces Homme-machine

Vous trouverez ci dessous les maquettes des interfaces graphiques:



(Maquette de l'interface homme-machine proposée pour importer un connectome)



(Proposition de maquette de l'interface principale connectome)