Rapport de projet de génie logiciel

JACQUINOT Angéline - GILLET Annabelle

Cahier des charges

Nous souhaitons produire une application qui permet de récupérer des informations d'une base de données pour l'enregistrer dans un fichier et inversement. Pour cela, nous avons besoin de plusieurs fonctionnalités.

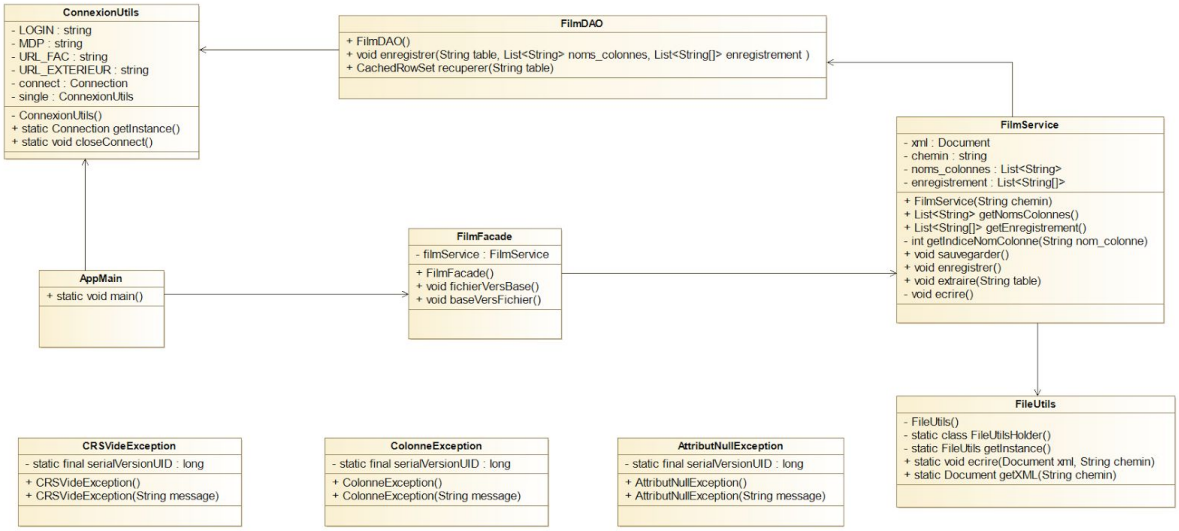
Nous devons tout d'abord mettre en place une connexion à la base de données, ainsi qu'une méthode pour lire et enregistrer les données de la base qui nous intéresse. Nous allons également avoir besoin de méthodes permettant d'écrire et d'extraire les données du fichier.

Il faudra ensuite mettre en lien ces méthodes dans le but d'obtenir l'application demandée.

Concernant les informations stockées, nous utilisons un fichier XML et une base de données Oracle.

Dossier d’analyse

Diagramme des classes



Dans ce diagramme de classes nous pouvons voir les liens entre les différentes classes créées pour ce projet. La classe ‘AppMain’ va permettre le lancement de l’application, elle fait appelle à ‘FilmFacade’ qui fera appel aux méthodes fichierVersBase() et baseVersFichier() qui décideront du sens dans lequel seront récupérées les données.

La classe ‘ConnexionUtils’ permet d’effectuer une connexion avec la base de données.

La classe ‘FilmService’ sera donc sollicitée car c’est elle qui est chargée du lien entre la base de données et le fichier XML. Les classes appelées sont respectivement ‘FilmDAO’ qui va communiquer avec la base de données et ‘FileUtils’ pour la gestion du fichier XML.

Ce diagramme présente également trois classes exceptions, créées afin de gérer certaines erreurs particulières à notre application.

La base de données

Notre base de données se construisant en fonction des éléments contenus dans notre fichier XML, il est prévu qu’elle ne reste pas forcément avec la même structure. Cependant pour les premiers enregistrements, nous possédons une table GL_film dans notre base : un attribut id de type INTEGER, un attribut titre de type VARCHAR2(50) et un attribut durée de type INTEGER.

Puisque nous recréons notre table en fonction du contenu de notre fichier, nous n'avons pas posé de contraintes. Cependant avec une évolution du projet, en utilisant les différentes fonctionnalités disponibles dans un fichier XML, nous pourrions préciser ces contraintes éventuelles, ainsi que le type des attributs plutôt que de les enregistrer uniquement en VARCHAR2 (voir FilmDAO). Nous avons donc préféré garder la flexibilité des enregistrements plutôt que de pouvoir poser des contraintes sur notre base de données.

Dossier de programmation

L'architecture

La programmation de cette application a été effectuée en plusieurs couches. Nous avons donc une couche d'utilitaires (FileUtils pour le fichier XML et ConnexionUtils pour la communication avec la base de données), une couche d'accès à la base de données (FilmDAO), une couche service permettant de faire le lien entre le fichier XML et la base de données (FilmService) ainsi qu'une façade permettant le lancement des méthodes nécessaires aux fonctionnalités de l'application.

Les classes

ConnexionUtils

Ce singleton sert à ouvrir et fermer la connexion à la base de données. Il teste la connexion depuis l'Université et, si cette connexion échoue, depuis l'extérieur de l'Université. Une méthode est également disponible pour fermer l'application lors que l'application se termine.

FileUtils

Cette classe est également un singleton. Elle permet d'extraire le contenu XML d'un fichier situé à un emplacement donné, et d'enregistrer un document XML à un emplacement, les deux étant fournis en paramètres.

FilmDAO

Les communications avec la base de données se font depuis cette classe. Elle permet donc de récupérer les tuples d'une table dont le nom est donné sous la forme d'un CachedRowSet, et d'enregistrer les données préalablement extraites du fichier XML dans la base.

Nous avons essayé d'automatiser au maximum ce procédé. Nous fournissons donc en paramètres la liste des noms de colonnes ainsi que les données et le nom de la table que nous souhaitons remplir. L'enregistrement est faisable même si les noms d'attributs que nous souhaitons ajouter sont différents de ceux qui ont pu être présents en base de données. Pour cela, nous supprimons la table si elle existe, puis nous la recréons avec les intitulés d'attributs qui nous intéressent. Pour nous affranchir des types, nous les déclarons tous en temps que VARCHAR2(100).

Nous préparons tout d'abord notre insertion avec la liste de noms de colonnes dans un PreparedStatement, puis pour chaque enregistrement nous insérons les valeurs correspondantes avant d'exécuter la requête.

FilmService

Cette classe sert à faire le lien entre l'extraction des données de la base de données et leur enregistrement dans le fichier, ou entre la récupération des données du fichier XML pour les sauvegarder dans la base de données.

Pour faire copier les enregistrements de la base de données au fichier XML, nous récupérons le CachedRowSet depuis FilmDAO. Nous récupérons ensuite le nom des colonnes grâce à ses métadonnées. En parcourant ensuite les tuples contenus dans le CachedRowSet, nous les ajoutons comme un noeud XML avec comme texte la valeur de l'attribut pour ce tuple. Une fois que le document XML est terminé, nous l'enregistrons grâce à la classe FileUtils.

En ce qui concerne la récupération des données depuis le fichier XML pour les insérer en base de données, nous parcourons les noeuds de notre fichier XML afin de vérifier son contenu et de le convertir en un format utilisable par notre classe FilmDAO (c'est-à-dire une liste de noms de colonnes et une du contenu des enregistrements).

Nous cherchons donc tous les noeuds s'appelant "film" afin de parcourir leurs éléments. Lorsque nous trouvons le premier élément "film", nous récupérons le nom de ses noeuds afin de pouvoir créer la liste des noms de colonnes. Nous récupérons également dans une autre liste le texte contenu dans les noeuds, ce qui nous servira pour l'enregistrement dans la base de données comme décrit dans la classe FilmDAO.

FilmFacade

Cette classe sert à appeler les fonctions servant à récupérer nos données (de la base vers le fichier ou dans le sens inverse) dans le bon ordre et avec les bons paramètres.

AppMain

C'est depuis cette classe que le lancement de l'application s'effectue. Une JOptionPane permet de choisir l'action à effectuer (extraction des données depuis la base ou depuis le fichier), et permet d'exécution avec les méthodes de la classe FilmFacade.

Les classes d'exceptions

Afin de gérer au mieux les différentes erreurs que notre application peut rencontrer, nous avons créé des exceptions personnalisées. Nous en avons donc une pour savoir si un CachedRowSet ne contient pas de résultats, une pour savoir si un noeud du fichier XML ne possède pas de texte, et une pour savoir si une différence ou un problème est rencontré lors du parcours des attributs d'un film depuis le fichier XML.

Les tests

Les tests unitaires mis en place permettent de reconnaître la plupart des erreurs que notre application peut rencontrer. Ces tests ont été réalisés sur les classes ConnexionUtils, FileUtils, FilmService et FilmDAO.

ConnexionUtils ne comprend qu'un seul test : il vérifie que la connexion récupérée comme instance du singleton ne soit pas nulle.

FileUtils a besoin de plus de tests. Avant de les lancer, nous créons des fichiers qui vont nous servir à tester les différents cas de figure rencontrés. Nous avons donc un fichier XML construit correctement ainsi qu'un fichier texte vide. Nous testons ensuite l'écriture et la lecture pour voir si les exceptions qui sont renvoyées correspondent à celles que nous attendons.

Pour FilmDAO, nous testons les accès à la base de données (en lecture ou en écriture). Là aussi nous créons un fichier XML valide avec de s'en servir pour nos tests, ainsi qu'une table sans tuple dans la base de données. Nous lancons ensuite la sauvegarde ainsi que la récupération des données dans différents cas.

Enfin, FilmService a des tests plus orientés sur le parcours du contenu du fichier XML. Les méthodes de communication avec la base de données étant déjà testées dans FilmDAOTest, nous n'avons pas besoin de le refaire ici. Là encore nous créons plusieurs fichiers XML avant de lancer l'exécution de nos tests : un avec un attribut qui n'a pas de valeur, un avec un attribut répété plusieurs fois, un qui ne possède pas de noeud "film" et un correctement formé. Nous testons alors notre méthode de sauvegarde, qui parcourt le contenu du fichier XML afin de le transposer dans un format plus simple à gérer pour l'insertion en base de données.

Bilan de l’implémentation

Ce projet nous a permis de mettre en place une application structurée ainsi que des tests permettant de contrôler le bon fonctionnement de l’application. Cela a été très intéressant d’implémenter ces concepts puisque nous ne les avions que très peu vu, et encore moins mis en place.