Rapport de Projet : Minimalist Tower Defense (Type 2)

par Anthony FERNANDES et Thomas LU, groupe 176

Introduction :

Ce rapport présente un compte rendu du projet réalisé dans le cadre de l'UE Programmation orientée objet, dont le but était de réaliser un jeu de type tower defense. Nous avons choisi d'implémenter le tower defense de type 2, dans lequel les ennemis avancent sur un chemin prédéterminé et des tours sont placées autour du chemin pour les empêcher d'atteindre la fin du chemin.

Nous commencerons par détailler les différentes parties du cahier des charges ayant été traitées ainsi que les problèmes rencontrés lors du développement. Puis nous nous pencherons sur les parties non implémentées ainsi que d'autres pistes d'amélioration et ajouts de fonctionnalités possibles. Enfin nous ferons une brève présentation de la structure du projet.

Menu d'accueil de Minimalist Tower Defense

Parties du cahier des charges traitées :

- **Environnement graphique**

Le jeu démarre sur un menu principal, qui constitue le point d'entrée central. Depuis ce menu, le joueur peut choisir entre les deux modes de jeu proposés en accédant à leur menu de paramétrage, ou bien il peut consulter les scores de ses précédentes parties.

Dans le menu des niveaux, le joueur peut sélectionner un niveau parmi les 8 proposés. Au début seul le niveau 1 est débloqué, puis chaque victoire débloque le suivant.

Dans le menu du marathon, le joueur peut librement choisir l'une des 8 cartes ainsi qu'un niveau de difficulté pour y jouer à l'infini. Les résultats des parties précédentes sont ensuite accessibles depuis le menu score, où ils sont classés par difficulté.

Un menu Game Over intervient à la fin de la partie. Dans ce menu, le joueur peut consulter différentes statistiques concernant sa partie et on peut revenir dans le menu de création de partie pour rejouer.

- **Jeu fonctionnel** (encore heureux...)

Le jeu est composé d'une grille sur laquelle des ennemis apparaissent, (potentiellement à plusieurs endroits) et se déplacent le long d'un chemin vers une (ou plusieurs) bases à défendre.

Le haut de l'interface affiche au joueur les informations concernant le mode de jeu en cours, la vie qui lui reste, l'or à sa disposition et l'avancement de la partie. L'avancement est représenté par le nombre de vagues depuis le début ainsi que le nombre total de vagues en mode niveau.

Le joueur peut sélectionner en bas de l'écran un type de tour pour en acheter un exemplaire. Cette tour peut être placée sur l'un des emplacements prévu à cet effet sur la grille, permettant au joueur d'établir une stratégie de protection des bases en plaçant les tours à des endroits clés de la carte. L'achat de tour est effectué avec l'or que le joueur gagne en tuant des ennemis. Les tours suivent du canon leur cible et tirent des projectiles à travers la grille.

- **Diversité des tours**

Minimalist Tower Defense propose trois types de tours pour s'adapter aux ennemis et au chemin. La tour basique n'a, comme son nom l'indique, rien de très spécial (dégâts, vitesse de recharge et portée normaux). La tour canon est plus lente et a une portée plus courte que la tour basique mais elle inflige significativement plus de dégâts. La tour sniper se recharge légèrement plus lentement que la tour basique mais inflige les mêmes dégâts à plus longue distance.

- **Diversité des ennemis**

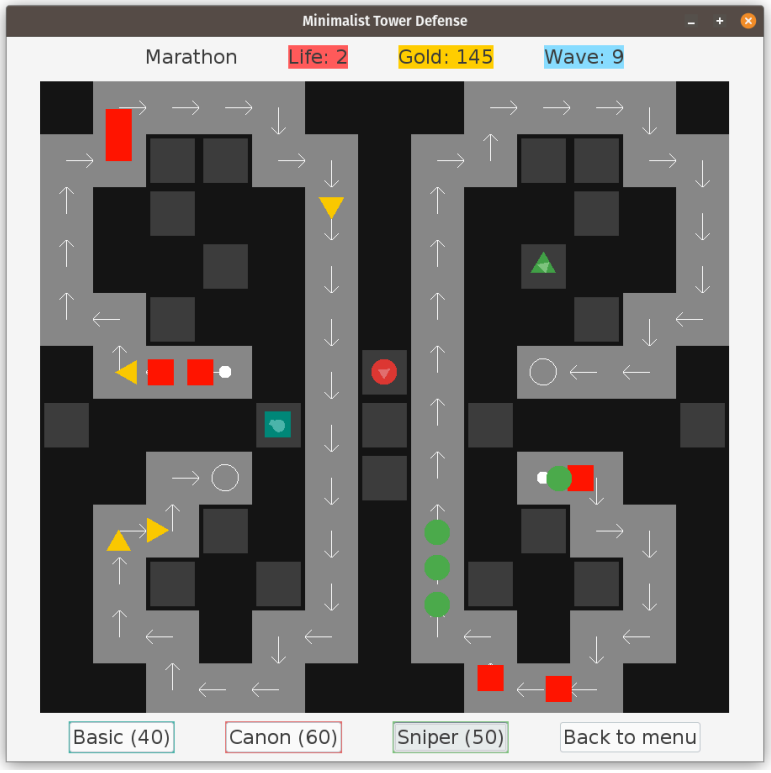
  Sont également disponibles, trois types d'ennemis, chacun avec des caractéristiques différentes. Les ennemis basiques, sans aucune particularités. Les ennemis rapides, qui ont moins de santé que les ennemis basiques mais sont plus rapides. Les ennemis forts, qui se déplacent bien plus lentement mais ont plus de vie, ils rapportent également plus d'or que les autres.

- **Distribuable en .jar** (hors cahiers des charges imposé par le cours)

  Le jeu est conçu depuis de début de telle façon qu'il est entièrement indépendant de l'architecture du système de fichier de l'ordinateur à l'extérieur du projet. Ainsi il peut être compressé en un fichier .jar facilement distribuable et exécutable par tout le monde.

Problèmes rencontrés :

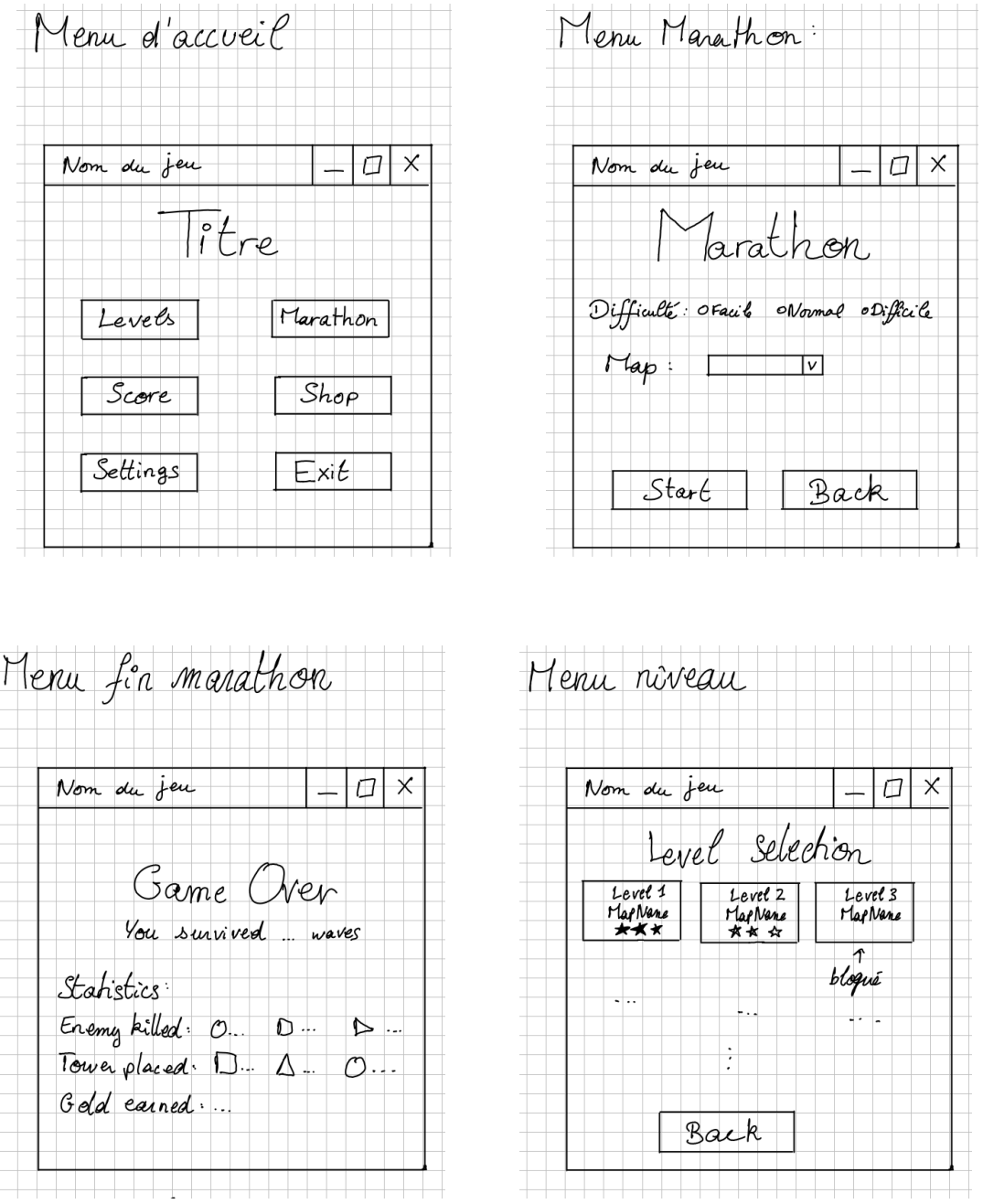
L'interface graphique est la partie nous ayant posé le plus de difficultés car l'idée était de faire en sorte que le jeu soit totalement redimensionnable. Pour ce faire, nous avons adopté des graphismes minimalistes, entièrement dessinés avec les outils de swing. C'est la raison pour laquelle tout est représenté par des formes géométriques comme, on peut le voir sur la capture d'écran ci-dessous.



Toutes les positions sont ainsi calculées à la volée au moment de l'affichage, en fonction de la taille de la fenêtre. De plus, la vitesse était initialement exprimée en pixels par seconde, nous avons dû changer cela en cellules par seconde grâce un système de

coordonnée interne, pour rendre le jeu portable. En effet, puisque la vitesse dépendait des pixels, deux ordinateurs munis d'écrans avec une résolution différente avaient des vitesses très différentes.

Pour garantir une interface lisible, même une fois redimensionnée, nous avons abondamment utilisé le `LayoutManager GridBagLayout` proposé par swing. Ce qui explique pourquoi presque toutes les interfaces (par celles des parties) ont été conçues comme des grilles dès le début.



La grille est moins visible sur le game over mais elle est bien là

Pistes d’extensions non implémentées :

Le sujet indique que pour le tower defense de type 2, il faut que le chemin puisse se croiser lui-même. Le système actuel de direction ne le permet pas car les seules directions autorisées sont `UP`, `DOWN`, `LEFT`, `RIGHT` et `END_OF_PATH`. Ce problème pourrait sûrement être réglé par l’ajout d’une direction `CROSSING`, puis en modifiant de la logique de changement de direction des ennemis pour ne pas les faire tourner quand ils rencontrent un croisement.

Les systèmes de sauvegarde et d'amélioration des tours suggérés par le sujet n’ont pas été implémentés par manque de temps. Une amélioration passive des tours avec un certain nombre d’ennemis tués serait l’option la plus facilement implémentable.

Il pourrait être intéressant d’ajouter une monnaie globale, conservée entre les parties, pour acheter des choses permanentes. Le problème ici est que l’achat de nouveaux types de tours ou de personnalisation de l’apparence du jeu serait complexe, étant donné que tout est dessiné avec les moyens de swing.

On pourrait aussi envisager une histoire au mode niveaux, la difficulté ici serait plutôt de trouver des éléments scénaristiques pertinents, l’implémentation de cette fonctionnalité en elle-même n’est probablement pas difficile avec Swing.

On peut aussi penser à des sons pour rendre le jeu plus immersif, le plus important serait d’ajouter des effets sonores, on pourrait ensuite ajouter des musiques de fonds pour les différents menus et différents modes de jeu.

Architecture du projet :

Le projet contient un total de 48 classes réparties en neuf packages. Il suit le patron de conception Model-View-Controller. Parmi les neuf packages, quatre sont à la racine du classpath : les trois classiques `model`, `view` et `controller`, ainsi qu'un package `util`. On trouve dans celui-ci les enums partagées par les trois autres packages (difficultés, directions et statuts), ainsi que le système de coordonnées et une interface fonctionnelle représentant les fonctions à trois arguments.

Les cartes prédéfinies ainsi que les index des cartes sont sauvegardées sous forme de fichier texte dans le répertoire `src/resources`.

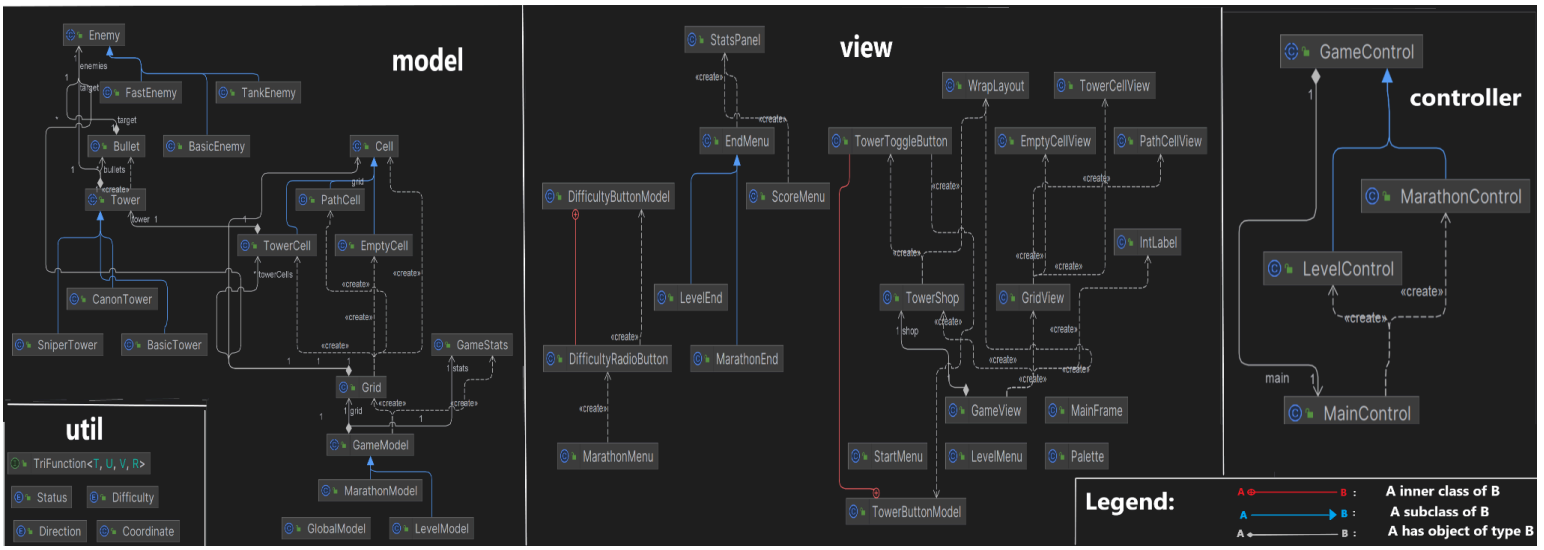


Diagramme de classe du projet (fichier en meilleure résolution dans le répertoire du projet)