Concepts Objet et Java

1ère année du Cycle Ingénieur en Informatique
Année 2024-25

Projet : Vigie des médias

Contexte

Les médias français sont souvent possédés par des groupes industriels ou des individus. Un groupe industriel peut lui-même être partiellement possédé par un autre groupe ou un individu. L'objectif est de pouvoir déterminer qui possède et/ou contrôle un média donné.

Objectifs du Projet

1. **Modélisation des Entités** : Créer des entités (personnes, groupes, médias) et leurs relations.
2. **Manipulation de Données Ouvertes** : Utiliser des données ouvertes pour alimenter l'application.
3. **Représentation et Traitement des Collections** : Opérer des traitements simples sur les collections de données.
4. **Système de Transmission de Messages** : Mettre en place un système de programmation événementielle.

Première approche

Le projet a débuté par l'analyse du cahier des charges et l'exploration du jeu de données fourni. Après avoir compris les enjeux, j'ai esquissé un premier diagramme de classes (voir Annexe 1). Ce diagramme a guidé mes premières implémentations, mais j'ai dû réviser plusieurs choix conceptuels au fil du développement, notamment en ce qui concerne les relations entre les entités.

Conception des classes

J'ai commencé par créer un package model regroupant toutes les classes liées aux entités. La classe de base *Entite* contient un nom et deux variables comptabilisant les organisations et/ou médias possédés. Initialement abstraite, elle a finalement été rendue concrète pour des raisons de compatibilité avec d'autres composants, notamment les modules.

Ensuite, j'ai implémenté les trois principales sous-classes d'*Entite : Media, GroupInd et Personne.* Ces classes sont relativement simples, avec quelques méthodes supplémentaires ajoutées pour permettre le tri ou l'affichage.

Pour modéliser les liens de propriété, j'ai d'abord suivi le diagramme UML en utilisant des chaînes de caractères, mais cela s'est révélé peu pratique pour les modules. J'ai alors conçu une classe *Proprietaire paramétrée*, ce qui m'a permis de relier les entités de façon typée et robuste. J'ai décliné cette classe en plusieurs sous-classes selon les types d'entités concernées.

Importation des données

J'ai ensuite créé une classe *Importer* (hors du package model) chargée de lire les fichiers de données. Je me suis appuyé sur les exemples du site *CodeGym* pour la lecture de fichiers, en les adaptant à mon projet.

Un point faible reste que les méthodes d'importation ne sont pas généralisées : j'ai dû écrire une méthode par type de données, ce qui entraîne beaucoup de redondance.

Simulation d'événements

J'ai créé deux classes pour gérer les événements :

- ***Publication*** (dans model) représente une publication faite par un média. J'ai cette fois directement utilisé des types comme *Media* ou *Entite* pour éviter les erreurs précédentes.
- ***SimulationEvenement*** (hors du model) contient deux méthodes principales : *Publication() et Rachat().*

Méthode Publication

Elle simule l'émission d'une publication à partir des entrées utilisateur (auteur, média, contenu, mentions). Ce développement m'a permis de découvrir deux concepts en Java *:*

- ***Optional,*** utilisé pour simplifier certaines recherches, comme expliqué sur *laulem.com*.
- ***Les flux (streams),*** pour filtrer les données d'une collection, en m'inspirant du guide de *J.M. Doudoux*.

Méthode Rachat

Elle permet de simuler le rachat de parts d'un média ou d'une organisation par un autre acteur. Deux défis se sont posés :

1. Identifier un vendeur possédant suffisamment de parts.
2. Gérer le cas où l'acheteur n'a pas encore de lien de propriété. Ce point est discutable car le sujet précise « la liste des entités ne peut pas évoluer ». Mais de mon point de

vue, si une entité achète des parts, elle devient forcément propriétaire et doit être ajoutée.

J'ai donc implémenté ce comportement, en cherchant l'entité acheteuse dans les listes existantes, en générant un nouvel ID, et en créant une nouvelle instance de *Proprietaire.*

Modules et système de surveillance

Le cœur du projet réside dans la mise en place des modules de surveillance. J'ai créé un package modules avec notamment la classe *Vigie,* simple mais centrale, qui gère les alertes et conserve un historique.

En m'appuyant sur le **patron Observateur**, j'ai créé une interface *ModuleSpecialise* définissant deux méthodes :

- *traiterEvenement()* pour les publications,
- *traiterRachat()* pour les opérations financières.

J'ai ensuite implémenté deux modules :

- *SuiviPersonne* pour surveiller les mentions et acquisitions par des personnes.
- *SuiviMedia* pour surveiller les médias eux-mêmes.

Ces classes incluent aussi des méthodes d'aide comme *estProprietaire(), afficherStats() et* des méthodes de création.

Console et intégration

Pour relier toutes les briques, j'ai développé la classe *Console*, qui constitue l'interface utilisateur. Elle gère l'affichage, l'importation des données et l'accès à toutes les fonctionnalités. Cette classe est moins élégante dans sa structure, car principalement constituée de *System.out.println()* et de logique de navigation. Et petite information, lorsque que vous voulez utiliser une fonction proposée par la console il faut entrer le numéro affiché à côté de celle-ci. Concernant les entées de noms, prénoms de personnes, médias ou organisations, il faut juste faire attention aux espaces et aux accents, normalement il n'y a pas de casse avec les majuscules car j'ai utilisé *equalsIgnoreCase().*

Exécution du programme

Extraction des fichiers :
Tout d'abord, vous devrez extraire le dossier contenant les fichiers de mon projet ainsi que ce rapport. Dans le dossier *Projet_final*, vous trouverez l'exécutable nommé *Projet_final.jar*.

Exécution de l'exécutable :
Vous avez deux options pour exécuter le programme :

- **Double-cliquer sur l'exécutable :** Vous pouvez simplement double-cliquer sur le fichier *Projet_final.jar* pour lancer le programme.
- **Utilisation de l'invite de commande :**
  Si vous préférez utiliser l'invite de commande, vous devrez naviguer vers le dossier *Projet_final*. Par exemple, si le dossier se trouve sur votre bureau, vous pouvez utiliser la commande suivante : cd Desktop\Java\PAOLO BRACQ\Projet_final

Vérification de la version de Java :
Il est important de vérifier que vous disposez de la version correcte de Java, car j'ai développé le projet avec **Java 23**. Les versions antérieures de Java pourraient entraîner des problèmes de compatibilité. Vous pouvez vérifier votre version de Java en utilisant la commande suivante : java -version

Exécution du projet :
Une fois dans le bon répertoire et avec la bonne version de Java, vous pouvez exécuter le projet en utilisant la commande suivante : java -jar Projet_final.jar

Le programme devrait alors se lancer, et le **Menu Principal** s'affichera, bien qu'il puisse également afficher des messages concernant certaines entités non trouvées.

Points à considérer :

- Lors de l'utilisation du programme, il est nécessaire de **saisir des entiers** lorsque cela est demandé, car le scanner ne reconnaît pas les chaînes de caractères (String) lorsque des entiers sont attendus.
- L'affichage dans l'invite de commande peut ne pas prendre en charge tous les caractères spéciaux, ce qui peut entraîner l'apparition de points d'interrogation à la place de certains caractères.

Mon interprétation du projet

J'aimerais mettre en avant certains choix que j'ai dû effectuer en raison d'un manque de précisions dans la documentation.

Tout d'abord, concernant le jeu de données, il est mentionné que « la liste des entités ne peut pas évoluer ». J'ai donc interprété cela comme l'impossibilité d'ajouter manuellement de nouvelles personnes, médias ou organisations. De ce fait, j'ai également décidé que, pour qu'un rachat de parts soit possible, l'acheteur devait impérativement faire partie des entités déjà présentes dans les données. Par exemple, si l'on tente un rachat avec "Tintin" en tant qu'acheteur alors qu'il n'est pas présent dans les données, l'action sera bloquée.

Ensuite, en cas de relation de propriété comme « organisation-média », j'ai imposé que l'organisation concernée figure impérativement dans les données des organisations. Dans le cas contraire, un message d'erreur est affiché avant même l'apparition du menu de l'application.

Enfin, concernant le format de rendu attendu « un fichier archive avec l'ensemble des sources et une version exécutable de l'application », je n'étais pas certain de l'interprétation. J'ai donc choisi de fournir un fichier compressé contenant l'ensemble de mes codes sources ainsi qu'un fichier .JAR, que j'ai compris comme étant un exécutable compatible avec les environnements Java.

Conclusion

Ce projet m'a permis de consolider mes compétences en programmation orientée objet avec Java, tout en explorant des concepts essentiels comme les flux de données, le patron Observateur, et la structuration modulaire d'une application. Malgré certaines imprécisions dans le cahier des charges, j'ai pris des décisions raisonnables pour assurer la cohérence du système. Ce travail m'a également donné une meilleure compréhension des enjeux liés à la modélisation de données complexes et à l'intégration d'interfaces simples mais fonctionnelles.

Annexe 1 : Diagramme de classe