1. Introduction

Nous allons présenter le contexte de notre projet. Nous allons également présenter un scénario dans lequel notre projet prendrait son sens. Les objectifs ainsi que l'organisation du rapport seront également abordés durant ce chapitre.

1.1 Contexte du projet

Ce projet consiste à concevoir une application pour simuler l'exploration d'une équipe dans des environnements qui contient des dangers tels que des monstres variés, des obstacles, des forêts et des trésors. Le but du projet est de trouver tous les trésors le plus rapidement possible mais aussi d'éviter au maximum les pertes des explorateurs. Ce projet s'inscrit dans le domaine de la programmation multi-agents en temps réel et communicant, qui permet de modéliser des systèmes composés d'entités autonomes qui interagissent les unes avec les autres dans un environnement donné. Il met en œuvre des concepts clés tels que la planification, la communication, la coopération et la coordination. Pour réaliser ce projet nous avons besoin de maîtriser des compétences techniques telles que la programmation en Java multithreading et l'utilisation d'une IHM graphique comme Java Swing. Il est également très important de comprendre les notions de communication entre les agents et de stratégie de coordination, mais aussi les différentes approches possibles pour atteindre les objectifs fixés en amont. Enfin, ce projet nous offre une opportunité pour travailler en équipe, de développer nos compétences en matière de collaboration et de gestion de projet, ainsi que de mettre en pratique les compétences acquises tout au long de notre formation en Licence Informatique.

1.2 Vision d'ensemble du rapport

Dans ce rapport, nous présentons la mise en place d'un logiciel d'exploration autonomes et communicants. Dans une première partie nous présenterons ce dont nous attendons du logiciel. Dans une deuxième partie nous verrons en détail la structuration et la conception du logiciel. Dans une troisième partie nous présenterons le déroulement de la conception durant ce semestre. Enfin nous terminerons par un retour sur notre expérience et nous présenterons de futures améliorations concernant notre application.

2 Spécification du projet

Notre logiciel se base sur des notions bien définies, conduisant à des fonctionnalités souhaitées et décrites dans cette partie

2.1 Notions de base du projet

2.1.1 Player

Les explorateurs sont les personnages clés de l'exploration. Leur mission est de parcourir l'environnement à la recherche de trésors, tout en évitant les dangers tels que les monstres et les obstacles. Les explorateurs peuvent être configurés avec différentes stratégies d'exploration, en fonction des objectifs de l'équipe. La communication entre les explorateurs est également essentielle pour une exploration réussie. Les explorateurs doivent être en mesure de communiquer entre eux pour partager des informations sur les trésors trouvés, les dangers rencontrés, et pour coordonner leurs mouvements. La définition d'un protocole de communication efficace est donc un élément clé de la réussite de l'exploration. En résumé, les explorateurs sont les protagonistes de l'exploration, leur mission est de trouver des trésors et d'éviter les dangers. Ils sont configurés avec des stratégies d'exploration différentes et doivent communiquer entre eux pour coordonner leurs mouvements et maximiser les chances de succès de l'équipe.

2.1.2 Monstre

Les monstres jouent un rôle important dans la simulation de l'environnement d'exploration, car ils constituent la menace pour l'équipe d'explorateurs. Les monstres peuvent attaquer les explorateurs s'ils les rencontrent sur leur chemin, réduisant ainsi leur effectif et mettant en péril la réussite de l'exploration. Les monstres sont souvent placés dans des zones spécifiques de l'environnement, comme des forêts ou des montagnes, ou près des trésors pour les protéger. Leur présence peut donc rendre la récupération des trésors plus difficile et nécessiter une stratégie adaptée pour éviter les pertes. En conséquence, il est très important de bien gérer les monstres pour réussir l'exploration. Les explorateurs doivent être en mesure de repérer leur présence et de déterminer comment les éviter ou les combattre en fonction de leur stratégie d'exploration. Cela nécessite une coordination efficace de l'équipe et une bonne gestion des ressources pour minimiser les pertes et maximiser les chances de succès. En résumé, les monstres sont une menace pour l'équipe d'explorateurs et nécessitent une stratégie adaptée pour éviter les pertes. La gestion efficace des monstres est donc un élément clé de la réussite de l'exploration.

2.1.3 Trésor

Les trésors jouent un rôle central puisqu'ils constituent la condition de victoire pour l'équipe d'explorateurs. Le but de l'exploration est en effet de trouver et de récupérer tous les trésors présents dans la carte le plus rapidement possible. Les trésors sont souvent cachés ou protégés par des obstacles ou des dangers tels que des monstres. La récupération des trésors peut se faire de manière individuelle ou collective, en fonction de la stratégie adoptée par l'équipe d'explorateurs. Cependant, il est important de noter que la réussite de l'exploration dépend de la capacité de l'équipe à trouver tous les trésors sans subir de pertes importantes. En effet, les monstres présents dans l'environnement peuvent attaquer les explorateurs et réduire leur effectif, ce qui peut compromettre la réussite de l'exploration. En résumé, les trésors constituent l'objectif principal de l'exploration et sont un élément

clé de la victoire de l'équipe. Leur récupération nécessite une bonne coordination et une stratégie efficace pour minimiser les risques et maximiser les chances de succès.

2.1.4 Carte

La carte est un élément essentiel de l'exploration en équipe dans un environnement complexe. Elle permet de visualiser la topographie du terrain, les éléments qui le composent (trésors, obstacles, forêts, dangers), ainsi que la position des membres de l'équipe d'explorateurs. La carte permet également de suivre la progression de l'exploration, en identifiant les zones déjà explorées et celles qui restent à découvrir.

2.2 Fonctionnalités attendues du projet

2.2.1 Carte au trésor

L'application doit permettre la création de l'environnement en créant une carte au trésor avec des éléments tels que les trésors, les obstacles, les forêts et les dangers. Dans notre projet, la carte de trésor est créée dans un fichier texte qui contient une série de nombres. Chaque nombre correspond à une texture, c'est-à-dire un bloc de la carte. Par exemple, le nombre 2 correspond à une tuile d'herbe, le nombre 16 à celui d'un arbre vert. La carte est donc composée de différentes tuiles qui représentent les éléments de l'environnement : les trésors, les obstacles, les forêts, les dangers, etc. Ces tuiles sont arrangées dans une structure de grille qui représente la carte dans son ensemble. En utilisant cette méthode, nous pouvons facilement créer et modifier la carte en modifiant simplement les nombres dans le fichier texte. Cela nous permet également de stocker de grandes quantités d'informations sur la carte de manière efficace et de la charger facilement dans l'application.

```
2 12 10 10 13  2  2  2  2
2  8  3  3  7  2  2  2  2
2  8  3  3  7  2  2  2  2
2  8  3  3  7  2  2  2  2
2  8  3  3  7  2  2  2  2
2  8  3  3  7  2  2  2  2
2  8  3  3  7  2  2  2  2
49 8  3  3  7 49 49 49 49
49 8  3  3  7 49 49 49 49
```

Figure 1 - Exemple d'un fichier texte d'une carte

Figure 2 - Tuile d'un arbre vert

2.2.2 Stratégie de l'équipe d'explorateurs

La création et la configuration de la stratégie de l'équipe d'explorateurs sont essentielles pour la réussite de l'exploration de la zone. Les règles qui régissent le comportement de l'équipe pendant

l'exploration sont cruciales pour garantir que tous les trésors sont trouvés et que l'équipe reste en sécurité.

Nous avons 11 règles prédéfinies qui décrivent le déroulement de l'exploration du terrain et la recherche des trésors qui sont :

1. Tous les personnages atteignent l'entrée de la zone d'exploration avant de chercher le trésor.
2. Après avoir atteint l'entrée de la zone, l'emplacement du trésor est connu.
3. Après avoir atteint l'entrée de la zone, l'emplacement du trésor est inconnu.
4. Si les monstres sont trop puissants, l'explorateur va demander de l'aide à un autre explorateur qui viendra l'aider.
5. Ne pas tuer les monstres, il faut contourner les monstres (les monstres seront considérés comme des obstacles).
6. Continuez à marcher dans la direction actuelle, ne changez pas de direction jusqu'à ce que vous rencontriez un obstacle.
7. Les personnages sont connus pour explorer l'entrée de la zone. (?)
8. On ne trouve qu'un trésor par personne cela suffit.
9. Il faut trouver tous les trésors sur la carte.
10. Le point de départ du personnage se situe dans la zone d'exploration.
11. Le point de départ du personnage est en dehors de la zone d'exploration.

Et ainsi avec ces différentes règles établies, nous avons fait 3 stratégies qui sont :

- Stratégie 1 : Cette stratégie commence par placer tous les personnages en dehors de la zone d'exploration, puis les faire avancer en ligne droite jusqu'à l'entrée de la zone. Une fois que tous les personnages ont atteint l'entrée de la zone, ils se séparent pour explorer la carte. La priorité est donnée aux trésors proches et aux zones encore inexplorées (1, 2, 5, 7, 9, 11).

- Stratégie 2 : Cette stratégie commence par placer tous les personnages à l'intérieur des zones d'explorations, près de l'entrée. Les personnages se séparent pour explorer la carte, avec une priorité donnée aux trésors proches et aux zones encore inexplorées (1, 2, 5, 7, 9, 10).

- Stratégie 3 : : Cette stratégie commence par placer tous les personnages en dehors de la zone d'exploration, puis les faire avancer en ligne droite jusqu'à l'entrée de la zone. Les personnages ne connaissent pas l'emplacement des trésors et doivent parcourir la zone d'exploration et affronter les monstres s'ils n'ont pas le choix. Cependant, ils doivent appeler de l'aide aux autres explorateurs si le monstre est trop puissant (1, 3, 4, 11).

Figure 3 - Choix des stratégies

2.2.3 Protocole de communication entre explorateurs

Le protocole de communication entre explorateurs est crucial pour assurer une exploration efficace et coordonnée. Dans notre projet, plusieurs méthodes ont été utilisées pour la communication entre les explorateurs. Tout d'abord, le multithreading Java a été utilisé pour permettre à plusieurs explorateurs de travailler simultanément sur la même carte. Pour garantir que chaque explorateur reçoive l'information au bon moment, les méthodes wait() et notify() sont utilisées pour permettre à un thread d'attendre un signal d'un autre thread avant de continuer son exécution.

Ensuite, pour éviter les conflits lors de l'utilisation de variables partagées entre les explorateurs, un système de verrouillage (lock) a été mis en place. Ce système garantit qu'un seul explorateur accède à une variable partagée à la fois, évitant ainsi les conflits de données. Enfin, les variables ont été partagées de manière sélective et isolée pour garantir que chaque explorateur dispose des informations nécessaires sans surcharger le système avec des données inutiles. Par exemple, seules les informations pertinentes telles que la position des trésors et des monstres encore en vie sont partagées entre les explorateurs. Dans l'ensemble, ces méthodes de communication entre les explorateurs garantissent que l'exploration se déroule de manière coordonnée et efficace, permettant ainsi de maximiser les chances de succès de l'équipe.

2.2.4 La restitution de l'exploration avec des résultats intermédiaires et finaux

La réalisation du déroulement de l'exploration consiste à suivre les différentes étapes de l'exploration en temps réel, en fournissant des résultats intermédiaires et finaux. Lors du démarrage de la simulation, toutes les informations de la carte sont disponibles pour l'utilisateur. Toutefois, pour éviter toute surcharge d'informations à l'écran, seules les informations pertinentes telles que le nombre de trésors trouvés et le nombre de monstres encore en vie seront affichées. En outre, le temps de simulation sera aussi visible pour permettre à l'utilisateur de suivre l'évolution de son équipe d'explorateurs. Cela permettra de fournir des résultats intermédiaires en temps réel, ainsi qu'un résultat final une fois la simulation terminée. Et ces résultats finaux permettront d'évaluer la réussite de l'exploration et de comparer les performances des différentes stratégies choisis.

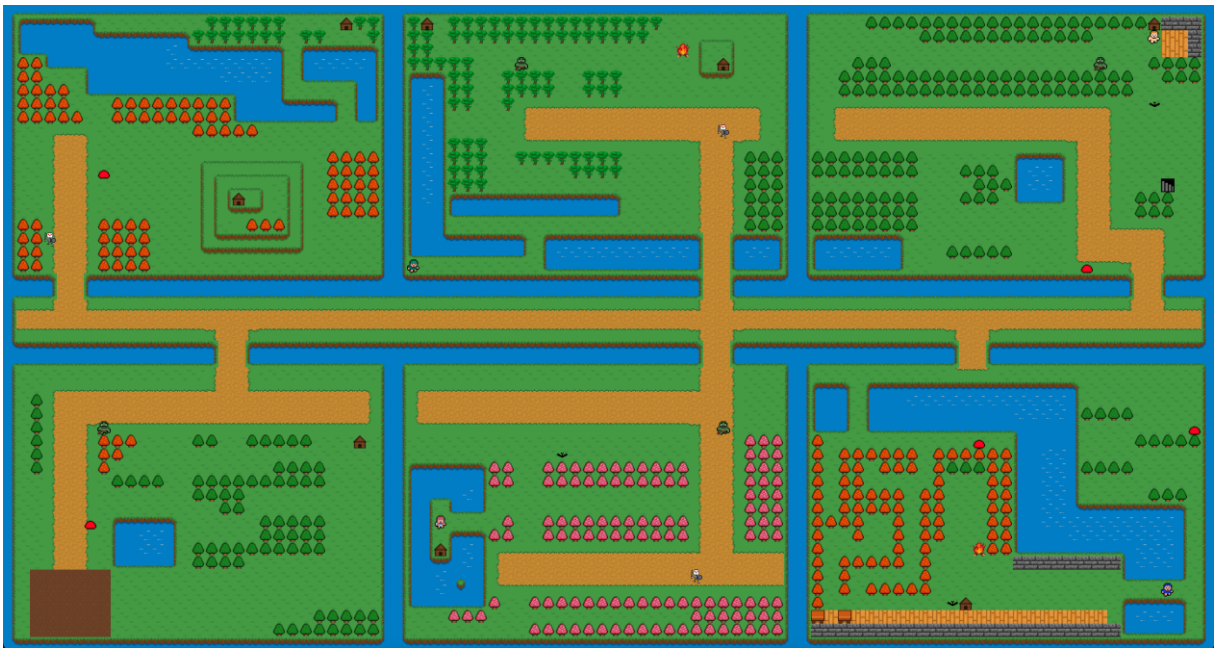

Figure 4 - Carte finale lorsque tous les trésors ont été trouvés

3 Conception et réalisation du projet

Ce projet comporte six paquets, « Astar », « config », « engine », « gui » et « res », chacun ayant son propre rôle à jouer. Vous trouverez une description de ce que chaque paquet fait à son tour et les classes Java qu'il contient.

1. Le package « Astar » contient la classe « Astar » et la classe « Node », qui implémente l'algorithme A* pour la recherche automatique de chemin lorsque le personnage a une destination connue.

2. Le package « config » contient la classe « GameConfiguration », qui définit les variables globales, conçoit la taille de la fenêtre, la taille de tuile et le taux de rafraîchissement de la vitesse de mouvement.

3. Le package « engine » contient 4 sous-packages en même temps, à savoir le package « element », le package « map », le package « process » et le package « util ».

   3.1. Le sous-package « element » contient les classes « ExploreEdge », « Mission », « Monster », « Object », « Treasure » et « Player ». Il définit les principaux éléments du jeu, tels que les monstres, les 4 players et les missions de player, etc. La classe « Object » est la classe basique des trois autres classes : « Monster », « Treasure » et « Player ».

   3.2. La classe « Monster » hérite les attributs de la class « Object », elle a aussi sa propre attribut : type (0, 1, 2, 3). Type(0) présente un monstre chauve-souris. Type(1) présente un monstre vase. Type(2) présente un monstre lutin. Type(3) présente un monstre squelette. Par défaut, Type(3) est trop fort, il faut deux players à battre.

   3.3. La classe « ExploreEdge » définit le bord de la zone indiquée, elle permet aux players de connaître la zone d'exploration et de ne déplacer pas en dehors de la zone d'exploration.

   3.4. La classe « Mission » définit la mission indiquée, elle permet aux players de connaître sa mission courante, tels que l'entrée de la zone, le bord de la zone, la priorité (0 ou 1) de la mission. La priorité 0, c'est-à-dire une mission d'exploration. La priorité 1, une mission d'aide.

   3.5. Le package « map » contient deux classes « Block » et « Map ».

   3.6. La class « Block » définit chaque block de la carte. Il a comme attributs : ligne, colonne et la tuile correspondante. Le numéro de tuile de 0 jusqu'à 50, chaque numéro représente de différentes éléments, tels que un arbre, chemin, maison et herbe, etc.

   3.7. La class « Map » a 3 attributs : blocks (Tableau à deux dimensions), lineCount et colCount. Elle enregistre les informations de la carte, la taille de la carte.

   3.8. Le package « util » contient une seule class « ObstacleChecker », il permet aux players de détecter l'étape suivante est un obstacle ou pas.

   3.9. Le package « process » contient 3 classes: « ElementManager », « GameBuilder » et « PlayerManager ».

3.10. La classe « ElementManager » définit des données différentes, par exemple un récipient qui contient des monstres, un récipient qui contient des trésors et encore un récipient qui contient des missions, etc.

3.11. La classe « GameBuilder » permet aux utilisateurs d'initialiser le jeu. Elle crée des monstres en mettant ses propres positions, crée des trésors en mettant ses propres positions, crée 4 players(thread) qui implémentent l'interface de 'Runnable'. Elle insère des récipients comme des paramètre pour la fonction de constructeur de player pour construire des données partagées, etc.

3.12. La classe « PlayerManager » qui implémente l'interface de 'Runnable' est le protagoniste de ce projet. Il contient plusieurs de fonctions pour permettre aux players de déplacer. Aussi, il contient la fonction « randomMove » , qui implémente le mouvement aléatoire, qui est utilisée pour l'exploration aléatoire du personnage dans un environnement inconnu. D'ailleurs, il contient une méthode 'switch' pour sélectionner un des stratégies.

4. Le package « gui » contient les trois classes « GameDisplay », « MainGUI » et « PaintStrategy ».

4.1. La classe « PaintStrategy » utilisez le package Graphics pour réaliser le dessin de la carte, ainsi que des éléments de dessin sur la carte, des trésors, des personnages, des cartes, etc.

4.2. La classe « GameDisplay » utilise le package Graphics et appelle la méthode paint de « PaintStrategy » pour réaliser le dessin de la carte, ainsi que des éléments de dessin sur la carte, des trésors, des personnages, des cartes, etc.

4.3. La classe « MainGUI » implémente l'initialisation de la fenêtre de la carte, l'exécution du thread principal et l'instanciation de La classe « GameDisplay », et appelle la fonction d'initialisation de l'élément de « ElementManager ».

5. Le package « res » contient tous les ressources d'éléments. Les images de players, des monstres, de trésors, d'herbe, d'arbre, d'eau, etc.

4 Rendu final

Nous allons donc voir comment un utilisateur pourra se servir de notre logiciel et comment il pourra visualiser le bon fonctionnement de notre projet.

4.1 Interface utilisateur finale

Le thread principal définit une méthode d'entrée qui permet aux utilisateurs à taper le choix de stratégie dans console.

```
Enter the strategy you want(1, 2 or 3):
```

Figure 5 - Choix des stratégies

4.2 Tests utilisateur

Voici ce que voit l'utilisateur lorsqu'il lance le logiciel de simulation d'exploration, on voit une grande carte constitués d'un pont au centre qui relie les 6 îles. Ainsi que les explorateurs, les monstres, les trésors et les décors.

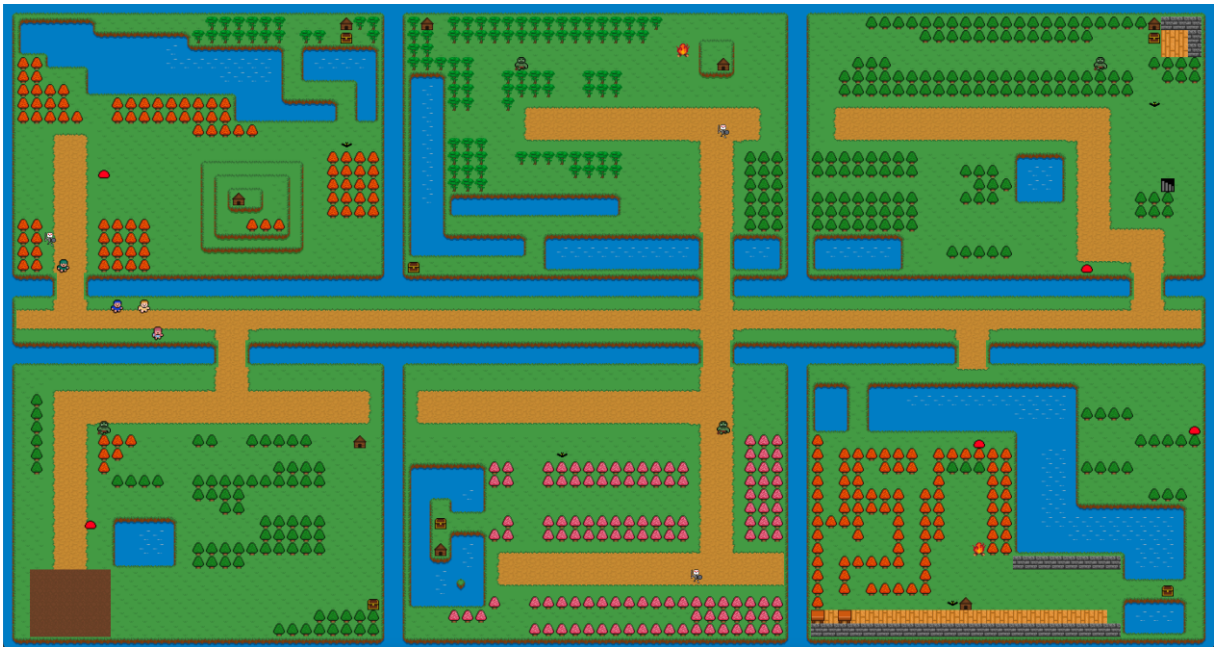

Figure 6 - Initialisation de la carte

5 Gestion de projet

Dans ce chapitre, nous allons décrire la répartition des tâches, les méthodes de travail et les choix faits lors des imprévus qui sont arrivés durant le projet

5.1 Méthode de gestion

Au cours de notre projet, nous avons choisi d'utiliser Discord pour faciliter la communication entre les membres de l'équipe. En plus de fournir un moyen simple et rapide pour communiquer, Discord a également permis un partage facile des codes sources et des documents, ce qui nous a permis de travailler efficacement même à distance. Nous avons créé un serveur Discord dédié à notre projet et chaque membre de l'équipe a été invité à rejoindre le serveur. Nous avons utilisé les différents canaux de discussion pour organiser des réunions en ligne, échanger des idées, poser des questions et discuter des avancées du projet. L'utilisation de Discord a été bénéfique pour notre projet car cela nous a permis de rester en contact régulier et d'avoir une communication fluide. Nous avons également pu utiliser les fonctionnalités de partage de fichiers pour échanger rapidement des codes sources, des captures d'écran et des documents de travail. Dans l'ensemble, l'utilisation de Discord a été une méthode efficace pour travailler en équipe à distance et nous recommandons cette plateforme de communication pour tout projet collaboratif même si utiliser GitHub reste mieux pour le partage de code.

Pour organiser l'avancement du projet nous avons donc planifié des releases :

Date	Fonctionnalités prévues	Fonctionnalités réalisées
Janvier	Conception des règles et des stratégies. Réalisation initialement de l'interface graphique	Implémentation de visualisation de l'interface graphique
Février	Amélioration de la structure du projet.Procéder à l'initialisation des éléments requis de la carte.	La boucle principale est bien construite, les éléments (monstre et trésor, etc.) sont bien met dans la carte, et 4 players déplace aléatoirement.
Mars	Implémentation des 3 stratégies. Utilisation de mécanisme de mutex et de 'wait' et de 'notify'.	Les 3 stratégies sont presque finies.
Avril	Amélioration des 3 stratégies. Ajout d'une entrée permet aux utilisateurs de choisir une stratégie. Ecriture du rapport	Les 3 stratégies sont bien réalisées.

Tableau 1 - Planification des avancées du projet

5.2 Répartition des tâches

Yicheng QIAN et **Linxiang CONG** sont responsables de la construction de la structure du projet, de l'écriture du code et de l'écriture du rapport.

Thanh-Tri CALIMOUTTOU est chargé d'écrire une partie de code, de vérifier la qualité du programme et de rédiger le rapport.

Qinyu ZHANG prépare les ressources des images, fait la conception des règles de jeu et de tester le jeu.

6 Conclusion et perspectives

La finalisation de ce projet nous permet d'avoir un regard plus critique sur sa conception mais aussi sur ses possibilités d'évolutions

6.1 Bilan final

Ce projet nous a permis d'apprendre l'organisation d'un projet et la structuration des données. Nous avons pu constater les différents problèmes pouvant survenir dans la conception mais aussi dans la gestion des tâches à réaliser. Nous n'avons pas réussi à implémenter toutes les tâches souhaitées et en avons déduit que le temps est une grande contrainte dont la maitrise est essentielle. C'est grâce à des expériences comme celle-ci que nous pourrons nous améliorer dans notre organisation individuelle et collective mais aussi dans notre compréhension du langage.

6.2 Améliorations possibles du projet

L'état du projet à ce point est une version que nous pouvons bien sûr améliorer. Nous pouvons tout d'abord ajouter les points non traités par manque de temps, comme le choix des types d'explorateurs, une vraie fenêtre qui affiche tous les statistiques de la partie (pendant et après la simulation). Les exemples cités font partie de la forme mais nous pouvons aussi améliorer le contenu proposé. Nous pouvons ajouter plus de choix de types d'explorateurs, plus de biomes, une plus belle interface, de la musique. Ce sujet nous laisse une grande possibilité d'amélioration et d'ajout de contenu permettant une grande créativité dans la conception.

Remerciements

Avant de conclure, nous tenons à exprimer nos sincères remerciements à tous ceux qui ont contribué à la réalisation de ce projet. Tout d'abord, nous remercions notre encadrant M.LIU pour sa précieuse aide, ses conseils avisés et son soutien tout au long de ce projet. Nous sommes reconnaissants pour la qualité de ses enseignements et de son accompagnement. Nous remercions également tous les membres de l'équipe qui ont travaillé dur pour mener à bien ce projet. Leurs efforts, leur créativité et leur persévérance ont été indispensables à la réussite de cette expérience. Nous souhaitons également remercier les personnes qui ont pris le temps de lire ce rapport et de suivre notre travail. Nous espérons que vous avez pu apprécier notre sujet et notre raisonnement. Enfin, nous remercions CY Cergy Paris Université pour nous avoir donné l'opportunité de réaliser ce projet et de mettre en pratique nos connaissances. Nous espérons que cette expérience nous sera utile pour notre avenir professionnel.