# Rapport Programmation Orientée Objet

Travail 1 : système de gestion des employés d'une entreprise

Dans le cadre du laboratoire de programmation orientée objet nous avons été demandé de créer depuis une demande client un programme qui gère les employés d'une petite entreprise de consultants. Nous avons du passé par une modélisation UML et finir par quelques tests unitaires.

Nous avons travail en binôme en utilisant GitHub pour partager notre travail.
(https://github.com/Sambertrand/Poo1)

## Mode d'emploi de notre programme

### a) Les Entrées

Pour entrer les données nous avons décidé de séparer tous les fichiers textes d'entrée pour plus de clarté lors du remplissage des données et une possibilité de restreindre des accès à certains fichier textes pour certains employés de l'entreprise.

Quelques notes avant de vous expliquer le format des fichier textes :

- Les matricules choisis sont « arbitraires » mais par logique nous avons choisi 2 lettres qui indique le rôle puis des chiffres ordonnés.
- Les missions doivent obligatoirement être écrite en ordre chronologique de date de début.
- Un matricule client est déjà utiliser sans être à rentrer dans la base de données car c'est celui de l'entreprise elle-même (CL00).
- Il est interdit d'entrer deux fois Le matricule.

Voici le format de chaque fichier texte, entre les guillemets se trouve le regex ce qui doit figurer à chaque ligne du fichier texte :

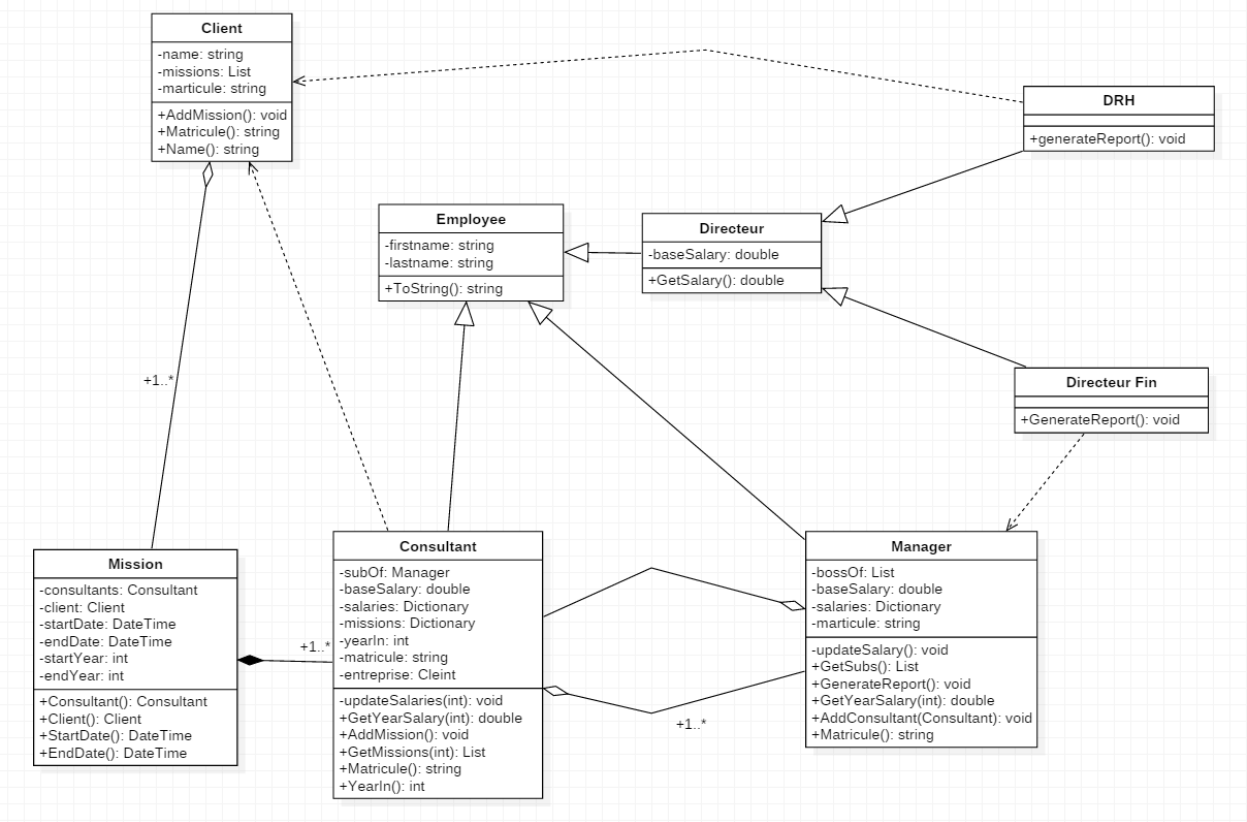
- Clients : « ^\w+ CL\d+$ » (Nom, Matricule)
- Consultant : « ^\w+ \w+ MA\d+ \s{2}( ?:\d{2}) CO(1)\d+$ » (Prénom, Nom, Manager, Année d'arrivée Matricule(attention ici le matricule à 2 lettre puis 2chiffres pour l'année d'arrivée ensuite le numéro))
- Directeurs : « ^D[IFH] \w+ \w+$ » (Code directeur (DI : directeur, DF : directeur financier, DH : directeur des ressources humaines), Prénom, Nom)
- Managers : « ^\w+ \w+ MA\d+$ » (Prénom, Nom, Matricule)
- Missions : « CL\d+ CO\d{2}\d+ (((31-((0[13578])|(1[02])))|([012]\d)-((0\d)|(1[012]))|(30-((0[13456789])|1[012])))-\d{4}) (((31-((0[13578])|(1[02])))|([012]\d)-((0\d)|(1[012]))|(30-((0[13456789])|1[012])))-\d{4})» (Matricule client, Matricule consultant, Date de début, Date de fin)

Si nécessaire les fichiers d'entré son déjà remplis d'exemples pour toute hésitation quant aux entrées.

### b) Les Sorties

Nos sorties sont très simples, par l'interface console vous sélectionner le rapport désiré, une fois le rapport généré il se trouve dans le répertoire du programme sous format .txt prêt à être lus.

## L'UML et structure de code




Le code est structuré en Class. Tout d'abord nous avons une class « Employee » qui sera la super class de toutes les class qui sont des employés de l'entreprise à gérer. Les class Directeur, Manager et Consultant héritent toute les trois d'Employee et apporte à chaque type d'employé de l'entreprise des fonction particulières à leur poste. Il y a deux types de directeurs spécifiques. Le DRH et le Directeur financier héritent donc de la super class directeur et son toute les deux uniquement créent pour générer les rapports. Pour écrire ces rapports le DRH a besoin d'utilisé les informations présentes dans le client et le directeur financier utilise des informations présentes dans le manager. Manager et Consultant sont deux class agrégée entre elles car chaque consultant a un manager et chaque manager à un ou plusieurs Consultants. Pour le bien des entrées et de la clarté des rapports nous avons créé une class Mission. La class consultant est composée de missions, car à tout moment dans la vie du consultant dans l'entreprise il est en mission. La mission doit aussi être liée à un client, ce client peut être l'entreprise elle-même. Client et mission sont donc agrégés car un client à une ou plusieurs missions.

Une fois toutes ces classes crées et liée ensemble nous avons la class program, cette class est à exécuter pour prendre les entrées et générer les rapports. Elle va d'abord chargée tous les fichier texte d'entrée et crée toutes les variables correspondant à ce qui est mis dans les inputs. Il va ensuite y avoir une interface console avec laquelle il faut interagir pour générer un rapport. Le rapport est alors écrit et posté dans le répertoire du programme dans un format .txt.