[Prénom Nom] [Matricule]

[Prénom Nom] [Matricule]

[Prénom Nom] [Matricule]

[Prénom Nom] [Matricule]

Patrick Limoges, limp2601

[Prénom Nom] [Matricule]

Projet en programmation agile

Jeu d’échec

Rapport Équipe

Dans le cadre du cours IFT232

Université de Sherbrooke

[date]

Table des matières

[Introduction 3](#_Toc469422824)

[Patron de conception 6](#_Toc469422825)

[1. Factory 6](#_Toc469422826)

[2. Singleton 6](#_Toc469422827)

# Introduction

Pour le cadre du projet finale de programmation orientée objet, notre équipe a décidé de programmer et présenter un jeu d’échec. Ce jeu d’échec permettra à deux joueurs de jouer sur un même ordinateur, suivant les règles standards des échecs, tout en y intégrant quelques ajouts, comme la sauvegarde et le chargement de partie.

Pour ce rapport, nous commencerons par présenter l’architecture du jeu d’échec, celle-ci étant présenté grâce à un diagramme, construit selon les standard UML.

En deuxième lieu, nous vous présenterons l’analyse initiale du projet, puis nous la comparerons avec l’analyse de la version finale du projet, pour présenter un compte rendu du travail effectué.

Suivant cela, nous présenterons les différents patrons de conception utilisés dans le projet, incluant une analyse de leurs utilités dans la globalité du projet.

Nous nous attarderons quelque peu sur les différents logiciels utilisés, pour présenter notre méthode de travail.

Puis, pour vous permettre d’appréciez au maximum notre jeu, nous inclurons un petit paragraphe afin de vous permettre d’effectuer une partie d’échec sans désagrément.

Nous finirons alors, par une conclusion qui fera la synthèse du projet, ainsi que de ci document.

# Patron de conception

Dans cette section du rapport, nous feront l’analyse des différents patrons de conception, utilisés pour résoudre des problèmes ou simplifier l’entretien du jeu.

Nous commencerons donc par démontrer l’efficacité de l’utilisation du patron de la fabrique, couplé à celle du singleton, dans la classe « PieceFactory ».

Puis nous aborderons le patron de conception des commandes utilisées pour simplifier l’exécution des différents boutons de la barre d’outils du jeu.

## Factory

Le premier patron de conception analysé est celui de la fabrique. Nous l’abordons en premier, car elle est très rapidement distinguable face aux autres patrons de conception, du fait que la classe qui l’utilise porte le nom du patron : « PieceFactory ».

Le fait d’utilisé le patron de la fabrique peut sembler être trivial, mais aussi semblé très inutiles.

En effet, dans le cadre d’un jeu d’échec, lorsqu’une nouvelle partie est créée, c’est toujours les mêmes pièces, disposées au même endroit, qui sont placées sur le damier. Cependant, la création de pièce ne s’arrête pas là.

Comme dit antérieurement, notre jeu d’échec supporte le chargement de partie, ce qui veut dire que l’on peut avoir de nouvelle partie dont le nombre de pièce peuvent différés. De plus, cela serait oublier un règle du jeu d’échec – la promotion – qui permet à une pièce de changer.

De ce fait, l’utilisation du patron de conceptions de la fabrique est pleinement justifiée, car la création de pièce peut varier et survenir à n’importe quel moment dans la partie.

## Singleton

Fortement couplé avec la fabrique, il va de soit de le présenter un deuxième. De plus, comme cette dernière, l’utilisation principale du singleton est faite dans la classe : « PieceFactory ».

L’utilisation de ce patron de conception est fortement liée à la conception de la classe qui l’utilise. En effet, lors de la conception de « PieceFactory », notre équipe a déterminé qui serait bon d’y implanter ce patron de conception dans un but future.

En effet, telle qu’il est maintenant, le jeu d’échec ne permet que d’avoir un seul damier d’ouvert à la fois, ce qui rend l’application du singleton inutile, car une instance de « PieceFactory » dans la classe « Board » est suffisant.

Cependant, à cause de l’évolution naturelle du projet, nous avons décidé de placer la classe « PieceFactory » en singleton pour ne pas avoir une redondance inutile entre les instances de « Board » qui utilisent tous exactement la même instance de « PieceFactory ».

Donc cela justifie pleinement l’utilisation du singleton.

## Command

L’utilisation du patron de conception de la commande s’est fait tardivement dans le projet, mais n’est pas totalement dénué de sens.

En effet, nous avons déterminé, lors de la création de notre barre d’outils qu’il serait intéressant d’ajouter à cela des raccourcis clavier. Cependant, nous ne désirions pas réécrire le code déjà mis en place, donc nous avons préféré mettre en place le patron de conceptions des commandes.

Certes, dû au petit nombre d’outils mis à la disposition du joueur, cela peut sembler obsolète, cependant une évolution du jeu est prévisible.

De ce fait, si le jeu augmente de taille, nécessitant l’ajout d’outils dans la barre d’outils, l’implémentation du patron de conception se retrouvera rentabilisée.