Alex MAINGUY
Alexandre JOUSSET

Département Informatique
1er année – Groupe D
Année 2018 - 2019

RENDU FINAL DU PROJET

Projet de programmation : Quoridor

Destinataire

M. Sébastien LEFEVRE

Table des matières

1) Présentation de l'organisation du projet

Durant la première période à 4, nous avons un peu tarder à commencer le cahier des charges mais finalement, nous avons réussi à bien finir dans les temps. Ensuite, lors de l'élaboration du cahier des spécifications, nous avons essayé de faire un peu de projet chaque semaine, mais il est arrivé que certaines semaines nous ne continuons pas le projet. Mais nous avons quand même réussi à le finir à temps. On a aussi essayé de trouver du temps pour travailler ensemble. Ensuite, durant la deuxième période à 2, nous avons su bien se partager les tâches pour pouvoir finir l'application dès le vendredi.

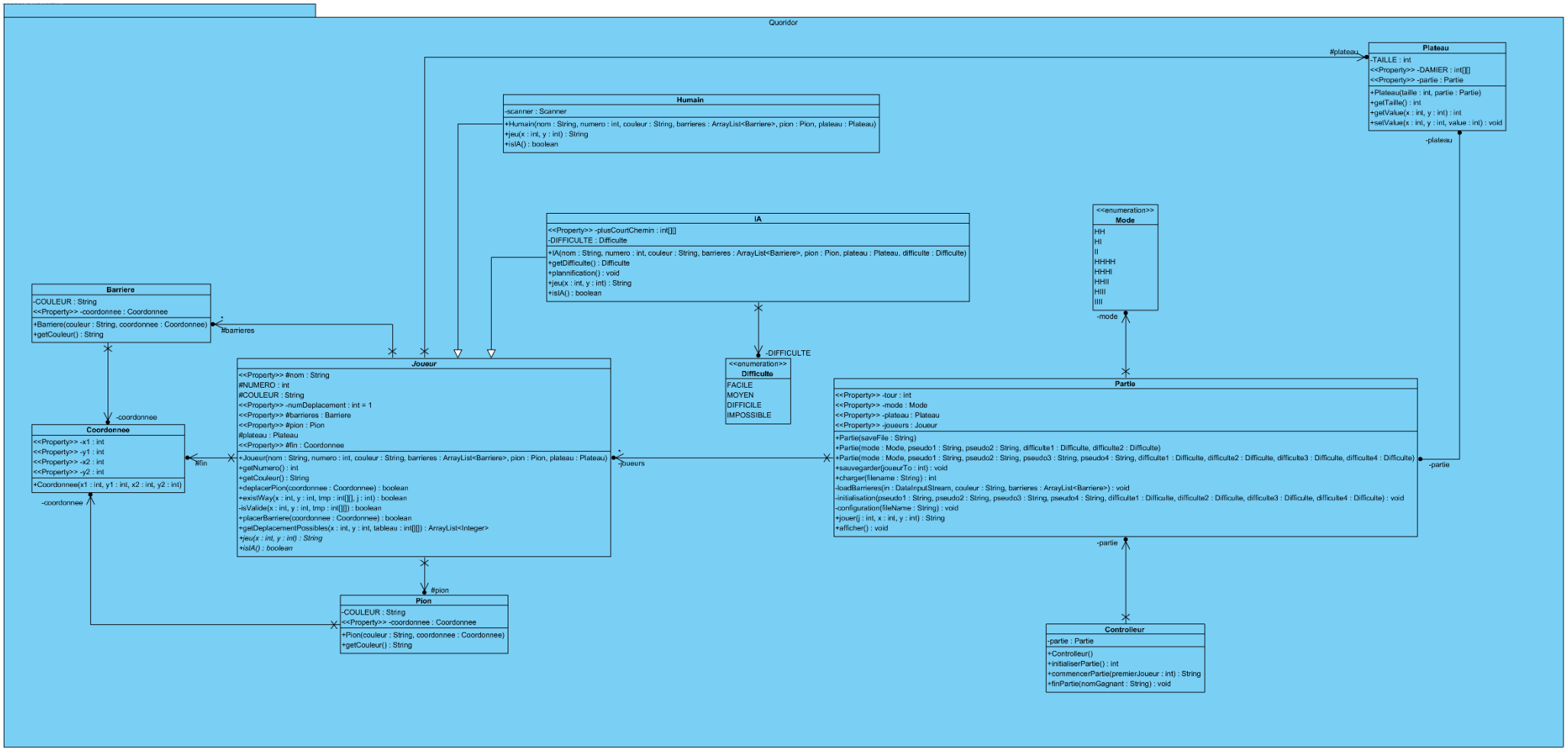
Comme dit précédemment, nous avons essayé de tenir le planning prévu mais il est arrivé qu'on décale le planning prévu et donc faire plus de choses en moins de temps. Mais durant la période de codage, nous n'avons pas eu de problème de planning car on avançait bien chaque jour sur le projet.

Utilité de planifier et d'organiser un projet : 8/10 -> Utiliser un planning est très important car il permet de voir quelle sont les dates de rendus des livrables, les temps que l'on a pour le faire, les temps qui peuvent être utilisé pour travailler et les temps où on ne peut pas travailler. Mais tenir réellement un planning et respecter bien tous les délais n'est pas possible, il y a toujours des contraintes qui font que le planning ne peut pas être tenu.

Les recommandations que nous pouvons faire aux prochains étudiants de première année est de bien savoir quand sont les dates des rendus et ne pas se dire qu'une chose sera rapide à faire et donc le faire plus tard car quand on ne l'a pas commencé, on ne se rend pas compte de la charge de travail.

2) Modification des diagrammes de classes par rapport au cahier d'analyse et de conception

Diagramme de classe des classes principale de Quoridor (Model) :



Le diagramme de classe a été modifié par rapport à celui du cahier d'analyse et de conception. On a ajouté des méthodes pour pouvoir clarifier le code de certaines méthodes de base. On a aussi modifier, enlever et ajouter des méthodes car soit ils en manquaient ou certaines n'étaient pas utile.

3) Description des choix techniques et algorithmiques

Au début du jeu, on demande au joueur s'il veut utiliser un fichier de sauvegarde ou faire une nouvelle partie. Ensuite, l'utilisateur choisit le nombre de joueur et le nombre d'IA. Une fois cela fait, on crée un nouvel objet Partie avec les choix de l'utilisateur. Cela va créer le Plateau avec les pions des joueurs dedans (numéro du joueur sur le tableau à deux dimensions d'entier), les Joueur, les listes de Barriere des Joueurs et enfin les Pions des Joueurs. Pour mémoriser la position des Pions et des Barriere (même s'ils ne sont pas placés), on utilise la classe Coordonnee qui détermine des coordonnées.

Ensuite, quel que soit le mode de jeu utilisé (en mode terminal ou en mode graphique), on attend que le joueur joue (sauf si c'est une IA, le jeu commence directement avec 1 seconde d'attente avant qu'il joue, tous les joueur IA fonctionne comme cela). En mode terminal, on demande au joueur s'il veut déplacer son pion ou placer une barrière, ou sauvegarder et quitter, et on récupère les coordonnées x et y rentré par le joueur. En mode graphique, on attend que le joueur clique sur le plateau (soit pour déplacer son pion ou soit pour placer une barrière) et on récupère aussi les coordonnées x et y, et il peut aussi sauvegarder et quitter en allant dans le menu pause.

Puis, le programme lance le tour d'un joueur (la méthode jouer() de Partie) qui elle-même lance l'action du joueur (méthode joue() de Humain). Si le joueur veut déplacer son pion, cette méthode va lancer deplacerPion() et va vérifier si le joueur peut se déplacer à cet emplacement en utilisant le méthode getDeplacementPossible() qui renvoie tous les déplacement possible. Sinon, s'il veut placer une barrière, la méthode placerBarriere() sera lancé et va utiliser un algorithme récursif (le backtracking). Il va vérifier si aucun joueur ne sera bloqué si la barrière est placée. Pour cela, il va essayer de trouver un chemin vers sa destination de victoire. Il va retenir chaque emplacement par où il est passé et s'il se retouve bloqué, il va retourner en arrière pour voir s'il y a un autre chemin. Si la méthode de déplacement de pion ou la méthode de placement de barrière trouve que ce qu'a donné le joueur est possible, alors les coordonnées du pion est changé et le numéro du joueur sur le plateau est déplacer ou sinon les coordonnées d'une barrière du joueur est modifié (s'il lui reste des barrière non placé) et on ajoute le numéro qui signifie que c'est une barrière sur le plateau (numéro 5). S'il y a une erreur dans les coordonnées x et y donné par le joueur, alors la méthode renvoie un message d'erreur à la méthode précédente, sinon elle renvoie une chaine vide. Et ensuite, la méthode jouer de Partie récupère le message, et s'il n'y a pas de message d'erreur, elle vérifie si le joueur qui vient de jouer a gagné. Si c'est le cas, elle renvoie gagner, sinon une chaine vide ou le message d'erreur. S'il y a un message d'erreur, il est affiché et le joueur doit redonner des coordonnées, sinon le plateau est réactualisé et le prochain joueur doit jouer. Et si c'est gagné, un message montre quel joueur a gagné.

Comme expliqué auparavant, si le joueur est une IA, cela est un peu différent. La méthode jouer de Partie est lancé et lance la méthode jeu de IA. Pour le moment, seul le mode facile de l'IA est implémenté, donc il y a des méthodes dans IA qui sont utile pour les autres niveaux de l'IA mais qui n'ont pas eu le temps d'être implémenté. L'IA joue donc aléatoirement en mode facile. Tout d'abord, il y a un choix aléatoire de soit déplacer le pion ou soit de placer une barrière. Ensuite, les coordonnées sont aussi déterminé aléatoirement en utilisant les méthodes deplacerPion() et placerBarriere(), et tant que le placement est impossible, il recalcule des coordonnées.

Et donc à la fin, quand la partie est finie, le jeu s'arrête en mode terminal et renvoie au menu principal pour le mode graphique.

4) Description de la campagne de tests effectués

Lors du développement du jeu nous avons pu effectuer quelques tests afin de vérifier le bon fonctionnement des méthodes principales de notre programme, tel que les constructeurs, les différents Getters et Setters. Pour faire effectuer ces différents tests nous avons utilisé l'outil JUnit que nous avons ajouté à ANT ce qui nous a donc permis donc de générer le jeu mais en même temps d'effectuer tous les tests.

5) Etat d'avancement du développement

Nous avons réussi à implémenter tout ce qui était prévu dans le cahier des charges. Tout fonctionne correctement donc nous avons fini le développement de l'application dans les temps. Les seuls points que nous n'avons pas eu le temps de faire sont les fonctions secondaires qui auraient pu être implémenté si on avait eu le temps. Ce sont par exemple le mode pédagogique, le mode tournoi, le mode 2 contre 2, et l'implémentation des autres modes de difficultés de l'IA (moyen, difficile).

6) Synthèse des difficultés rencontrées et des solutions apportées

Lors du développement du jeu nous n'avons pas rencontré de nombreuses difficultés mais le peu que nous avons rencontré mettaient le fonctionnement du jeu final en péri car notre méthode qui génère les déplacements possibles nous autorisais d'effectuer des mouvements qui ne sont pas autorisés dans le jeu. Les solutions que nous avons apportées ont été de revoir cette méthode bout de code par bout de code afin de connaitre la cause et de pouvoir la corriger ce que nous avons réussi à faire.

7) Bilan personnel du travail réalisé

Le bilan est plutôt positif nous avons tous les deux acquis des connaissances supplémentaires en Java, nous avons pu faire un travail d'équipe assez intéressent avec Git afin de pouvoir travailler a plusieurs en même temps tout en pouvant utiliser le code de l'autre assez simplement.

Nous pensons avoir à peu près atteint ce que nous pensions réaliser malgré le fait que nous aurions pu un peu plus développer la partie IHM du jeu avec l'ajout de composant graphique plus jolie ou bien ajouter des animations lors des différentes actions…