UNIVERSITÉ DE MONTPELLIER

FACULTÉ DES SCIENCES

Informatique

RAPPORT DE PROJET

T.E.R. HLIN405

Quarto

Proposé par :

CAPRIO LOMBARDI Thomas & DUCHON Damien & GOLDSTEIN Solal & GUILLEN Victor & QUINTANE Alexis

Année : 2019

Encadrant :

Vincent Boudet

Nous remercions notre encadrant M Vincent Boudet pour son accompagnement, ses conseils, sa bonne humeur et pour s'être rendu disponible tout au long de ce semestre pour réaliser ce projet et ce rapport.

Table des matières

Introduction

Histoire du jeu

"Le Quarto est un jeu de société combinatoire abstrait au tour par tour, créé par Blaise Muller, [...] et édité depuis 1991 par Gigamic. " - Wikipedia

Présentation du Quarto

Le Quarto est composé d'un plateau de 16 cases (4 par 4) et de 16 pièces. Les pièces sont toutes différentes, suivant 4 caractéristiques : la couleur (claire ou foncée), la hauteur (haute ou basse), le sommet (pleine ou creuse) et la forme (ronde ou carrée).

Les règles sont simples : la partie se joue tour par tour. Un tour est divisé en deux tâches : il faut d'abord jouer (donc placer) le pion sur une case libre. Ce pion est imposé par l'adversaire. Ensuite il faut donc choisir la pièce que jouera l'adversaire. Le gagnant est celui qui, avec une pièce reçue, crée un alignement de 4 pièces ayant au moins un caractère commun.

Règles officielles du jeu disponible au lien suivant :

`https://www.gigamic.com/files/catalog/products/rules/quarto_rule-fr.pdf`

Elements de motivation

Quelles ont été nos motivations pour avoir choisi ce sujet ? Il nous a paru intéressant de le traiter puisque avant tout c'est un jeu, avec des règles, des pions, un gagnant et un perdant. C'est ce qui nous plaît ! Alors nous avons laissé nos âmes de joueurs (multiplateformes) parler et nous nous sommes lancés.

Développer un jeu permet d'explorer plusieurs horizons. Nous allons devoir solliciter notre créativité pour travailler sur l'aspect visuel du jeu, une jolie interface intuitive et simple d'utilisation. Partir à la découverte des algorithmes de jeu, plus ou moins complexes, et en produire une intelligence capable de jouer contre une personne. Nous voulions également découvrir un nouveau langage de programmation (de haut niveau), et bien d'autres... Ce sujet nous parait idéal. C'est une aventure que chacun d'entre nous voulait vivre.

Nous pensons que ce projet permet d'expérimenter plusieurs facettes de développement d'un programme collaboratif. Celui-ci satisfaisant les intérêts de chaque membre du groupe. C'est une expérience plus que positive en vu de nos projets futurs, nos ambitions, nos choix d'études.

Toute occasion de mise en pratique des notions que l'on étudie dans le cadre de nos études est une source de motivation et suscite d'avantage l'intérêt que nous portons pour l'informatique, et la programmation en particulier.

Chapitre 1

Organisation du projet

1.1 Mise en place d'un cahier des charges

> "Le but du projet est d'implémenter un jeu de quarto. Le programme devra proposer les fonctionnalités suivantes : tout d'abord, le programme devra permettre à deux joueurs humains de s'affronter. Enfin, le programme devra être capable de sauver et de charger une partie en cours. Ensuite, le programme devra permettre si c'est possible de jouer contre l'ordinateur et de proposer des conseils." - Sujet de Vincent Boudet

Avant de nous lancer dans le développement du jeu Quarto, nous nous sommes réunis dans le but de se mettre d'accord sur les aspects du programme que nous voulions approfondir le plus. M.Boudet nous laissant libre sur le niveau des ambitions pour réaliser ce projet.

Nous avons alors convenu que, par soucis d'intérêt de l'ensemble du groupe, nous élaborerions une intelligence artificielle. L'objectif commun étant d'approfondir et de mettre en pratique des notions algorithmiques plus complexes.

Le design du jeu se présentera sous forme d'un plateau et de pièces en 2D, vu du dessus, en inspiration de jeux de plateaux déjà existants comme les échecs ou le jeu de dames. De plus nous voudrions un affichage des pièces en 3D ou en perspectives afin que l'utilisateur puisse mieux se les représenter.

Nous pensions également à une "boite à messages" pour rendre compte de l'avancée du tour ou de la partie, par exemple.

Évidemment pour pouvoir plus aisément naviguer dans le programme, des boutons comme "retour au menu principal" ou "relancer la partie" pourraient être implémentés.

1.2 Méthode de travail

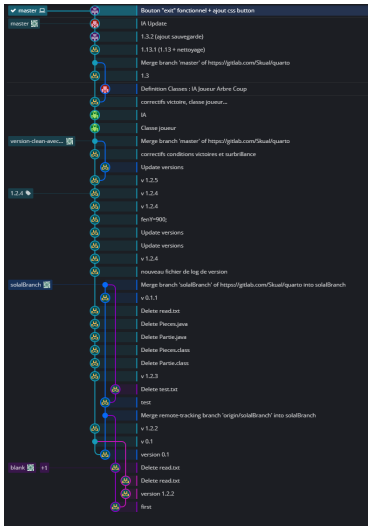
Nous nous sommes retrouvés quotidiennement pour rendre compte des avancées de chacun. L'avancement du projet s'est fait dans l'ordre de priorité défini au départ. Nous nous sommes également réunis au complet à plusieurs reprises pour effectuer un bilan de mi-parcours, vérifier si les deadlines étaient respectées et discuter de la suite du projet.

Nous avons travaillé soit chacun de notre côté sur des tâches simples, soit en groupe lorsque les tâches étaient un peu plus complexes. Nous nous retrouvions donc sur le campus de la faculté ou en appel vocal sur nos ordinateurs respectifs pour travailler ensemble.

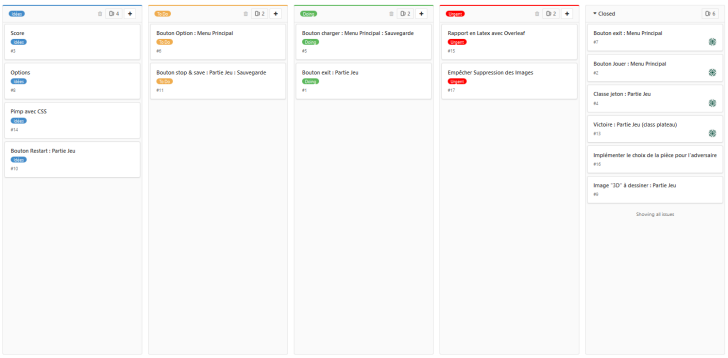
Environ toutes les deux semaines, parfois plus, nous nous sommes réunis avec notre encadrant M. Boudet afin de faire le point sur l'état d'avancement du projet. Ces réunions nous ont permis de discuter des difficultés rencontrées au fur et à mesure avec notre encadrant. M. Boudet nous a aussi donné des pistes de documentation et des conseils pour la réalisation du projet, par exemple pour les algorithmes d'intelligence artificielle.

1.3 Outils de travail

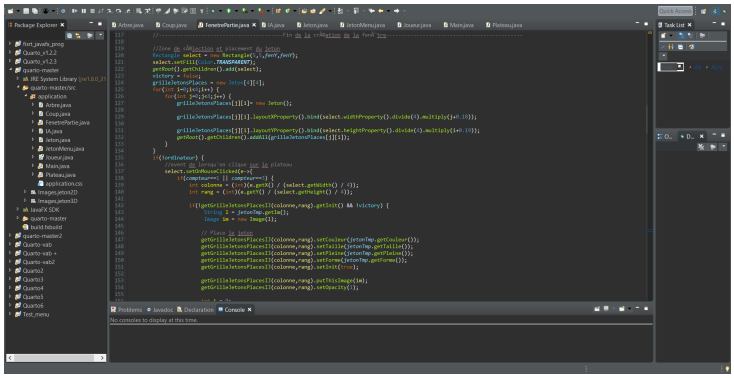
Nous avons choisi d'utiliser Git au travers du serveur GitLab. Il permet la gestion des versions du projet. Git est très pratique et facile d'utilisation. Il facilite le travail en collaboration sur un projet informatique tel que celui-ci.



Pour gérer l'avancement du projet et que chacun puisse en tenir compte, nous avons rangé par problèmes ou "issues" dans un tableau. Un service proposé par Gitlab, un extrait ci-dessous :



Nous avons choisi le langage de programmation Java, qui est un langage de programmation orienté objet. Nous avons tous utilisé l'IDE Eclipse afin d'utiliser aisément la bibliothèque graphique javaFX. Leur utilisation est détaillée plus tard.



Nous avons utilisé le logiciel Cinema4D afin de produire une visualisation 3D des jetons. Par exemple :

Chapitre 2

Conception et implémentation du Quarto

2.1 Langage de programmation et librairie utilisées

2.1.1 Pourquoi avoir choisi Java ?

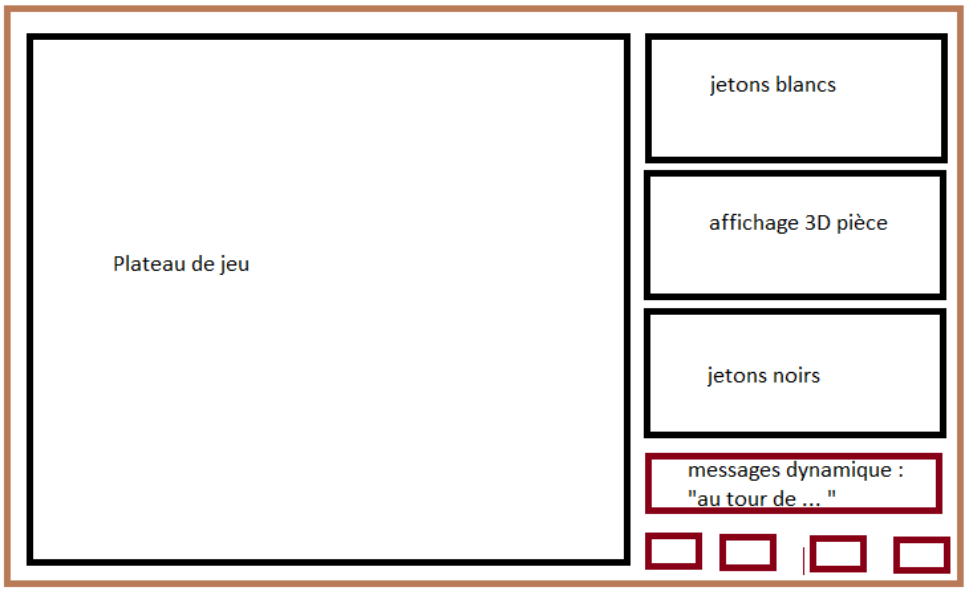
Après avoir mis en place le cahier des charges ainsi que les différents outils nécessaires à la réalisation du projet, nous devions choisir un langage de programmation. Dans un premier temps, nous avons longuement hésité entre les deux langages C++ et Java. Nous avions déjà tous abordé le C++ à travers différentes U.E, notamment cette année avec le module HLIN302. C'est pourquoi, le Java nous étant inconnu, et, ayant une U.E de programmation orientée objet en Java (HLIN406) ce semestre, nous avons pensé qu'il était intéressant de pouvoir mettre rapidement ce module en application.
Le Java étant un langage de programmation orienté objet, il serait parfaitement adapté à la modélisation d'un jeu comme le Quarto. En effet il s'agira ici de beaucoup de manipulation de jetons, images, etc..

2.1.2 La bibliothèque JavaFX

Dès lors que le langage a été choisie, nous avons cherché une bibliothèque graphique compatible à notre langage, mais surtout, à notre cahier des charges. Ici nous hésitions entre Slick2D et JavaFX. Pour faire un choix, nous avons fais une ébauche de plateau de jeu avec les deux bibliothèques. Nous avons tenté du cliqué-glissé, et un simple clique (sélection) et clique (positionnement). Finalement nous avons retenu la bibliothèque JavaFX car elle semblait plus adaptée à notre projet. En effet, les méthodes de gestion de formes et d'images y sont très bien fournies.

2.2 L'interface

Toute cette partie a été traitée sur un tableau blanc. Nous avons fait de nombreux schémas afin d'en trouver un qui soit le plus ergonomique possible tout en gardant un aspect visuel agréable. Au début nous voulions mettre le plateau de jeu au dessus de tous les jetons et boutons, mais de cette façon, il restait beaucoup de "blancs" sur l'interface. Nous voulions palier à ce problème et pour cela nous avons décidé de mettre le plateau de jeu à droite comme ci dessous.



2.2.1 L'aspect graphique

Le Quarto étant un jeu intuitif, nous voulions garder une simplicité au travers de l'aspect visuel. C'est pourquoi nous avons choisi de représenter les jetons en 2D (modélisation via Photoshop) et un plateau simple où les contrastes de couleur permettent d'identifier clairement chaque jeton. Cette représentation étant simpliste, afin d'éviter tout problème d'ambiguïté nous avons décider d'implémenter des jetons en 3D (modélisation via Cinema 4D).
De plus, la bibliothèque graphique JavaFX permet l'implémentation de code CSS. Ce qui nous a été très utile quant à la personnalisation des boutons par exemple.

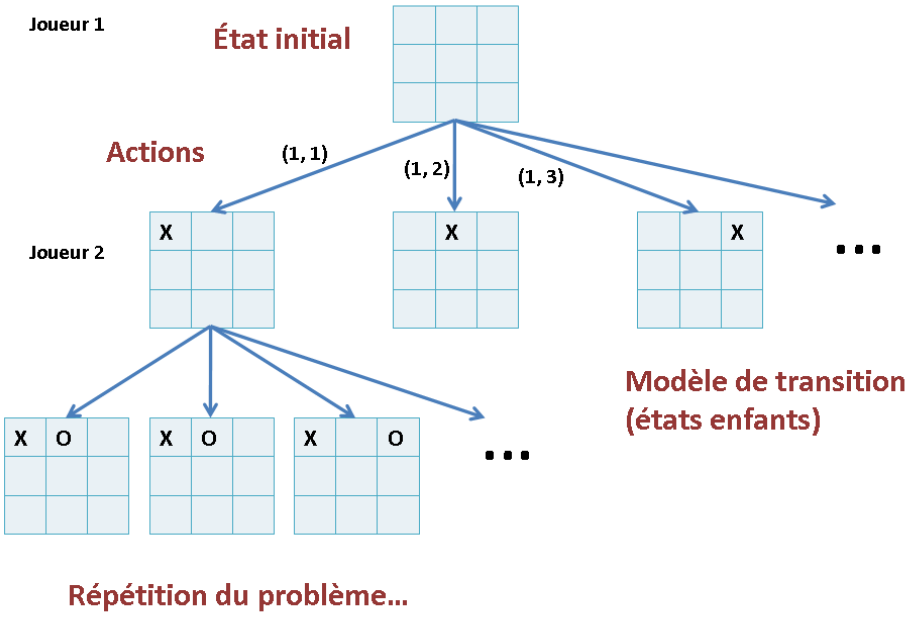
2.2.2 Gestion de la partie

La classe jeton permet de modéliser chaque jeton par ses 4 caractéristiques. La sélection se fait par le biais de "radiobutton". Lors de sa sélection, toutes les caractéristiques du jeton en question vont être stockées dans un jeton tampon et transmises au réel jeton, lorsqu'il sera placé sur le plateau via un simple clic (grâce aux Events). A chaque placement de jeton, une fonction teste toutes les conditions de victoire. Si il y a une victoire, la ligne gagnante apparaît en évidence, le joueur gagnant est annoncé et la partie est terminée (impossible de placer de nouveaux jetons) ; sinon la partie continue normalement. Toutefois, il est possible de sauvegarder une partie grâce au bouton "stop & save" et de la reprendre à tout moment. En effet, lorsqu'on clique sur ce bouton, un fichier .txt est généré avec toutes les informations nécessaires au chargement de la partie en cours ; le tour en cours et le contenu de chacune des cases du plateau.

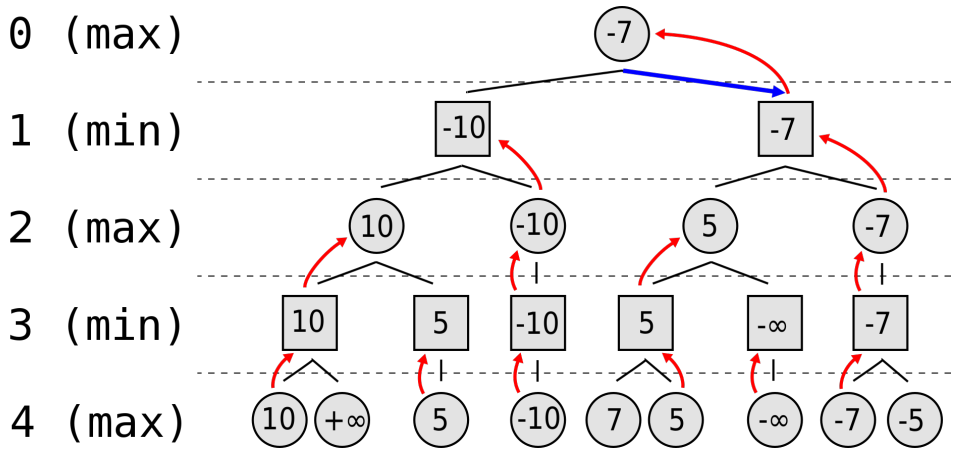
2.3 L'intelligence artificielle

2.3.1 L'Algorithme MinMax

Afin de pouvoir jouer contre "l'Ordinateur", on utilisera l'algorithme MinMax, appelé aussi Minimax. Celui-ci est adapté aux jeux qui se jouent à deux joueurs, l'un après l'autre. On l'utilise notamment pour le jeu de Go, l'Othello, ...
Le but principal de cet algorithme est de renvoyer le meilleur coup à un instant précis. Pour cela, l'algorithme va simuler toutes les issues possibles, à partir de la situation actuelle jusqu'à plus ou moins loin dans le "futur". Une fois les prévisions faites, celui-ci doit faire remonter le coup menant à l'issue la plus intéressante pour l'ordinateur.
Pour cela, l'algorithme utilise un arbre. La racine représente le jeu dans son état actuel, et pour ses fils tout les coups possibles, et ainsi de suite... Exemple simple avec le jeu du morpion :



Évidemment, plus on "regarde loin", plus il y a d'issues possibles, cela dépend du jeu. Une fois l'arbre construit, on donne une valeur à ses feuilles, valeur plus ou moins grande selon si l'issue est intéressante pour l'IA ou pas. Cette dernière va chercher l'issue avec la valeur la plus grande (ou la plus petite selon comment vous programmez l'algorithme) lorsque c'est son tour. Lorsque c'est le tour de son adversaire, elle va supposer que celui-ci choisira le coup le moins intéressant pour elle, d'où cette alternance Min/Max qui donne son nom à l'algorithme.



Ainsi, l'algorithme va faire remonter l'issue la plus intéressante, en prenant compte des choix possibles de l'adversaire.

2.3.2 Utiliser MinMax avec le Quarto

Pour notre Quarto, l'algorithme fonctionnera de la même manière. Chaque fils représentera un coup, c'est-à-dire, les coordonnées de la pièce qui nous a été attribué et de la pièce qu'on aura choisi pour l'adversaire.
Le problème viendra du nombre de coup possible. Par exemple, au premier tour on aura, 16 coups possibles, qui représente la pièce qu'on choisit de jouer pour commencer. Au deuxième tour, on doit alors poser cette pièce quelque part sur le plateau et choisir celle que posera l'adversaire. On a donc 16 positions et 15 pièces possibles, fois les 16 premiers coups, on a donc sur les premiers tours : 16 * 16 * 15 = 3840 issues juste pour les deux premiers tours ! Puis 806400 si on continue un tour de plus...
C'est pour cela qu'on ne fera pas fonctionner l'algorithme sur les premiers tours. Il ne faudra pas abuser non plus sur la hauteur de l'arbre afin de ne pas surcharger la mémoire.

2.3.3 Réalisation en Java

Pour pouvoir faire fonctionner l'algorithme MimMax, on a besoin de trois classes :
Une classe Coup, qui contient les coordonnées de la pièce donné par l'adversaire et la nouvelle pièce. On utilise des "bytes" plutôt que des "int" pour justement diminuer la quantité de mémoire utilisée, étant donné le nombre de coups possibles...
Une classe Arbre, avec une racine et ses fils. La racine contient le coup du noeud et ainsi que sa valeur.
La classe IA est contient le tout, avec notamment l'algorithme MinMax, l'arbre et la méthode jouerTour, qui gère le tout : l'initialisation du l'arbre après le 3ième tour, etc.

Chapitre 3

Bilan et difficultés rencontrées

3.1 Avancement du projet

Dans le chapitre précédent, nous avons vu la conception et l'implémentation du projet. Cette prochaine partie sera consacrée à l'avancement du projet. Bien que le Quarto soit un jeu relativement simple, il est clair que nous avions sous-estimé le travail à fournir pour ce projet. Dès lors que certains outils furent installés correctement, que le cahier des charges fût validé et que les premiers fragments de codes furent fonctionnels, le projet a évolué à une vitesse linéaire. Nous avons essayé de répartir de la meilleure manière possible les différentes tâches à effectuer. En effet, nous avons réparti certaines tâches en fonction des compétences de chacun. Il est non négligeable que fournir un travail constant et régulier a permis la bonne évolution du projet.

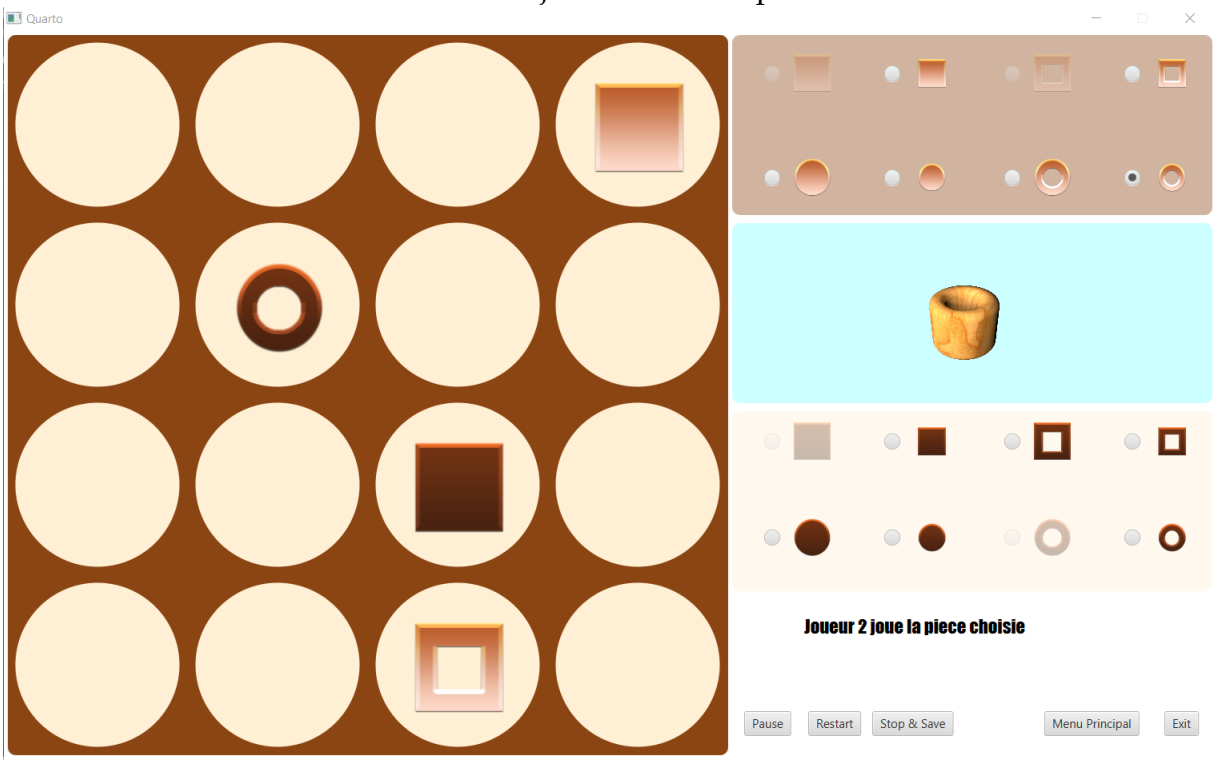
3.2 Difficultés rencontrées

Malgré l'utilisation d'outils de travail à priori adaptés au bon déroulement de ce projet, nous avons rencontré quelques difficultés que nous verrons ensemble dans cette partie . Les différent outils utilisés tels que GitLab ou bien encore Eclipse, nous ont été d'une grande utilité quant à l'avancement du projet. Cependant, la diversité des ordinateurs de chaque personne du groupe et les différentes versions de chacun de ces outils étant parfois incompatibles entre elles. Certains langages possèdent des normes d'écriture pour palier à un souci de lisibilité et de compréhension. Fort heureusement pour nous, nous nous sommes rendus compte du non respect de certaines de ces normes avant que le projet ne soit trop avancé.

3.3 Connaissances et apprentissages

Dans la partie suivante, nous verrons les différentes connaissances que nous avons pu mettre en pratique lors de ce projet. La plupart de ces connaissances nous viennent des différents enseignements suivis lors du parcours universitaire et grâce à ce projet, nous en avons consolidé certaines et acquises d'autres. Le module HLIN202 qui s'étend de la première année à la deuxième avec le module HLIN302 nous a permis d'aborder la programmation orientée objet et ces principes fondamentaux. Mais, le module de programmation orientée objet en Java de cette année (HLIN406) nous a permis de mieux apprendre à modéliser, notamment à l'aide de diagramme UML, et mieux résoudre certains problèmes. De plus, le cours de Technique de Communication et Conduite de Projet (HLIN408) a permis à certains de découvrir des outils tels que Gitlab ou bien encore Latex et d'approfondir leurs connaissances quant à leur utilisation. Ce projet nous a aussi permis de développer de nouvelles compétences et connaissances autour de la conduite de projet. A l'aide de ces nouvelles connaissances ainsi que des recherches personnelles, nous avons donc modélisé le jeu de la manière suivante.

Interface du jeu durant une partie :



Conclusion

Dans un premier temps, nous avons vu l'organisation du projet, la conception du jeu, puis nous avons fait un premier bilan, nous pouvons maintenant conclure ce projet. Dans notre premier chapitre, nous avons expliqué la mise en place du cahier des charges. Celui-ci ayant été défini en grande partie par nous même grâce aux libertés accordées par notre enseignant, nous avons pu respecter celui-ci dans les délais imposés. Ce projet nous a permis à tous de nous perfectionner sur un aspect rédactionnel mais aussi sur les conventions et normes présentes dans la réalisation d'un projet informatique. La conduite de projet mais également son développement nous à appris a mieux nous préparer et de nous faire réaliser le travail fourni derrière un simple logiciel ou application. De plus, le travail en groupe a donné à certains une première expérience en équipe mais également technique, avec l'utilisation de logiciels de partage de travaux.

Perspectives

Le cahier des charges ayant été rédigé, il arrive que les concepteurs aient des idées tardives. Nous consacrerons donc cette partie aux idées et améliorations qui auraient pu être apportées. Certaines des ces améliorations n'ont pas été développées pour nous permettre de respecter au mieux le cahier des charges et surtout d'insister sur l'intelligence artificielle du jeu. Une amélioration graphique aurait été envisageable en implémentant un menu de personnalisation ou de choix de différents thèmes. En plus de l'aspect graphique, il aurait été intéressant de développer l'aspect intuitif, en y implémentant un « Glissé/déposé » des jetons. Pour terminer, l'ajout de son lorsqu'un joueur effectue une action aurait dynamiser la partie. Une musique d'ambiance aurait été agréable.

Annexes

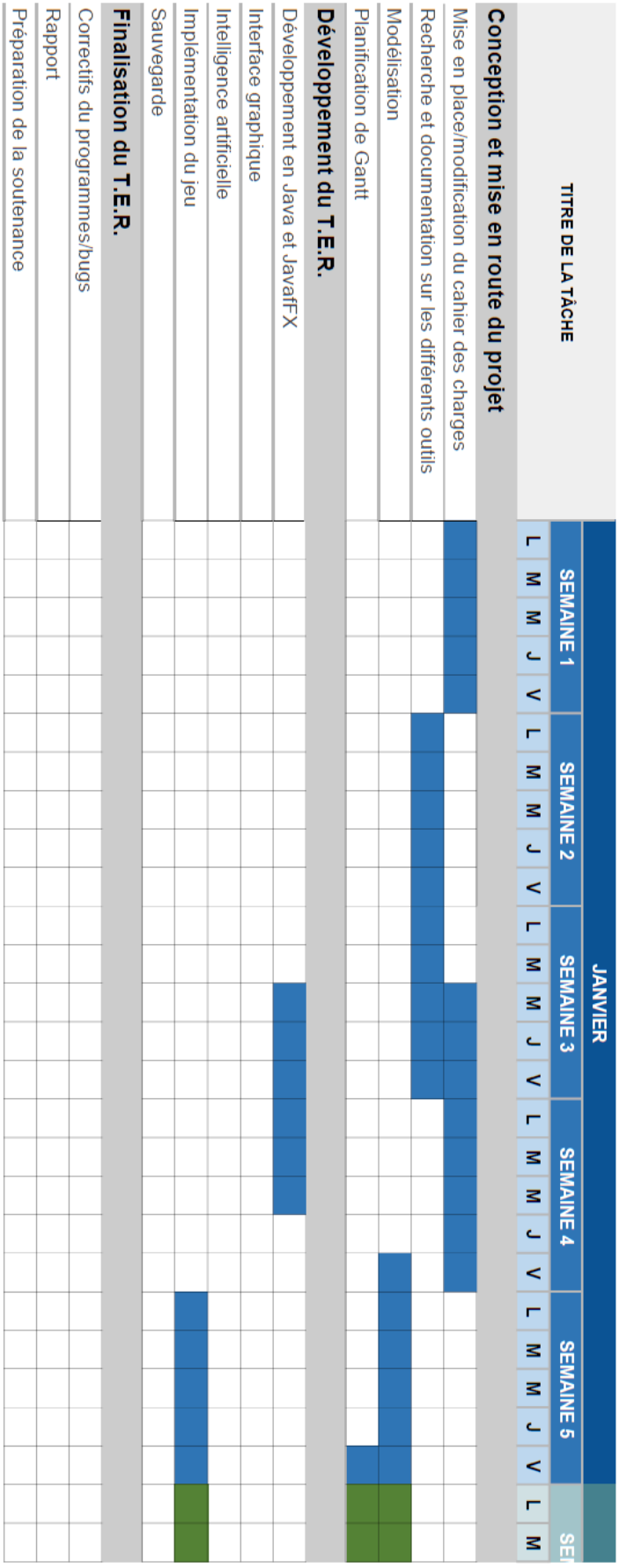
Algorithmes IA

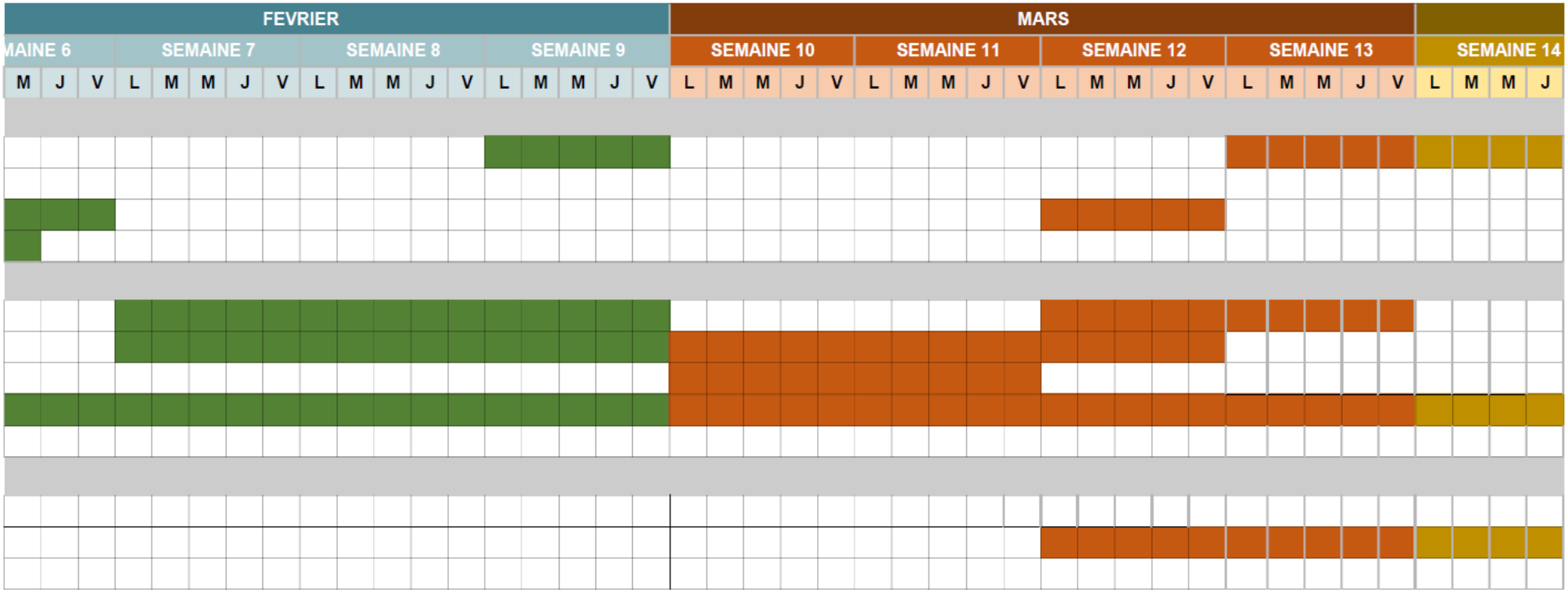
Extraits de code :

Algorithme MinMax :

```
private int algoMinMax (Arbre noeud, int level, boolean tourIA) {

    if (level == 0 || noeud.getFils().size() == 0) {
        return noeud.coupGagnant() ? 100 : 0;
    }


    int tmp;

    if (tourIA) {

        noeud.setVal(noeud.getFils().get(0).getVal()) ;

        for (Arbre fils : noeud.getFils()) {

            tmp = algoMinMax(fils, level-1, false);

            if (noeud.getVal()<tmp) {
                noeud.setVal(tmp);
            }

        }

    }

    else {

        noeud.setVal(noeud.getFils().get(0).getVal());

        for (Arbre fils : noeud.getFils()) {

            tmp = algoMinMax(fils, level-1, true);

            if (noeud.getVal()>tmp) {
                noeud.setVal(tmp);
            }

        }

    }

    return noeud.getVal();
}
```

Diagramme de Gantt




FEVRIER
MARS
AINE 6
SEMAINE 7
SEMAINE 8
SEMAINE 9
SEMAINE 10
SEMAINE 11
SEMAINE 12
SEMAINE 13
SEMAINE 14
M J V L M M J V L M M J V L M M J V L M M J V L M M J V L M M J V L M M J V L M M J V L M M J
M J V L M M J V L M M J V L M M J V L M M J V L M M J V L M M J V L M M J V L M M J V L M M J

AVRIL
MAI
SEMAINE 15
SEMAINE 16
SEMAINE 17
SEMAINE 18
SEMAINE 19
SEMAINE 20
V L M M J V L M M J V L M M J V L M M J V L M M J V L M M J V