S3 Période B

PROJET JEU DE GO

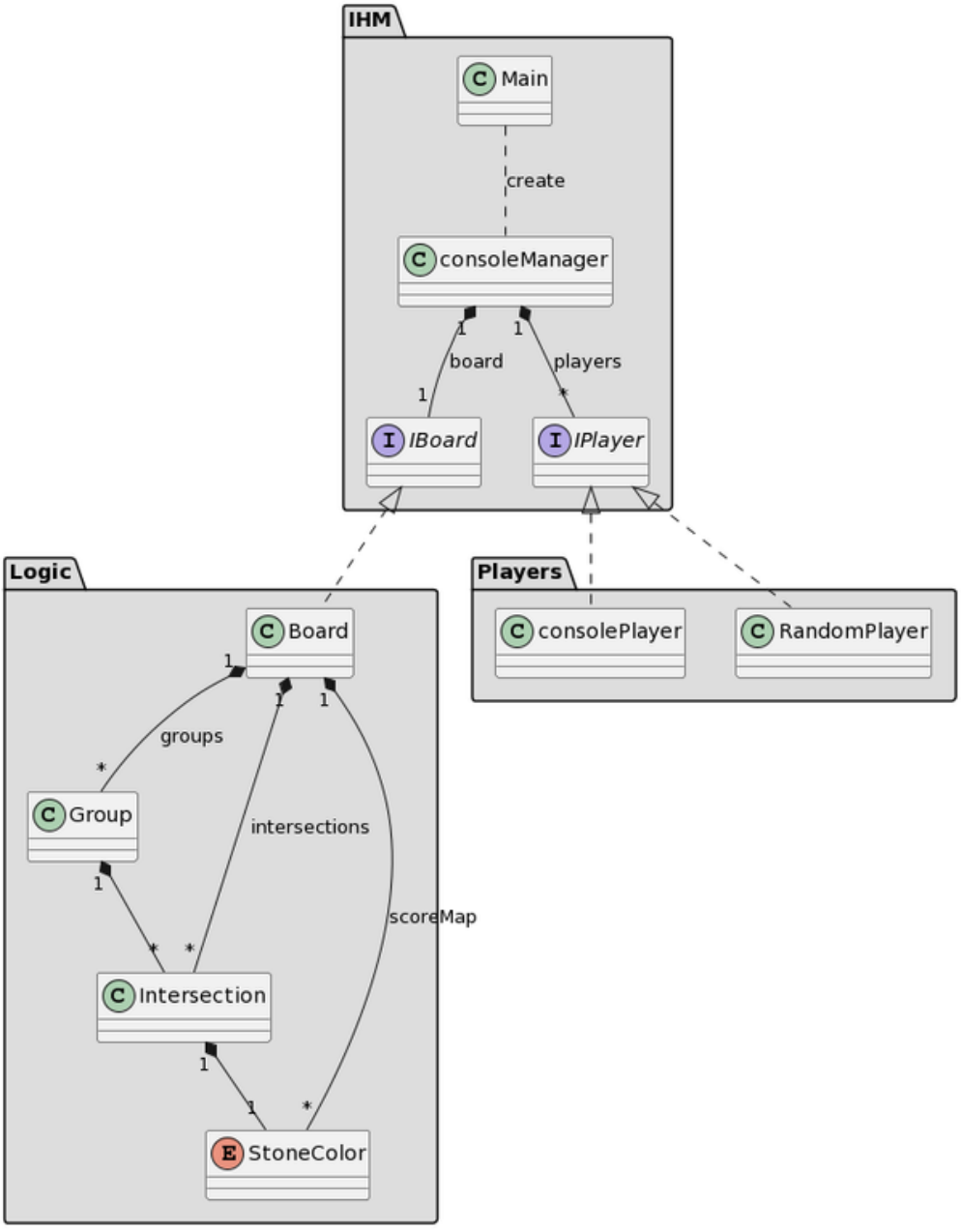
Francesca VOICU, Yasmine BENALI, Alicya-Pearl MARRAS | 205

SOMMAIRE

INTRODUCTION

Les sprints 1 et 2 ont été réussis, les commandes *play*, *clear_board*, *showboard*, *boardsize* et *quit* fonctionnent, la gestion des erreurs et de l'identifiant de commande compris.
Concernant le sprint final, le type de joueur a été implémenté et l'IA fonctionne. Le joueur noir commence toujours, puis alterne avec le blanc. Si un joueur est de type console, il peut passer. Lorsque le goban est plein ou que deux joueurs console ont passé, le jeu se termine et le score final est affiché.
Nous avons décidé de ne pas implémenté le suicide et le ko. Notons qu'une pierre peut se suicider mais est capturé par le joueur adverse s'il occupe toutes ses libertés.

DIAGRAMME D'ARCHITECTURE



LISTE SYNTHÉTIQUE

FONCTIONNALITÉS TESTÉES :

- Partie entre deux IA
- Partie entre une IA et un joueur console
- Gestion des captures (sans suicide et sans ko) : pour des groupes ainsi que pour des pierres seules
- Commande liberties
- Commande player
- Commande pass (disponible que pour les consolePlayers)
- Commandes de base (boardsize, showboard, quit)
- Fin de partie lorsque deux joueurs console passent leur tour successivement
- Fin de partie quand le goban est plein

BILAN

Lors de la réalisation de ce projet, nous avons réussi à avancer tout en étant confronté à quelques difficultés. En effet, il a été fastidieux de trouver comment gérer la capture des pierres et surtout des groupes, et quelles classes étaient nécessaires à cela. Le besoin de respecter l'encapsulation et les principes SOLID a aussi compliqué notre avancée. Par ailleurs, nous avons trouvé complexe la tâche de refactoring sur un code long pour rembourser notre dette technique. Nous nous sommes rendu compte de l'importance de la garder la plus petite possible tout au long du projet.

Nous avons tout de même réussi à développer un jeu de Go ayant un système de capture fonctionnel ainsi qu'une IA. Nous pensons que l'implémentation des classes *Intersection* et *Groupe* ont nettement facilité la création et la compréhension de l'algorithme du jeu.

Enfin, nous sommes conscientes que notre architecture est encore améliorable, surtout dans le package *logic*. Il nous semble notamment judicieux de créer une classe de gestion des captures *captureManager*, servant à alléger la classe Board, et de factoriser le code permettant la gestion des coordonnées du plateau. Aussi, il serait possible de réduire les dépendances à l'énumération StoneColor. L'implémentation du non-suicide elle permettrait de trouver une solution au soucis de pierres seules capturées dès leurs suicides.