Projet C++ M2 MIAGE IF - Apprentissage 2022/2023

Professeur: Mathlouthi Emna
Groupe: Mohamed Amine Tabbakh, Rafik Hadjadj

1 Introduction

Le projet C++ de notre programme de Master MIAGE IF pour l'apprentissage de l'année 2022/2023 vise à mettre en pratique les concepts de financement structuré. Ce projet nous permettra de comprendre et d'implémenter les mécanismes de partage de financement entre différentes entités financières. Dans cette introduction, nous définirons l'objectif du projet.

2 Objectif du projet

L'objectif principal de ce projet est de créer un système de gestion de financement structuré en utilisant le langage de programmation C++. Nous allons concevoir et implémenter différentes classes et structures pour représenter les entités impliquées dans le processus de financement structuré, telles que les banques, les compagnies aériennes. Nous allons également mettre en place les fonctionnalités nécessaires pour gérer les Facilities (tranches de financement) et les Parts (remboursements). L'objectif final est de créer une application robuste qui permettra de gérer plusieurs deals, lenders, borrowers, facilities et parts.

3 Description du projet

3.1 Concept de financement structuré

Le financement structuré est un mécanisme utilisé lorsque les banques ne sont pas en mesure de financer entièrement un projet. Dans ces cas, elles font appel à d'autres institutions financières pour partager le financement. Ce concept permet de répartir le risque et de mobiliser des ressources supplémentaires pour soutenir des projets d'envergure.

3.2 Exemple de cas : compagnie Air France et banque CACIB

Prenons l'exemple de la compagnie aérienne Air France qui souhaite financer l'achat d'un avion. Étant donné le montant élevé de cette opération, la banque CACIB intervient en tant qu'agent principal (Agent) pour coordonner le financement. La banque CACIB forme un pool (Pool) en sollicitant d'autres banques pour participer au financement.

3.3 Rôles des différentes entités (Agent, Pool, Borrower, Lenders)

- Agent (Agent) : La banque CACIB, en tant qu'agent principal, gère le contrat (Deal) avec la compagnie Air France.

- Pool (Pool) : Les autres banques qui participent au financement forment le pool, partageant ainsi le risque et la responsabilité.
- Emprunteur (Borrower) : La compagnie Air France, qui emprunte les fonds nécessaires pour l'achat de l'avion.
- Prêteurs (Lenders) : Les banques, y compris CACIB, qui fournissent les fonds pour le financement.
- Facilities (Facilities) : Les tranches de financement débloquées par le pool, où un ou plusieurs prêteurs peuvent participer.
- Parts (Parts) : Les remboursements effectués par l'emprunteur (Air France) sur chaque tranche de financement.

3.4 Fonctionnement des Facilities et des Parts

Les Facilities représentent les tranches de financement débloquées par le pool. Chaque tranche a une période spécifiée et peut impliquer un ou plusieurs prêteurs. Les intérêts sont calculés selon un mode de calcul expliqué par la suite. Les remboursements effectués par l'emprunteur (Air France) sur chaque tranche sont appelés Parts. Les Parts réduisent le montant restant de la tranche.

3.5 Refund et calcul des taux

Une part correspond au remboursement d'une partie d'une facility. Toutes les parts sont référencées dans le portfolio du Borrower. Ainsi, lorsque le Borrower décide d'émettre une part pour rembourser une partie d'une facility, cette dernière est ajoutée à son portfolio.

Tout d'abord, il faut calculer le montant d'intérêt à payer. Pour cela, nous utilisons la méthode `refund` de la facility. Ainsi, nous calculons le taux d'intérêt à payer selon la formule suivante :

$$\frac{montantdelapart \times tauxd'intérêtdelafacility}{montantinitialnominaldelafacility}$$

, afin de maintenir une proportionnalité des taux.

Une fois le taux à payer calculé, le montant d'intérêt à payer pour cette part correspond simplement à ce taux multiplié par le montant de la part.

Automatiquement, nous diminuons le solde restant dû (outstanding balance) de la facility du montant remboursé par la part. Pour maintenir une cohérence globale, nous utilisons des références via des pointeurs.

Nous savons qu'un Deal est considéré comme clos ou "Finish" lorsque les soldes restants dûs (outstanding balances) de toutes les facilities qui le composent sont égaux à 0. Cela signifie que le Borrower a émis des parts permettant de rembourser tous les montants nominaux ainsi que les intérêts correspondants.

Enfin, le taux d'intérêt de la facility doit être ajusté car le solde restant dû (outstanding balance) a logiquement diminué. Afin de maintenir l'équilibre des

taux, nous utilisons la formule suivante : le nouveau taux d'intérêt de la facility sera égal à

$$\frac{ancientauxd'intérêt \times nouveausolderestantdû}{montantnominalinitialdelafacility}$$

.

3.6 Classes

3.6.1 Classe Borrower

Cette classe représente un emprunteur dans le projet.

Propriétés :

- `name` : une chaîne de caractères représentant le nom de l'emprunteur.
- `portfolio` : un objet de la classe Portfolio représentant le portefeuille de l'emprunteur.

Méthodes :

- `getName()` : renvoie le nom de l'emprunteur.
- `getPortfolio()` : renvoie le portefeuille de l'emprunteur.
- `addPart(Part p)` : ajoute une part (objet de la classe Part) au portefeuille de l'emprunteur.

3.6.2 Classe Deal

Cette classe représente un contrat dans le projet.

Propriétés :

- `contractNumber` : une chaîne de caractères représentant le numéro du contrat.
- `agent` : un objet de la classe Lender représentant l'agent principal du contrat.
- `pool` : un vecteur d'objets de la classe Lender représentant le pool des prêteurs.
- `borrower` : un objet de la classe Borrower représentant l'emprunteur du contrat.
- `projectAmount` : un double représentant le montant du projet.
- `currency` : une chaîne de caractères représentant la devise du contrat.
- `contractStartDate` : une chaîne de caractères représentant la date de début du contrat.

- `contractEndDate` : une chaîne de caractères représentant la date de fin du contrat.
- `dealStatus` : une chaîne de caractères représentant l'état du contrat.
- `facilities` : un vecteur de pointeurs vers des objets de la classe Facility.

Méthodes :

- `getContractNumber()` : renvoie le numéro du contrat.
- `getAgent()` : renvoie le nom de l'agent principal du contrat.
- `getPool()` : renvoie le pool des prêteurs.
- `getBorrower()` : renvoie le nom de l'emprunteur du contrat.
- `getProjectAmount()` : renvoie le montant du projet.
- `getCurrency()` : renvoie la devise du contrat.
- `getContractStartDate()` : renvoie la date de début du contrat.
- `getContractEndDate()` : renvoie la date de fin du contrat.
- `getDealStatus()` : renvoie l'état du contrat.
- `getFacilities()` : renvoie les facilities du deal.
- `totalRefund()` : met à jour l'état du contrat.

3.6.3 Classe Facility

Cette classe représente la classe facility dans le projet.

Propriétés :

- `startDate` : une chaîne de caractères représentant la date de début de la classe Facility.
- `endDate` : une chaîne de caractères représentant la date de fin de Facility.
- `trancheAmount` : un double représentant le montant de la tranche de Facility.
- `currency` : une chaîne de caractères représentant la devise de Facility.
- `lenders` : un vecteur d'objets de la classe Lender représentant les prêteurs de Facility.
- `interestRate` : un double représentant le taux d'intérêt de Facility.
- `outStandingBalance` : un double représentant le solde impayé de Facility.

Méthodes :

- `getStartDate()` : renvoie la date de début de Facility.
- `getEndDate()` : renvoie la date de fin de Facility.
- `getTrancheAmount()` : renvoie le montant de la tranche de Facility.
- `getCurrency()` : renvoie la devise de Facility.
- `getLenders()` : renvoie les prêteurs de Facility.
- `getInterestRate()` : renvoie le taux d'intérêt de Facility.
- `getOutstandingBalance()` : renvoie le solde impayé de Facility.
- `refund(double amount)` : effectue un remboursement sur Facility et renvoie le montant payé en intérêts.

3.6.4 Classe Lender

Cette classe représente un prêteur dans le projet.

Propriétés :

- `name` : une chaîne de caractères représentant le nom du prêteur.

Méthodes :

- `getName()` : renvoie le nom du prêteur.

3.6.5 Classe Part

Cette classe représente une part dans le projet.

Propriétés :

- `amount` : un double représentant le montant de la part.
- `interest_payed` : un double représentant les intérêts payés sur la part.
- `facility` : un pointeur vers un objet de la classe Facility représentant Facility liée à la part.

Méthodes :

- `getAmount()` : renvoie le montant de la part.
- `getInterest()` : renvoie les intérêts payés sur la part.

3.6.6 Classe Portfolio

Cette classe représente un portefeuille dans le projet.

Propriétés :

- `parts` : un vecteur d'objets de la classe Part représentant les parts du portefeuille.

4 Conclusion

En conclusion, ce projet de gestion de financement structuré en C++ met en œuvre plusieurs classes qui permettent de modéliser les entités et les fonctionnalités liées à ce domaine. Les classes telles que `Borrower`, `Deal`, `Facility`, `Lender` et `Portfolio` sont utilisées pour représenter les acteurs impliqués dans le processus de financement, les contrats, les facilities, les prêteurs et les portefeuilles respectivement.

L'objectif principal du projet est de faciliter la gestion des contrats de financement structuré en fournissant des fonctionnalités telles que l'ajout de parts, le suivi des soldes, le calcul des intérêts et la gestion des dates.

Le concept de financement structuré est mis en évidence à travers l'exemple de cas où une compagnie aérienne (Air France) et les Banques (tel que CACIB, BNP...) sont impliquées.

Ce projet nous a permis de mettre en oeuvre les différentes compétences techniques apprises lors de ce semestre.