Projet de programmation : Gestion des compétences du personnel

**Diagrammes UML**

23/05/2017

HENTATI Mahdi – TOUTAIN David – UZAN Lucas

L3 MIAGE – FA

1. Lancement de l’application :

Au lancement de l'application, Program.java est exécuté : il charge les informations des fichiers csv, attribue les compétences aux employés pour les formations qui sont terminées puis affiche la fenêtre principale de notre programme avec la classe ProgramFrame.java. Ce dernier va ajouter à son interface le JPanel Header.java qui se décompose en deux parties : l'en tête de l'interface avec la navigation et le corps de l'interface. Le changement de pages s'effectue au travers de cette classe qui relie donc les actions utilisateur sur la navigation et l'affichage de la page souhaitée.

1. L’accueil de l’application :

La page d'accueil de notre programme est chargée à partir de la classe Accueil.java. Elle se compose d'un tableau de missions actuellement en cours, de missions à venir dans le mois ainsi qu'un tableau composé d'alertes avec les missions où les employés ne sont pas affectés ainsi que celles qui sont en retard.

Notons qu'un double clic sur une mission en retard permet de lui attribuer une date de fin réelle. Un diagramme camembert avec la répartition des missions par statut est également présent. Toutes ces informations permettent à l’utilisateur d’avoir un récapitulatif visuel de l’état des missions.

1. Les formulaires :

La classe Formulaire.java est la classe parente de toutes les pages suivantes. Elle prépare le fond de formulaire, met en place les boutons d’actions « Nouveau », « Modifier », « Supprimer », « Enregistrer » et « Annuler » et propose deux ArrayList destinées à stocker les composants graphiques qui devront être alternativement désactivés et activés.

De plus, chacune des fenêtres suivantes propos un système de filtre avec une barre de recherche et / ou un filtre par statut permettant à l’utilisateur d’arriver plus rapidement à l’objet recherché.

* 1. La page des compétences :

Chargé à partir de la classe Compétences.java, la page offre à l’utilisateur les possibilités de gérer des compétences ainsi que les libellés de chacune. Il peut mettre deux libellés pour chaque compétence : un en français et un en anglais. Le changement du libellé affiché par défaut pour une compétence peut être rapidement mis en place.

Le code d’une compétence doit impérativement respecter les normes Caractère.Chiffre. pour garder une homogénéité. Il ne doit également pas être déjà présent.

* 1. La page des employés :

Chargé à partir de la classe Personnel.java, la page offre à l’utilisateur les possibilités de gérer des employés et les compétences que l’employé possède. Il peut mettre un nom, un prénom, une date d’entrée et de lui attribuer les compétences souhaitées. Le nom, le prénom et la date d’entrée doivent être remplis pour enregistrer.

* 1. La page des missions :

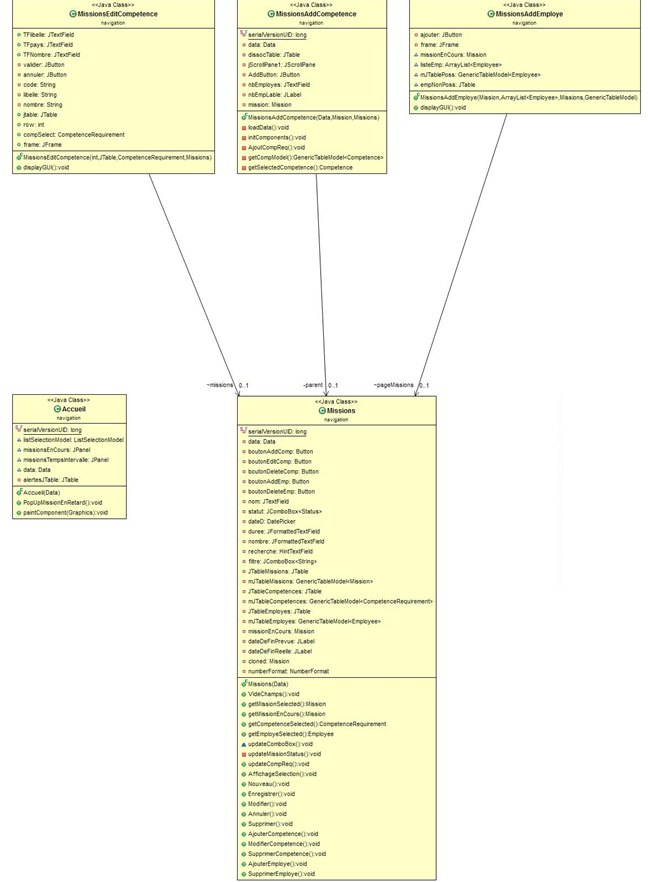
Chargé à partir de la classe Missions.java, la page offre à l’utilisateur les possibilités de gérer des missions, les compétences requises par la mission et les employés affectés. Il peut mettre un nom de mission, le nombre de personnes requises, la date de début de la mission, la durée en jours, les compétences requises et les employés affectés. Il peut également forcer la planification d’une mission en préparation qui n’a pas le nombre d’employés requis si cela est nécessaire.

Pour l’ajout d’un employé, un système de recommandation permet à l’utilisateur de savoir quel employé est le plus apte à exécuter la mission (plus d’informations à propos du fonctionnement des recommandations dans la partie concernée).

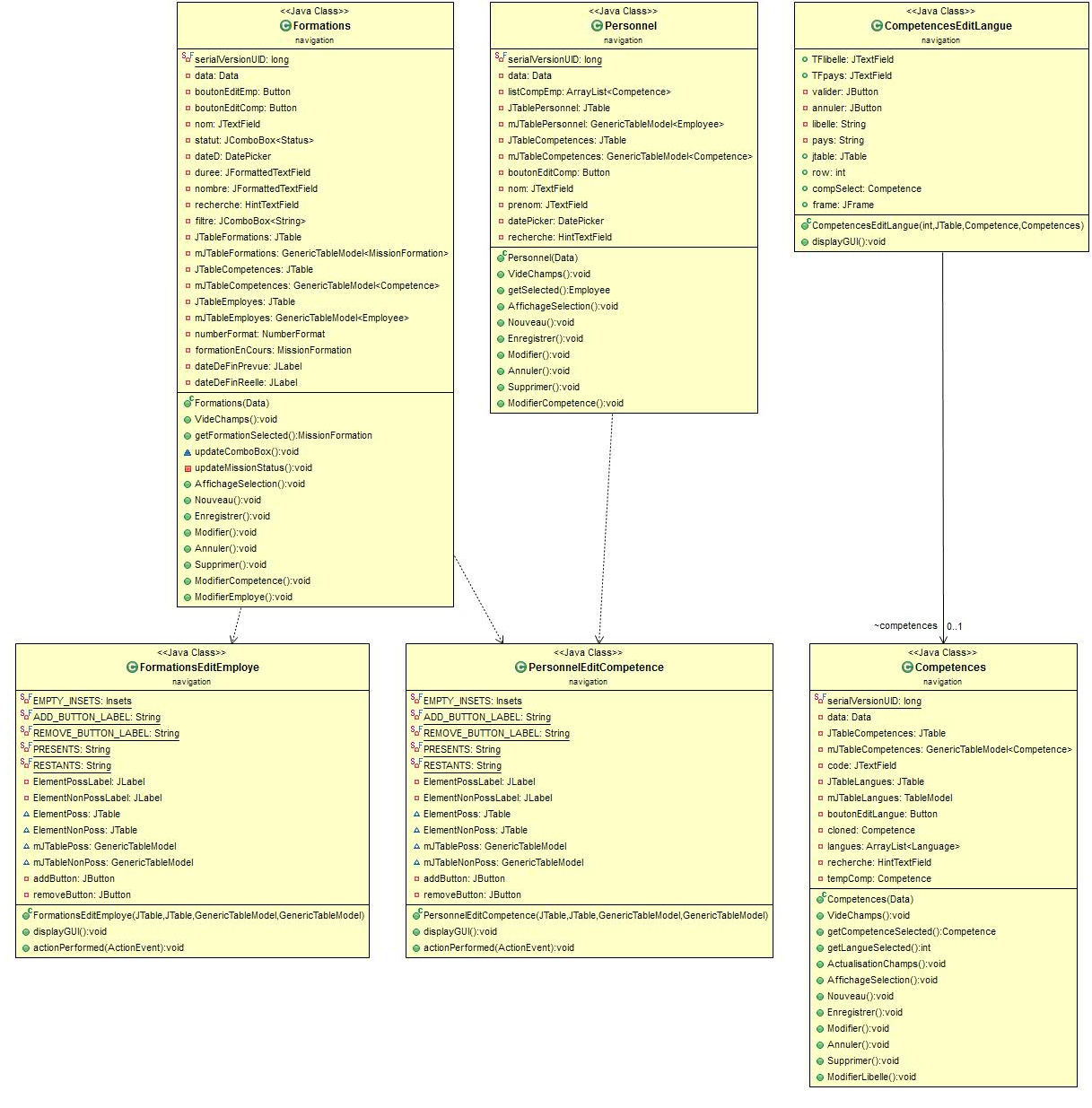
La date de fin prévue est calculée en interne en ajoutant la durée en jours à la date de début puis elle est affichée à l’écran. La date de fin réelle est initialisée à la date de fin prévue mais peut être destinée à être modifiée si la mission finie avant ou après ce qui était prévue. Elle est aussi affichée et permet d’avoir une trace du déroulement des missions.

* 1. La page des formations :

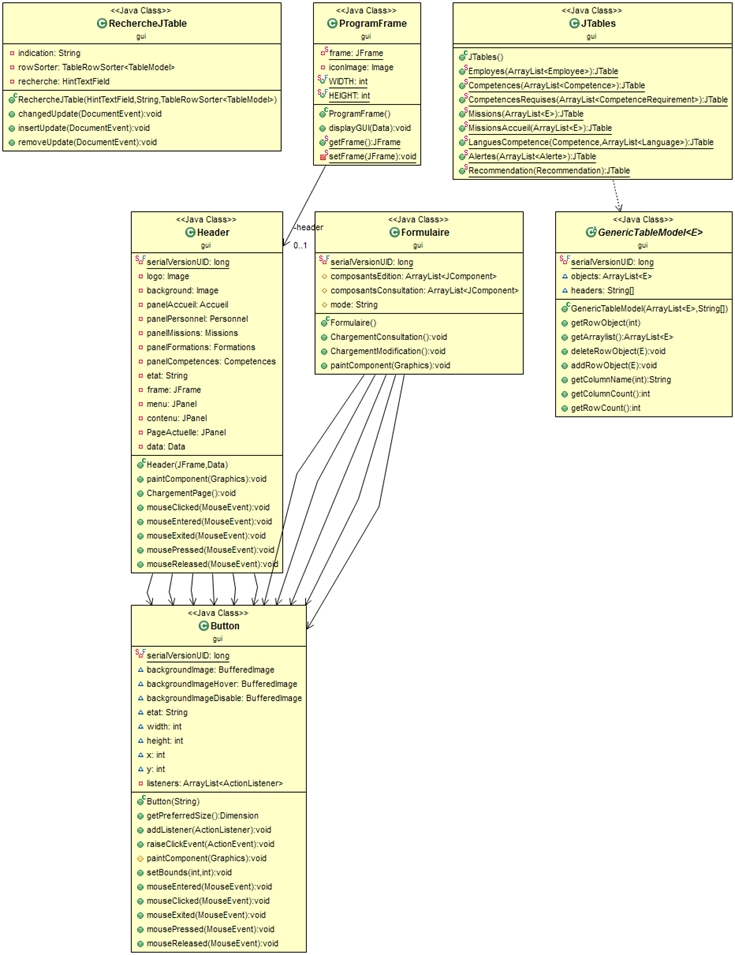
Chargée à partir de la classe Formations.java, la page offre à l’utilisateur les possibilités de gérer des missions de formation, les employés destinés à être formés et la liste des compétences qui leurs seront affectés automatiquement à la fin de la formation. La formation a exactement les mêmes informations qu’une mission (puisque c’est un type de mission !). L’attribution automatique des compétences aux employés concernés lorsqu’une mission est terminée se fait au lancement de l’application.



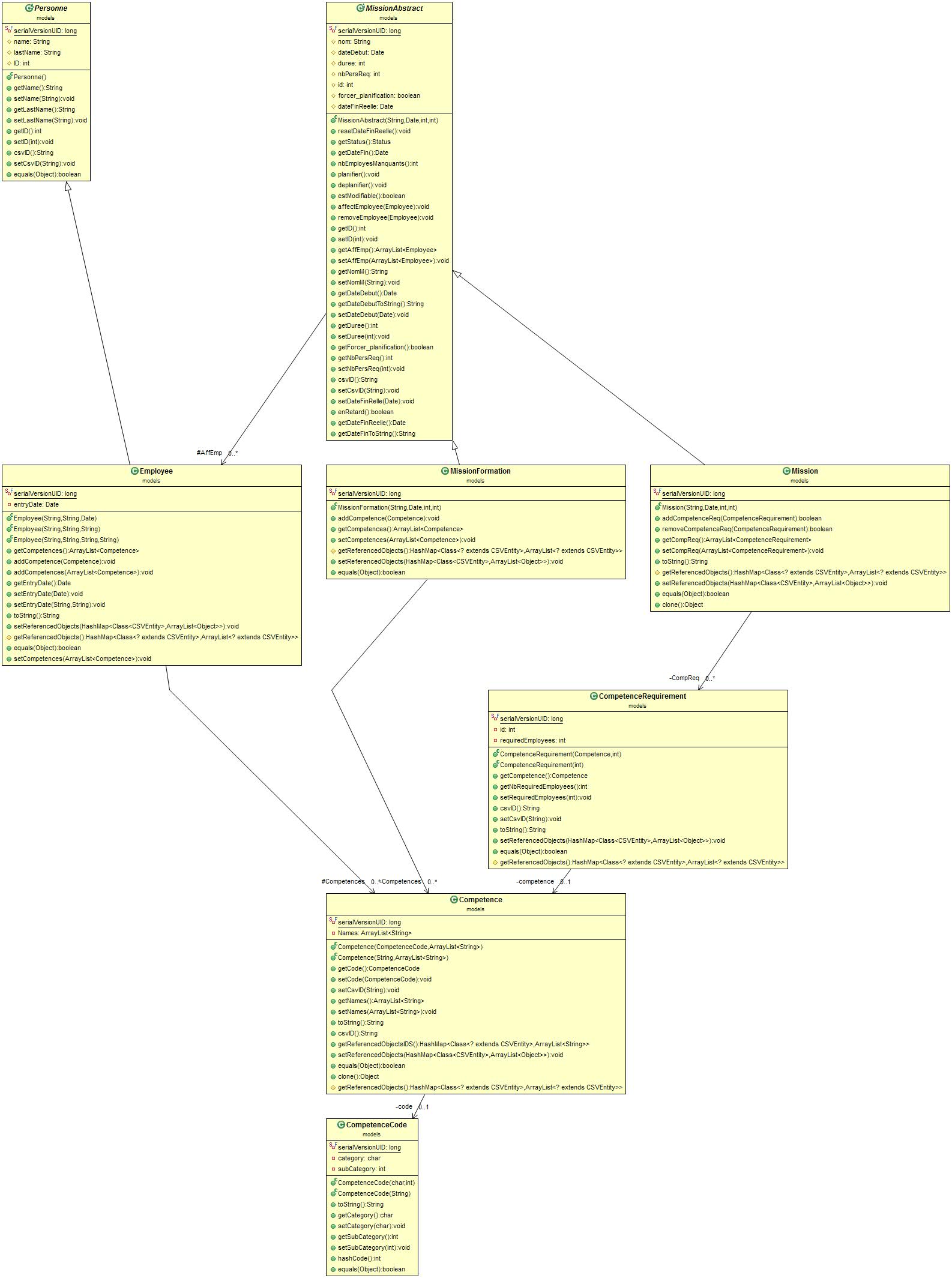
Package « Navigation » : Panneaux accueil et gestion des missions.



Package « Navigation » : Panneaux de gestion des compétences, du personnel, et des formations.



Package « GUI » : Formulaires et boutons de navigation



Package « Models » : Classes de données

Package « Models » : Système de recommandation des employés pour une mission

A) Présentation :

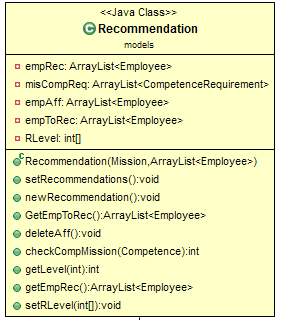
Le principe des recommandations est simple, il s’agit lors de l’affection des employés aux missions de dire quels sont les personnes qui sont le plus aptes à travailler sur cette mission

B) Principe :

L’algorithme qui nous sert à déterminer qui sont ces employés se base sur les compétences qui sont nécessaires et les employés qui les ont. Il est aussi calculer un degré de vérification pour le non gaspillage des ressources.

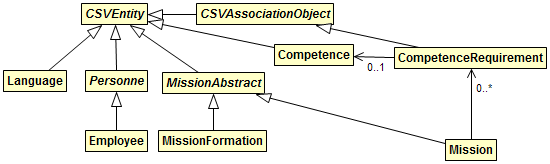
C) Fonctionnement :

On regarde combien de compétences ont les employés au total, on vérifie ensuite quels sont leur compétences puis on les compare aux compétences nécessaires sur la mission et nous prenons le rapport entre les compétences totales de l’employés, les compétences nécessaires qu’a l’employé et une fois convertie en pourcentages est transmis à l’interface pour donner un pourcentage de correspondance entre l’employé et les missions.



D) Algorithme :

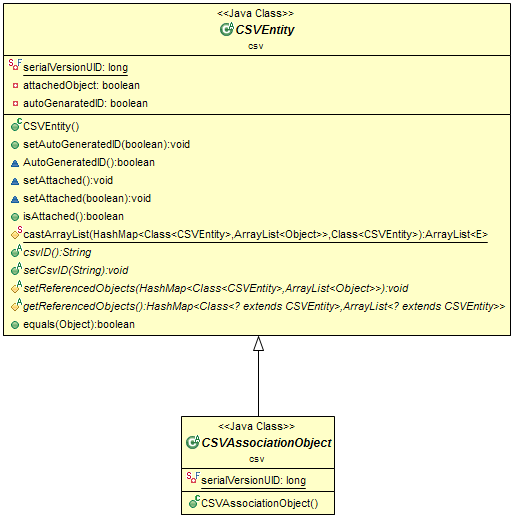




Persistance des données

**CSVEntity** : Classe parente de toutes les classes d’entités du programme.

**CSVAssociationObject** : Classe parente de toutes les classes d’associations ayant au moins une propriété.



Ce système permet de gérer et synchroniser automatiquement :

- Les dépendances entre les objets de données chargés en mémoire (grâce aux méthodes de CSVEntity)

- leur représentation dans les fichiers CSV

- la mise en « cache » (objets stockés par référence)

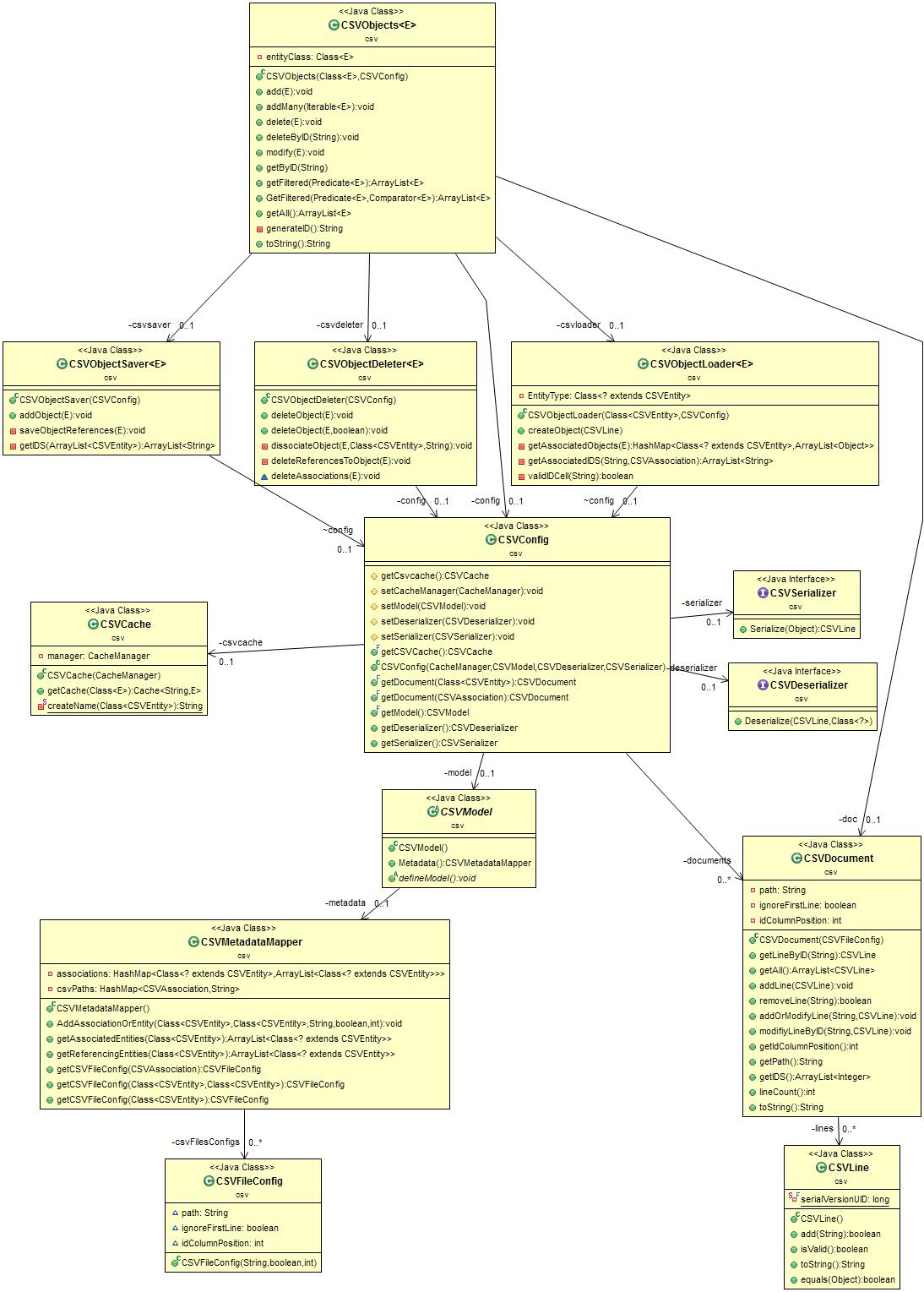
CSVObjects fournit les méthodes de lecture / écriture des objets de données et gère le cache. Les informations nécessaires pour le bon fonctionnement du package sont contenus dans CSVConfig :

- CSVModel qui contient les relations entre les classes d’entité et les fichiers CSV correspondants

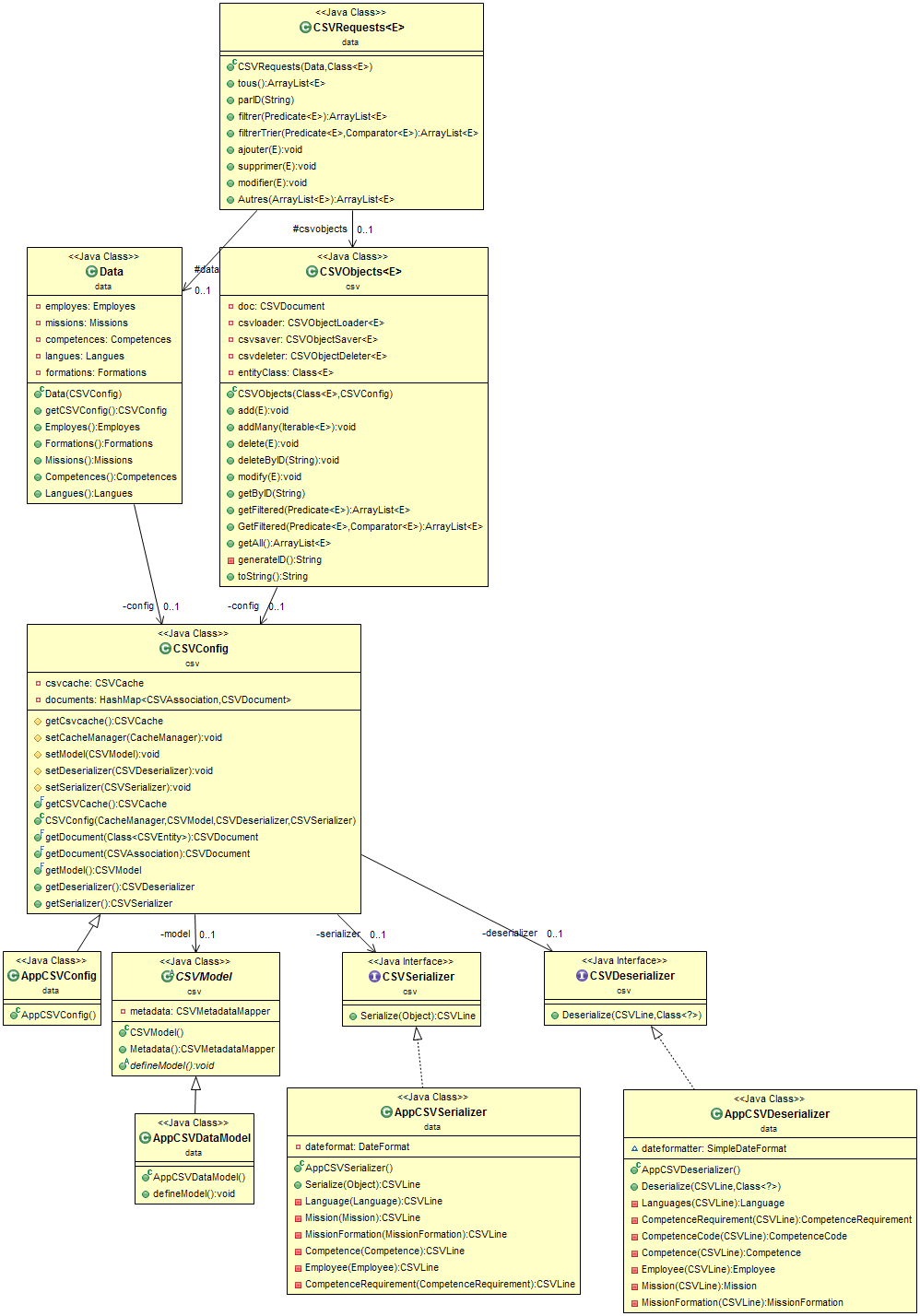
- Sérialiseurs et désérialiseurs des objets

- Gestionnaire de cache

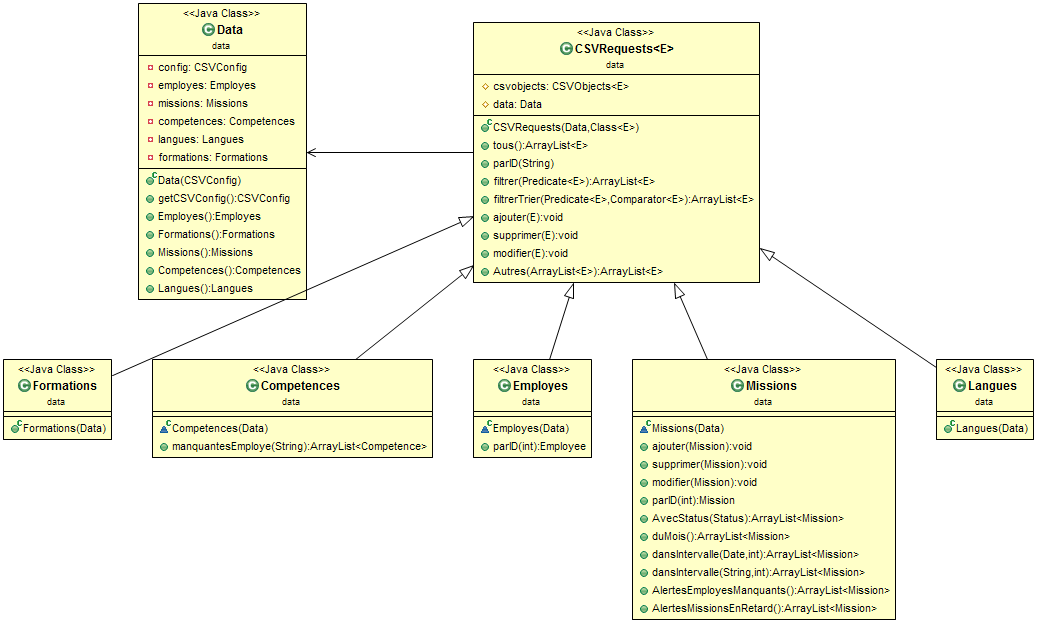
Cette configuration est encapsulé par l’objet Data qui est créé dans la classe Main et injecté par constructeur à toutes les classes qui dépendent des données persistantes.



Package « CSV »



Package « Data » : Configuration du modèle et de la sérialisation des données de l’application



Package « Data » : Interface d’accès aux données