**Projet en programmation java**

**Conception et développement d’une application informatique.**

Rapport d’analyse :

Sujet :

Coder un algorithme en java (en utilisant l’orienté objet) qui donne des instructions de navigation pour un déplacement à pied à partir d’une base de données OpenStreetMap en utilisant un service de calcul d’un itinéraire de l’IGN.

Interprétation brève du sujet

Pour donner des indications de navigation à un utilisateur, nous allons faire une interface graphique avec comme fond de cette interface graphique, une carte avec les données OSM. Sur cette interface graphique l’utilisateur pourra cliquer sur le bouton “Départ” et cliquer sur la carte à un endroit précis pour définir sa position de départ, idem pour l’arrivée. Un itinéraire va donc se générer à l’aide du service d’un calcul d’itinéraire de l’IGN. Cet itinéraire va se découper en plusieurs “steps” qui seront en format JSON et nous devrons les transformer en classe java dans le but d’en exploiter les données. Chaque step correspond à un point de décision de l’utilisateur. Pour chaque point de décision, à l’aide d’OpenStreetMap, on va connaître les objets dans un certain périmètre. Nous allons utiliser l’API d’Overpass pour exécuter des requêtes depuis les serveurs d’OpenStreetMap afin d’extraire leur données, comme par exemple le nom de l’objet ou encore la latitude et la longitude de l’objet. A l’aide d’un tri des toponymes, on prendra le toponyme le plus approprié pour donner l’indication à l’utilisateur à chaque point de décision, par exemple : à droite après X, à gauche avant Y. Si il n’y a pas d’objet dans le périmètre défini, on indique juste “à l’intersection”. L’utilisateur devra cliquer sur “suivant” lorsqu’il passe au prochain point de décision. L’application va donc lui donner des informations pour chaque “tronçons”, un tronçon est un itinéraire entre deux points de décision. Les indications données seront : la distance et le temps pris sur le calcul d’itinéraire de l’IGN.

A qui s’adresse ce projet et pour quoi faire :

Ce Projet est à destination de quiconque veut se déplacer à pied et utiliser un système de navigation. Il servira à donner des indications à chaque point de décision à prendre. Les indications seront données en fonction des enseignes visibles par l'utilisateur.

Diagramme UML :

Nous étions partis pour faire les quatres diagrammes mais après réflexion, seul le diagramme de classe et de cas d’utilisation nous sont vraiment utiles. En effet, le diagramme de cas d’utilisation nous à servis à nous structurer et à savoir quoi implémenter dans l’interface graphique. Et le diagramme de classe nous a vraiment aidé au début du projet pour dérouler le développement du code et savoir comment les classes allez interagir entre elles. Le diagramme présenté ici est le premier diagramme que nous avons fait sans même regarder le code source. Après avoir vu le code source, nous nous sommes rendu compte que nous aurons 4 packages : Exemple, OSM, IGN et Portail. Avec des classes qui se rajouteront ensuite. Lorsque nous avons commencé à coder, les idées nous sont venu et nous n’avons pas eu besoin de refaire un diagramme de classe.

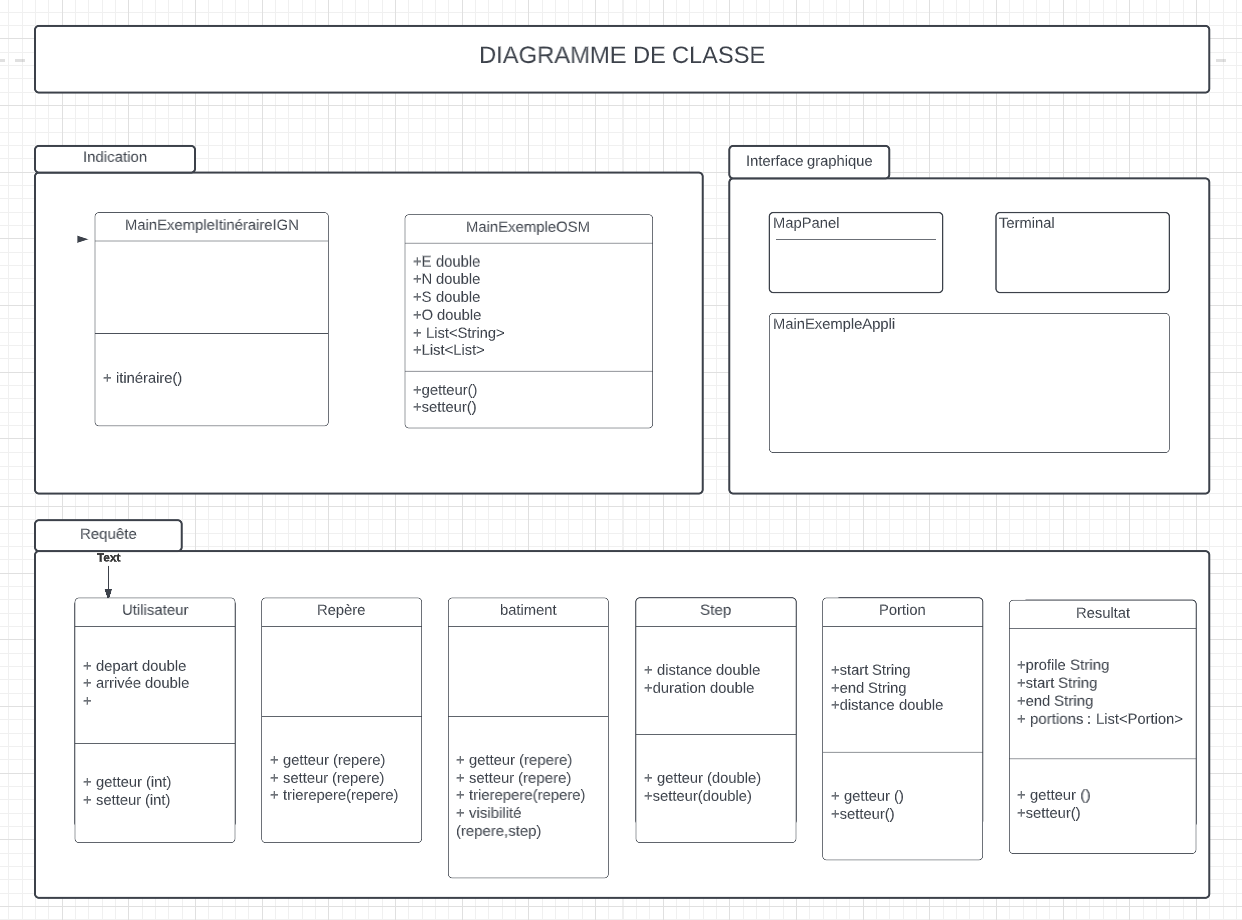
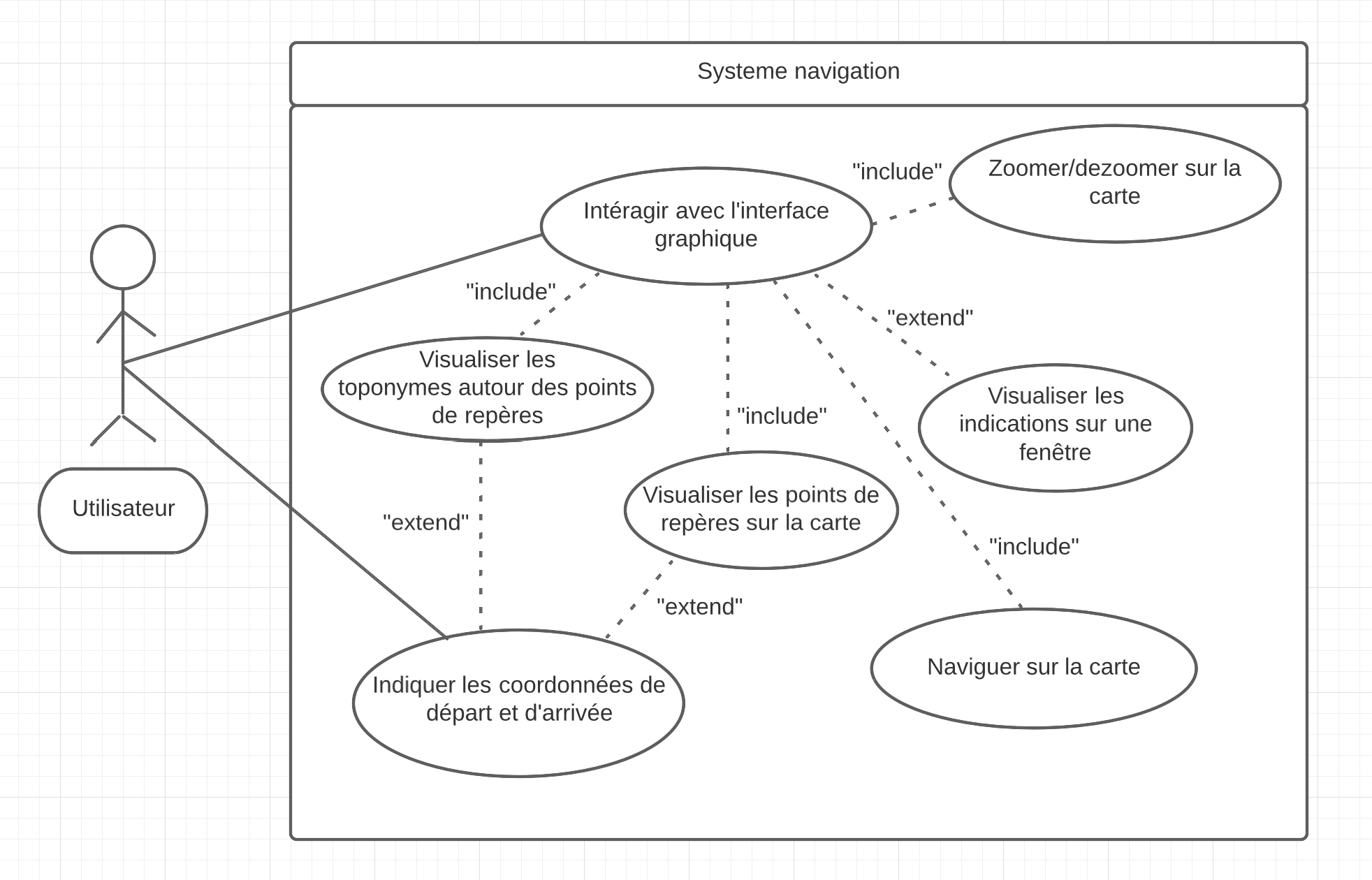
Diagramme de classe :

Diagramme de cas d’utilisation :



Analyse :

Pour l’analyse du sujet, nous sommes partis, chacun de notre côté pour savoir comment faire le code. Quelle nouvelles classes nous devons créer, quelles seront les futures intéractions entre packages et comment tout relier à l’interface graphique. Quelles sont les parties de code que nous possédons déjà avec le code source et comment les exploiter. Nous nous sommes partagé en 3, une personne pour l’interface graphique, une personne pour l’itinéraire issue du service de calcul de l’IGN et une personne pour la partie OpenStreetMap.

Pour la partie OpenStreetMap :

Tout d’abord, nous voulons récupérer les toponymes dans un périmètre donné, c'est-à-dire leur noms et leur coordonnées dans des listes afin de les utiliser ensuite. Le but est qu’à chaque point de décision, on prend les toponymes autour de ce point de repère dans un périmètre (pour que l’objet soit un minimum visible par l’utilisateur) et on lui indique la direction en fonction de ce toponymes. Nous avons aussi fait l’hypothèse qu’on peut trier les toponymes en fonction d’une variable, que nous n’avons pas encore décidé. Nous verrons ce que nous pourrons récupérer comme information sur les toponymes avec l’API d’Overpass. Par exemple la taille ou encore la visibilité de l’objet, un mcdonald's se verra forcément plus qu’un panneau ou autres si ils sont à la même distance de l’utilisateur. Ensuite, on calculera l’angle entre l’ancien point de décision et le nouveau avec comme origine de l’angle, l’actuel point de décision. Peut être que prendre l’ancien et le nouveau point de décision ne sera pas assez précis, nous pourrons peut être découper chaque itinéraire entre deux points en plusieurs points pour être plus précis par la suite. Cette analyse nécessite qu’on puisse avoir tous les points de décision, dans une liste par exemple, et qu’on puisse interagir avec.

Concernant l’itinéraire / le service IGN :

Nous voulons déjà récupérer les informations données par l’IGN pour un itinéraire donné dans des classes, résultats et portions. Dans le but de les trier et par la suite de les faire apparaître sur notre fenêtre ou l’utilisateur pourra interagir. Pour les trier, nous avions pensé à faire le vecteur directeur pour chaque point, 2 par 2, et ensuite les comparer ensemble. Chaque fois qu’un vecteur serait trop différent des autres, on prendrait le point d’origine de ce vecteur et on en ferait un point de décision. Après avoir regardé le fichier JSON, nous nous sommes rendu compte que les points étaient déjà triés. De cela, nous commençons à voir l’organisation et la hiérarchie que les classes créées allaient avoir entre elles (classes visibles dans “Requête” dans notre diagramme UML). A partir de ce que nous donne le service de l’IGN, on s’attend alors à créer des classes et des méthodes pour récupérer des directions, des distances, des itinéraires ou des points d’intersection par exemple.

Pour le portail / MapPanel

On tient dans ce package à créer une interface pratique et fluide qui rendra l’itinéraire visible et compréhensible. Sans connaître exactement les futures fonctions, on voudrait créer des panneaux de configuration où l’on peut rentrer les coordonnées du point de départ et d’arrivée et un autre qui donnerait explicitement l’instruction au client step par step ; en affichant potentiellement aussi la durée et distance totale, ou même step par step. D’un point de vue visuel, nous voulions que notre code affiche les différentes étapes dans un “Scrollmenu” ainsi que les instructions. Que l’utilisateur puisse entrer les coordonnées de départ et d’arrivée très facilement et qu’il puisse se repérer sur la carte à n’importe quel moment, donc qu’il y ait une indication de l’étape au moment donnée. Nous remarquons aussi qu’il n’y a pas de système gps, il faudra donc cliquer sur des boutons grâce à la souris pour indiquer ses intentions, par exemple passer à l’étape suivante.

Lien pour le gitlab du projet :

https://gitlab.com/vitto1/projet-java.git

Vittorio Toffolutti, Matheo Marechal, Marc Ham-Chou-Chong, élèves de l’ENSG