Tales Of The Apocalypse

Rapport de Projet

Yanis CHAABANE

Jordan FAILLOUX

Tanguy DESGOUTTES

Marc-Emmanuel RAIFFE

Table des matières

1 Introduction **5**
- 1.1 Présentation du projet . . . 5
- 1.2 Présentation de FIG . . . 5
- 1.3 L'équipe de développeurs . . . 6
  - 1.3.1 Tanguy Desgouttes (Chef de groupe) . . . 6
  - 1.3.2 Yanis Chaabane . . . 6
  - 1.3.3 Jordan Failloux . . . 7
  - 1.3.4 Marc-Emmanuel Raiffe . . . 7

2 Rappel du cahier des charges **8**
- 2.1 Objectifs . . . 8
  - 2.1.1 Les survivants . . . 8
  - 2.1.2 Un monde vivant . . . 9
  - 2.1.3 Rythme et dramaturgie . . . 10
  - 2.1.4 Les apocalypses . . . 10
- 2.2 Répartition du travail . . . 11

3 Conception **12**
- 3.1 Génération de villes . . . 12
- 3.2 Génération du monde . . . 13
- 3.3 Pathfinding . . . 14
- 3.4 IA . . . 14
- 3.5 Multijoueur . . . 16
- 3.6 Menus & Interfaces utilisateurs . . . 20
- 3.7 Textures, Modèles & Animations . . . 27
- 3.8 Autres fonctionnalités jouables . . . 31
- 3.9 Son & Musique . . . 36
- 3.10 Site Web . . . 36

4 Accord avec le planning **39**

5 Réalisation **40**
- 5.1 Fonctionnalités jouables . . . 40
  - 5.1.1 Lors de la première soutenance . . . 40
  - 5.1.2 Lors de la deuxième soutenance . . . 40
  - 5.1.3 Fonctionnalités jouables actuelles . . . 40
- 5.2 Possibles améliorations . . . 42
- 5.3 Logiciels utilisés . . . 43
- 5.4 Coûts matériels et logiciels . . . 44
- 5.5 Modèle économique . . . 44
- 5.6 Futur de *Tales of the Apocalypse* . . . 45
- 5.7 Problèmes & Solutions . . . 45

6 Synthèse des expériences individuelles **47**
6.1 Tanguy 47
6.2 Yanis 47
6.3 Jordan 48
6.4 Marc-Emmanuel 48

7 Conclusion **49**

8 Annexes **50**
8.1 Sources 50
8.2 Annexes 50

1 Introduction

1.1 Présentation du projet

Tales of the Apocalypse est un jeu vidéo en 3D visant à recréer une expérience d'un groupe de survivants lors d'une apocalypse. Le joueur aura alors le contrôle total de l'ensemble des membres du groupe et aura pour tâche de les faire survivre. Chaque survivant est unique et possède des traits, des compétences et des caractéristiques représentant ses forces et ses faiblesses. *Tales of the Apocalypse* n'est ni un jeu de stratégie compétitif ni un simulateur, mais est avant tout un générateur d'histoires, ainsi l'intérêt premier du jeu n'est pas de gagner mais d'assister aux différents événements, qu'ils soient épiques, comiques, tragiques ou dramatiques, vécus par le groupe de survivants.

FIGURE 1 – *Tales of the Apocalypse*

1.2 Présentation de FIG

Filiga Inspiration Games (ou FIG) est un groupe formé de quatre étudiants de l'EPITA : Tanguy Desgouttes, Yanis Chaabane, Marc-Emmanuel Raiffe et Jordan Failloux. FIG est spécialisé dans la création de jeux vidéos, nous travaillons actuellement exclusivement sur le jeu vidéo *Tales of the Apocalypse* (abrégé TotA). Nous espérons combler notre manque d'expérience dans le domaine par notre détermination au travail et nos connaissances dans le domaine du jeu vidéo. La création de notre groupe se repose sur un mélange harmonieux entre une bonne entente déjà existante, un mélange de différentes compétences et expériences dans le domaine du jeu vidéo et un intérêt partagé pour le concept de TotA.

Le nom de notre groupe, Filiga Inspiration Games, provient d'une réflexion sur nos forces potentielles : la diligence au travail ("Filiga" signifie "diligent" en Samoan) et nos connaissances de différents jeux dont nous allons nous inspirer ("Inspiration"). Enfin, "Games" représente simplement le domaine dans lequel notre groupe a décidé de se spécialiser.

Le logo de notre groupe est la représentation abstraite d'une figue, provenant de l'abréviation FIG du nom de notre groupe. La figue apparaît comme étant pixelisé reflétant la direction artistique de TotA tournée alors vers un style cubique.

1.3 L'équipe de développeurs

1.3.1 Tanguy Desgouttes (Chef de groupe)

Étant chef de projet et concepteur du projet, celui-ci me tient particulièrement à coeur. L'idée originale de *Tales of the Apocalypse* peut remonter jusqu'à 3 ans, le premier prototype s'appelait alors *Projet AZ*. Si le jeu a beaucoup évolué au cours des années, l'objectif principal est resté inchangé : satisfaire la question que j'ai pu souvent entendre, "Que ferais-je lors d'une Apocalypse ?". Cette idée de jeu m'a amené à découvrir l'informatique et à me familiariser avec de nombreux langages (de GML à C++). Au fil des années, je me suis découvert un intérêt tout particulier dans le développement de jeux vidéos et plus généralement des nouvelles technologies, la prise de décision lors du développement d'un projet me fascine. Ce projet est donc pour moi l'une des nombreuses raisons pour lesquelles j'ai rejoint l'EPITA. J'ai pu rencontrer et recruter des personnes compétentes et sérieuses qui, de par leurs différentes expériences, apportent un regard nouveau sur le projet. J'ai une confiance absolue en notre groupe et en la réussite du projet.

1.3.2 Yanis Chaabane

Ayant une passion pour l'informatique depuis 2 ans, je m'intéresse à tout ce qui touche aux nouvelles technologies, aux langages de programmation. De par cette passion j'ai d'abord fait le choix d'une Terminale S avec option ISN où j'ai fait mes premiers pas dans la programmation et où j'ai pour la première fois codé en Python et dans les domaines du web. Ce fut d'une grande aide pour moi lors du choix de mon orientation.

Le projet d'ISN fut un projet important qui m'a permis de développer un jeu sur de longs mois et m'a permis évidemment d'améliorer mes compétences dans différents langages de programmation mais aussi de prendre part à un travail de groupe avec la répartition des tâches, ce qui va pouvoir me servir dans ce nouveau projet.

De par cette expérience que j'ai pu avoir l'année dernière je me sens prêt à relever de nouveaux défis lors de ce projet beaucoup plus important. J'ai hâte de pouvoir voir l'avancé de notre jeu et de nos compétences. Étant organisé et perfectionniste, je me donnerai à fond dans ce projet afin de pouvoir en tirer un jeu qui sera capable de sortir sur les différents marchés de jeux vidéos mais aussi d'en tirer un réel avantage personnel pour mes années à venir à l'EPITA.

1.3.3 Jordan Failloux

Je m'appelle Jordan FAILLOUX et je viens de Tahiti. J'ai toujours été intéressé par l'informatique, c'est pour cela que j'ai choisi de faire une Terminale S option ISN. Pour mon avenir professionnel, j'hésitais alors entre la médecine et l'informatique. Étant présent sur le territoire Polynésien, j'ai d'abord décidé de faire PACES. Au cours de l'année, je me suis aperçu que ça ne me plaisait pas. J'ai donc cherché des écoles d'ingénieurs informatique en France et je suis arrivé ici, à l'EPITA.

En ISN, j'ai eu l'occasion de faire un site internet et un jeu vidéo en Python en tant que projet en équipe pour le bac. Ce dernier m'a permis de prendre goût à la réalisation de programmes en groupe.
Pour ce projet, je compte utiliser mes expériences acquises précédemment et ne pas refaire les mêmes erreurs pour avoir un meilleur résultat final.

1.3.4 Marc-Emmanuel Raiffe

J'ai toujours été intéressé par les sciences et à l'aise dans ce domaine. Ces deux raisons m'ont assez naturellement orienté vers un bac S-SI et l'affection que j'avais pour la programmation m'a amené à l'EPITA. Même si je n'ai jamais vraiment appris de langages informatiques avant mes études supérieures je reste confiant pour le projet car les travaux de groupe sont les activités où je suis le plus productif et j'ai toujours beaucoup apprécié les jeux vidéos ce qui m'a permis d'acquérir beaucoup de connaissances maintenant utiles.
En terme de compétences plus strictes je me suis découvert un niveau suffisant en programmation malgré le manque d'expérience. Cela dit je maîtrise certains logiciels comme magicavoxel qui permettront de créer les modèles 3D et illustrations utiles au projet.

2 Rappel du cahier des charges

2.1 Objectifs

Tales of the Apocalypse se présente comme un projet ambitieux mélangeant différents éléments de jeux populaires. Il se démarque par une génération procédurale [1] du monde en 3D, des personnages distincts et la présence d'événements afin de produire une histoire marquante chez le joueur.

2.1.1 Les survivants

Pour permettre l'attachement du joueur à ces survivants, chaque survivant se doit d'être unique. Ainsi, il a un nom, un âge, un sexe et surtout, des attributs et des compétences.

1. **Attribut :**
   Un attribut est une valeur comprise entre 1 et 100 utilisé passivement ou activement. Pour l'instant, nous avons décidé d'implémenter 5 attributs : Force, Perception, Social, Mental et Intelligence. Un exemple d'utilisation passive d'un attribut serait l'attribut Force pour déterminer le poids maximum qu'un survivant peut transporter sur lui.

2. **Compétence :**
   Une compétence peut être comparé à un attribut plus spécifique, elle peut être un bonus à un attribut dans certaines conditions (par exemple Mêlée donne un bonus de Force pour l'action Frapper) mais aussi une valeur reflétant un niveau d'expertise dans un domaine d'un survivant (par exemple, Soin et Bricolage influent sur les actions "Soigner" et "Construire une table" et ne donnent pas de bonus à un attribut).

3. **Trait et création d'un survivant :**
   Par défaut, les attributs d'un survivant sont tous égaux à 50 et ses compétences à 0 mais lors de sa création, des traits lui sont attribués, soit aléatoirement soit par le joueur. Ces traits influent sur ses statistiques et ses compétences, par exemple, le trait "fort" donne +10 à l'attribut Force, le trait "médecin" donne +20 à l'attribut Intelligence et +2 à la compétence Soin. Il existera un très grand nombre de traits afin de permettre l'unicité de chaque survivant.

1. Création de niveaux à grande échelle de manière automatisée répondant à un ensemble de règles définies par des algorithmes.

4. **Action :**
   Une action est une interaction entre un survivant et ce qui l'entoure (dont lui). La réussite d'une action dépend de la valeur de l'attribut associé à l'action du survivant qui réalise l'action. Un nombre aléatoire compris 1 et 100 est comparé à la somme entre la valeur dudit attribut et le facteur de difficulté de l'action (positif ou négatif), si le nombre aléatoire est inférieur ou égal à cette somme, l'action est une réussite, sinon c'est un échec.

5. **Besoins :**
   Chaque survivant a besoin de nourriture. La nourriture se trouve soit dans les bâtiments à explorer, soit en tuant des animaux, soit en la faisant pousser (à noter que la soif ne sera pas implémenté car rajoutant de la complexité inutile).
   Chaque survivant a aussi une jauge de moral et une jauge de stress, chacune augmente et diminue en fonction de ce qui arrive au survivant et à son entourage.
   L'état du survivant est représenté par l'état de ses organes et de ses parties du corps.
   La température à un impact sur la santé mental et physique du survivant.

6. **Inventaire, fouille et objet :**
   Les survivants, tout comme les meubles, ont des espaces de stockage représentés par un tableau. Un objet occupe toujours un emplacement du tableau. Un inventaire est aussi limité par le poids maximum que peut porter un survivant. Les meubles ou d'autres objets trop lourds ne peuvent pas être rangés dans l'inventaire.

2.1.2 Un monde vivant

Un monde vivant et interactif est vital pour une bonne histoire de survie.

1. **Biomes :**
   Le monde de TotA est un monde généré aléatoirement, il sera composé de différents biomes plus ou moins remplis de bâtiments préfabriqués. La ville possédera quelques bâtiments différents comme des immeubles, un supermarché, etc. Le joueur pourra se déplacer librement entre les différents endroits. L'objectif du générateur est de pouvoir ajouter des nouveaux biomes et des nouveaux bâtiments facilement.

2. **Faune et la flore :**
   En terme de faune, notre objectif est d'implémenter un rat en tant que preuve de concept. De même, seuls un arbre, un buisson à baie et un plant à tomate seront implémentés pour montrer certaines actions possibles comme couper un arbre, récolter et planter.

2.1.3 Rythme et dramaturgie

Le maintien d'un rythme est crucial. Il existe de nombreuses façons d'empêcher un jeu d'être répétitif et monotone.

1. **Événement :**
   Les événements dans TotA seront à intervalles plus ou moins réguliers et peuvent être positifs comme négatifs, par exemple une vague de froid, une attaque de bandit ou même l'apparition d'une horde de zombie.

2. **PNJ (Personnage non-jouable) :**
   Les survivants contrôlés par le joueur ne sont pas les seuls humains restants dans l'univers de TotA. D'autres groupes de survivants se maintiennent aussi en vie à leur façon, certains voudront rejoindre le groupe ou faire du troc mais d'autres voudront l'attaquer.

3. **Progression et évolution :**
   Au fur et à mesure du jeu, les connaissances du groupe s'enrichiront à l'aide d'autre survivants ou grâce à des livres. La recherche de ces technologies du passé permettent une avancée progressive du joueur dans le monde de TotA, la complexité du jeu augmente donc au fil de la partie. Individuellement, chaque survivant peut changer au fil du temps, et ce que ce soit en bien (apprentissage dans un certain domaine) ou en mal (traumatisme et blessure).

2.1.4 Les apocalypses

Nous souhaitons offrir au joueur une expérience optimale, ainsi le joueur avant de commencer une partie pourra choisir entre différents scénarios possibles. Le premier scénario que nous souhaitons implémenté est "Seul", le joueur contrôle un seul survivant face à un monde sans animaux ni humains. Deux autres scénarios que nous souhaitons fortement ajouter est "Grand Froid" (température glaciale, peu de vie, pas de zombie) et "Romero" (présence de zombie).

2.2 Répartition du travail

	Responsable	**Suppléant**
Programmation		
Génération de villes	Tanguy	Marc-Emmanuel
Génération du monde	Tanguy	Marc-Emmanuel
Pathfinding	Tanguy	Marc-Emmanuel
IA	Marc-Emmanuel	Jordan
Multijoueur	Tanguy	Yanis
Menus & Interface utilisateur	Yanis	Tanguy
Art		
Textures	Marc-Emmanuel	Jordan
Modèle & Animations	Marc-Emmanuel	Jordan
Son & Musique	Jordan	Tanguy
Site Web		
HTML	Yanis	Jordan
CSS	Jordan	Yanis
Rapport de soutenance		
Rapports de soutenances	Yanis	Tanguy

3 Conception

3.1 Génération de villes

Travail de Tanguy (responsable) :

Pour la première soutenance la génération de la ville a occupé sans aucun doute la plus grande partie de mon travail. Je souhaitais pouvoir créer une ville procéduralement que le joueur puisse explorer complètement. J'ai d'abord commencé par la génération de routes, nos routes étaient principalement caractérisées par leurs "ouvertures" dans les directions cardinales, il suffisait alors de relier des points d'intérêts entre eux en "ouvrant" au fur et à mesure la route. À ce moment, toutes les villes avaient 9 points d'intérêts plus ou moins reliés entre eux, les ouvertures Nord/Sud/Est/Ouest de la ville, un centre approximatif de la ville et 4 points répartis dans chaque cadrans de la ville.

Une fois ce système mis en place, j'ai commencé à réfléchir à un moyen de mettre des bâtiments dans la ville. J'ai décidé de générer des bâtiments seulement à côté des routes, tout d'abord des bâtiments de petites tailles (carrés de 1x1) simples à positionner face à la route puis des bâtiments plus grands (rectangles de 2*1).

Placer ces bâtiments rectangles a été l'une des plus grandes difficultés rencontrées pour la première soutenance, en effet les bâtiments devaient avoir une entrée du côté de la route tout en se générant sur deux emplacements. Au cours du développement de nombreux problèmes ont pu être rencontrés et résolus, par exemple la partie supplémentaire des rectangles n'empêchait pas d'autres bâtiments de se générer. Au final, ce système génère une ville satisfaisante tout en étant extrêmement flexible, l'ajout et la suppression de bâtiment ne demandant même pas de regarder le code.

Pour la deuxième soutenance et bien que la génération de ville fut complétée pour la soutenance précédente, le système a tout de même été amélioré. Tout d'abord, le système de création d'une route d'un point à un autre a été retravaillé, les routes sont désormais moins linéaires et moins monotones, ce tracé moins prévisible améliore grandement la qualité de la ville. De plus, des petites ruelles se génèrent entre les bâtiments de manière dynamique, précédemment ces ruelles étaient attachées directement aux routes. La structure générale des villes a aussi été revue, une fois les premières connexions de routes établies, des routes se génèrent par extension au niveau de routes déjà générées, cela permet de meilleures liaisons au sein de la ville ainsi qu'une génération de ville plus unique et plus réaliste. Enfin, comme prévu, le système de génération permet de générer des bâtiments de taille 2 x 2.

Une nouvelle notion importante faisant partie de la génération de la ville est la génération des "Prop" (accessoire en anglais), ce nom réfère à l'ensemble des meubles, décorations et autres objets. Pour générer ces "Props", j'ai implémenté un système d'emplacements, chaque emplacement étant défini par une liste de "Props" chacun lié à une probabilité de se générer. Pour inclure ce système dans la génération, il suffit alors de placer cet emplacement dans un bâtiment du jeu. Ce système d'emplacements est très flexible, on peut créer et placer des emplacements sans jamais avoir à toucher à du code.

3.2 Génération du monde

Travail de Tanguy (responsable) :

Lors de la première soutenance, nous générions une ville, pour la seconde soutenance nous avons été capable de générer un monde entier. Ce monde possède plusieurs biomes (pour l'instant de types plaine, forêt et aride) où la génération de ces biomes n'est pas complètement aléatoire, ils forment des espaces plus ou moins grands. À l'origine, le monde n'a pas de biomes, on remplit certaines cases qui composent ce monde de biomes, ceux-ci sont appelés origine du biome, on fait alors propager chaque case possédant un biome sur ces cases adjacentes (seulement si la case ne possède pas de biome), ce processus s'arrête une fois que toutes les cases du monde ont un biome. Il existe pour l'instant deux types de villes, les villes et les villages. Les villes sont les mêmes que celles présentées dans la partie "Génération de villes", les villages sont des villes plus petites avec moins de connexions de routes, pas de ruelles entre les bâtiments et possiblement des bâtiments différents de ceux dans les villes. Après avoir généré les biomes, on place aléatoirement les villes et les villages. Ensuite on relie les villes aux villes et aux villages proches par un processus similaire à la création de route entre deux points. Pour pouvoir générer ce monde, il a fallu revoir l'intégralité du système de génération de ville.

De plus, j'ai implémenté la génération de forêts pour la seconde soutenance. La densité et les types d'arbres variant en fonction des biomes. Le système de génération utilise le système de "Prop" décrit dans la partie "Génération de villes". Cette génération de forêt étant très coûteuse en performance, elle occupe une grande partie du temps de génération du monde.

Pour la troisième soutenance, la génération du monde fut finalisée par une optimisation générale du processus, principalement lors de la génération de la fôret, des routes et des bâtiments. La génération du monde et des villes étant donc finie, il ne restait plus qu'à utiliser le système de placement de "Props" créé pour la deuxième soutenance. On a alors pu meubler les bâtiments sans toucher au code. Ce système de placement a été amélioré par l'ajout de la notion du "PropMarkerMaster" qui permet de regrouper plusieurs emplacements de meubles sous une même condition de génération. Par exemple, le placement d'une commode dans un appartement ne dépend ni n'influe sur les autres meubles de l'appartement, maintenant imaginons qu'on veuille rendre certains appartements plus intéressants en faisant apparaître ce qui pourrait ressembler à un camp abandonné de survivants, on peut alors utiliser ce système de "PropMarkerMaster" pour rassembler des "Props" liés exclusivement à ce camp comme un feu de camp, des matelas, des caisses, des bidons remplis d'eau et même des zombies. Ces "Props" ne peuvent apparaître que si le camp apparaît. Ce nouveau système permet donc de créer des ambiances et des environnements plus intéressants, le monde semblant alors vivant.

3.3 Pathfinding

1. Travail de Tanguy (responsable) :

   Pour la première soutenance j'avais travaillé sur l'implémentation d'un système de déplacement par clic de la souris, pour déplacer les personnages, le joueur sélectionne le ou les personnages qu'il souhaite contrôler et a juste à cliquer pour les déplacer. Pour implémenter ce système j'ai utilisé la fonction "NavMesh" ("Navigation Mesh" ou "réseau de navigation") qui permet de définir une surface où un chemin d'un point A à un point B peut être trouvé.
   Le déplacement d'un objet sur cette surface est aussi géré par Unity, l'objet est alors appelé un "agent" (ou "NavMeshAgent"). Il reste alors la sélection de l'objet qu'on souhaite déplacer et la sélection de sa destination, pour se faire, on utilise le système de "RayCast" d'Unity qui permet de détecter des objets sur une ligne (ou un segment) donnée.
   Ainsi, le joueur peut sélectionner un ou plusieurs personnages avec le clic gauche de la souris puis marquer une destination avec le clic droit, tous les personnages sélectionnés vont se déplacer à cette destination. Ce système de déplacement bien que traditionnel convient parfaitement à notre jeu de par sa simplicité.

2. Travail de Marc-Emmanuel (suppléant) :

   L'implémentation du "pathfinding" ayant déjà été faite (jusqu'alors les personnages étaient des cubes se mouvant) il a fallu pour la première soutenance utiliser un "animator", changeant l'animation d'un modèle en fonction des données d'entrée. Les données d'entrée sont issues du "NavMeshAgent" qui nous permet de faire le "pathfinding" et sont retranscrites en une certaine quantité de vitesse et de degré de rotation par une portion de code. Une fois tout cela en place et les différents problèmes résolus on a pu avoir une animation fluide et qui suivait le déplacement du personnage.

3.4 IA

Travail de Jordan :

Pour la deuxième soutenance nous nous sommes donnés comme objectif de réaliser des intelligences artificielles, notamment un comportement de zombie et d'autres personnages non jouables. Nous avions donc pour la deuxième soutenance, comme personnage non jouable, un rat qui pouvait ensuite être adapté à d'autres animaux. Le déplacement aléatoire et la détection de joueur étant les principales caractéristiques de ces intelligences artificielles, toutes mes recherches se sont centrées sur celles-ci. J'ai commencé par coder le script du zombie car c'est un élément principal du jeu.

Le comportement du zombie est le suivant : il erre aléatoirement en s'arrêtant de temps en temps, dès qu'il détecte un joueur il le poursuit et l'attaque s'il est à bonne portée. Tout d'abord pour le déplacement aléatoire, je prends un point aléatoire A dans un rayon de déplacement prédéfini puis j'utilise une fonction d'Unity pour avoir le point le plus proche de A sur la surface

praticable du monde. Le zombie peut ainsi se déplacer sur cette surface. Une des particularités du zombie est qu'il s'arrête parfois de bouger, pour cela j'ai utilisé une coroutine. Une coroutine permet de mettre un temps de pause dans une boucle. Par conséquent, dès que le zombie arrive à destination, il s'arrête quelques secondes puis reprend son déplacement. Ensuite, pour la détection nous devons gérer plusieurs personnages car un joueur peut contrôler plusieurs personnages. Le zombie doit donc pouvoir gérer tous ces personnages. Pour cela, nous stockons tous les personnages à portée du rayon de détection dans une liste et le zombie poursuivra le joueur le plus proche de lui parmi ceux dans son angle de vue.

Le rat a pratiquement les mêmes caractéristiques : il se déplace aléatoirement sans s'arrêter et lorsqu'il rencontre un personnage il fuit au lieu de le poursuivre. De plus, le rat possède un angle de vue de 360 degré donc dès qu'un personnage entre dans son rayon de détection il fuit avec une vitesse augmentée.

J'ai également implémenté une génération aléatoire de noms et de prénoms pour les personnages du joueur. Pour cela, nous avons pris une base de données de noms et de prénoms et nous choisissons lors de la création d'un nouveau personnage un prénom au hasard et un nom au hasard.

Pour cette troisième soutenance, j'ai amélioré la conception du rat qui fuit les personnages lorsqu'ils s'approchent trop près. Si les personnages arrivent à le coincer, ils peuvent le capturer pour avoir de la viande. J'ai aussi implémenté une intelligence artificielle de personnage qui regroupe de nombreux éléments déjà implémentés. Cette intelligence artificielle se déplace aléatoirement, si elle a besoin de quelque chose elle se met à fouiller les bâtiments à la recherche généralement de nourriture ou de bandage, quand elle est blessée ou a faim elle utilise ces objets. Elle récupère aussi des matériaux nécessaires à la construction d'un lit afin de pouvoir se reposer quand elle est fatiguée. A l'approche d'autres personnages, elle fuit de la même façon qu'un rat si elle ne possède pas d'armes sinon elle ne change pas son comportement habituel. Si elle est attaquée par un personnage et qu'elle possède une arme elle attaque en retour sinon elle fuit. Elle fuit aussi automatiquement lorsqu'elle se trouve proche de zombies. Finalement, les personnages peuvent la convaincre de rejoindre leur équipe en utilisant leurs valeurs de Social. L'implémentation de cette intelligence artificielle a été vitale afin de rendre le monde vivant.

3.5 Multijoueur

1. Travail de Tanguy (responsable) :

Pour la première soutenance et après m'être renseigné sur Photon, j'ai commencé par implémenter un multijoueur fonctionnel, le joueur pouvait alors se connecter au serveur de Photon, créer une salle si elle n'existait pas déjà et rejoindre la salle si elle existait. Une fois la salle rejointe il pouvait déplacer un cube et voir se déplacer les cubes des autres joueurs. Il pouvait ensuite quitter la salle et retourner au menu. Le but de cette implémentation était d'avoir un système fonctionnel sans se soucier de l'esthétique. Par la suite, le système a été complété et amélioré par Yanis (notamment avec l'ajout d'une liste de joueurs dans les "chambres d'attente").

J'ai aussi mis en place un système de permissions qui permet au joueur de jouer en équipe contre d'autres équipes, un joueur ne peut contrôler un personnage que si ce personnage est lié à son équipe. À noter aussi que chaque joueur et chaque équipe ont une couleur attribuée qui s'affiche en surbrillance autour d'un personnage sélectionné. La mise en place du multijoueur m'a forcée à me renseigner sur les fonctions "RPC" de Photon, c'est-à-dire la transmission directe de données entre joueurs.

A ce stade du développement du jeu, le joueur pouvait donc se connecter dans une salle et contrôler des personnages qui appartenaient à son équipe, il pouvait aussi voir d'autres personnages être déplacés mais ne pouvait pas les déplacer, l'accès lui étant refusé par le système de permissions.

Pour la deuxième soutenance, mon rôle dans le multijoueur était d'adapter les systèmes mis en place et décrits dans les parties "Génération de villes" et "Génération du monde". Il existe une limite au nombre d'éléments pouvant être suivis par Photon sur le réseau, cette limite étant fixé à 999. Dépassant cette limite très rapidement (et ce même pour un petit monde), j'ai dû, tout au long de l'implémentation de ces systèmes, créer des alternatives n'utilisant pas d'observateurs Photon.

Contrairement à la première soutenance, les personnages n'avaient plus d'observateurs Photon. Il existe une instance qui a un observateur Photon qui reçoit et envoie sur le réseau les différentes entrées des joueurs par un système de commandes (très similaire à un système de commandes d'un IRC [2]). Par exemple, lorsque un joueur souhaite déplacer un personnage, la commande correspondante ("SetDestination" dans ce cas là) est envoyée localement à l'instance décrite précédemment, cette instance du joueur envoie la commande sur le réseau, cette même instance chez les autres joueurs reçoit cette commande et l'envoie au personnage indiqué par un des paramètres de la commande et exécute la fonction correspondante indiquant au personnage la destination indiquée par les paramètres de la commande.

2. IRC est un sigle qui signifie : Internet Relay Chat, traduisible en français par « discussion relayée par Internet » et est donc un protocole de communication textuelle sur Internet.

Un système de commande similaire à celui décrit ici a aussi été implémenté pour les "Props", ceux-ci reçoivent et envoient aussi des messages à une instance s'occupant de relayer ces messages sur le réseau. Ces systèmes de commandes permettent de se libérer des limites imposées par Photon.

Pour la troisième soutenance, les systèmes utilisés pour le multijoueur étant déjà implémentés aux soutenances précédentes, il a suffit de les utiliser lors de l'implémentation de nouveaux éléments de jeux comme pour l' agriculture ou les portes. Ce fut donc un travail de maintenance simplifié permis par notre avance prise précédemment.

2. Travail de Yanis (suppléant) :

Mon travail sur le multijoueur a commencé avec le menu du multijoueur. Lors de la première soutenance lorsque nous appuyions sur le bouton "Multiplayer" à partir du menu principal alors nous arrivions sur une nouvelle scène où nous pouvions nous connecter aux serveur fournis par Photon (celui-ci permettant d'avoir jusqu'à 20 joueurs connectés simultanément).
À partir de ce moment si le joueur se connectait, alors le jeu allait retenir son nom en l'ajoutant en plus dans les préférences du joueur, c'est-à-dire que lorsque le joueur relancera le jeu, un pseudo lui sera proposé automatiquement, le dernier qu'il aura entré lors de ses parties précédentes. Il avait alors le choix entre soit créer une nouvelle "salle" (nouvelle partie) ou en chercher une déjà existante avec comme nom celui entré par le joueur. Quel que soit son choix (hors celui du retour au menu) il arrivait dans une "salle d'attente" où il pouvait voir les joueurs déjà présents dans la partie ou dans la même "salle d'attente" que lui. Cette liste de joueurs se mettait à jour à chaque fois qu'un joueur arrivait dans cette "salle" ou la quittait.
Pour faire cela on utilisait les différents événements qui sont envoyés par Photon tels que : "OnJoinRoom", "OnPhotonPlayerConnected" et "OnPhotonPlayerDisconnected". Quand ces événements étaient appelés on mettait à jour la liste de joueurs.
Le joueur avait alors de nouveaux trois choix : celui de rejoindre la partie, celui de retourner au choix des différentes "salles" ou bien celui de retourner au menu principal (il se déconnectait alors du serveur Photon).

Après la première soutenance nous avons décidé de reprendre tout notre système de connexion au multijoueur en partant d'une idée de Tanguy :

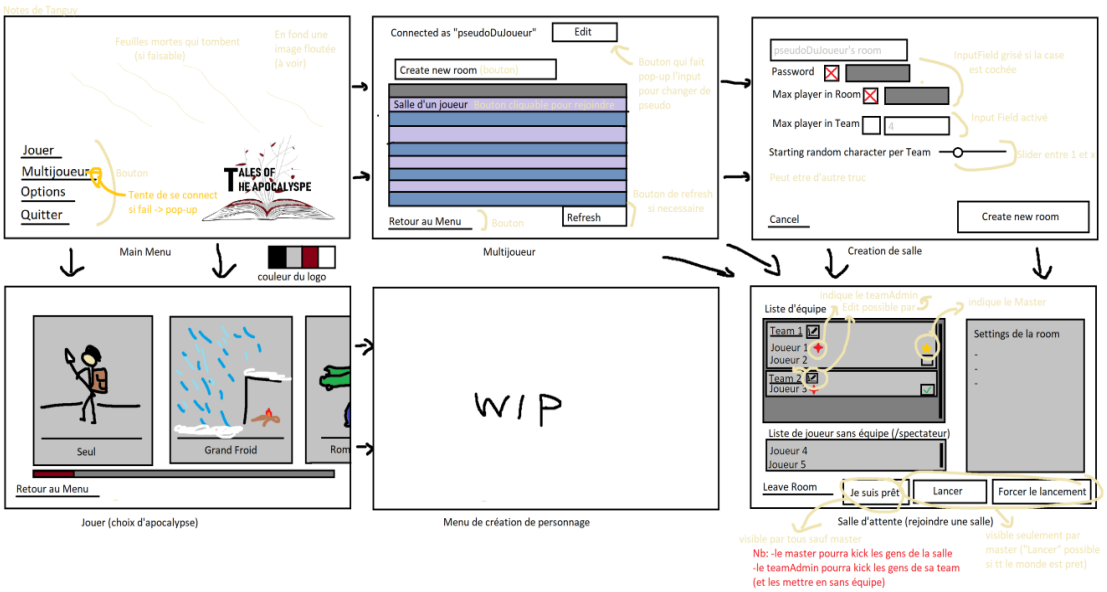

FIGURE 2 – UI

Nous avons donc fait une amélioration pour l'accès plus rapide et efficace à des salles mais aussi pour la gestion des joueurs et de leurs équipes, ainsi que pour le lancement du jeu par le "MasterClient" (celui qui a créé la salle) lorsque tous les joueurs ont déclaré être prêts.
Pour être plus précis le travail s'est divisé en différentes sous-parties :

— La recherche de salles : On utilise désormais les accueils implémentés par Photon et qui nous permettent d'accéder aux listes des salles de chaque accueil. Un tri est effectué sur la liste qu'on reçoit pour que l'on affiche que les salles qui soient accessibles, c'est-à-dire que le nombre de joueurs maximum de la salle n'est pas encore atteint et qu'elles soient considérées comme visibles et ouvertes.

— La création d'une salle : J'ai implémenté la création personnalisée par l'utilisateur de différents paramètres pour la salle à créer. Ces paramètres sont ajoutés à la liste des paramètres de la salle. Ils sont composés de la valeur maximale de joueurs autorisés dans la salle, de la valeur maximale de joueurs dans chaque équipe, du nombre de personnages initialisés dès le début de la partie dans chaque équipe, de la taille de la génération du monde.

— L'attente dans une salle : Pour l'attente dans la salle, soit créée, soit rejointe, j'ai implémenté avec Tanguy tout un système de joueurs et d'équipes avec deux classes "Team" et "Player" qui s'occupent de gérer la création d'équipes et de nouveaux joueurs dans ces différentes équipes ou de les retirer de celles-ci.

— Dans le jeu : Les différents joueurs et équipes sont sauvegardés grâce à la classe "PermissionsManager" qui utilise la fonction "DontDestroyOnLoad" fourni par Unity pour ne pas se faire détruire lors d'un changement de scènes afin de garder les différentes informations sur les joueurs et les équipes créées dans la scène précédente. Ainsi dans le jeu, le système de contrôle des personnages va dépendre de l'équipe à laquelle appartient le joueur. En effet chaque joueur peut contrôler les personnages qu'il a créés et ceux créés par les membres de son équipe.

Pour la troisième soutenance, je me suis occupé de faire un système de communications entre joueurs lors de la partie. Ce système de communication permet donc d'envoyer des messages soit aux joueurs de son équipe, soit à tous les joueurs de la partie en commençant le message par "/all". Ces messages sont transmis à tous les autres joueurs grâce au serveur Photon, récupérés et affichés dans l'espace de communication.

3.6 Menus & Interfaces utilisateurs

1. Travail de Yanis (responsable) :

   Pour la première soutenance :

   (a) Le menu principal :

   J'ai implémenté quatre boutons : un bouton "Solo" qui permet d'accéder au mode hors ligne, individuel du jeu. Suit le bouton "Multiplayer" qui permet d'accéder au mode multijoueur pour pouvoir jouer des parties avec des joueurs en ligne. Nous avons ensuite le bouton "Options" qui nous permet de régler les différents paramètres du jeu tels que le volume sonore, la qualité graphique du jeu ou encore la résolution du jeu c'est-à-dire la taille de l'écran utilisée pour jouer au jeu. Enfin un bouton "Quit" permet de quitter le jeu, de l'arrêter.

   (b) Connexion au Multijoueur :

   De nombreux éléments d'interfaces utilisateurs ont été ajoutés tels que : un bouton pour se connecter au serveur de Photon pour le multijoueur mais il ne marche que lorsque le joueur a rentré un pseudo dans un champ dédié situé juste en-dessous. Après cette étape les deux éléments cités ci-dessus se désactivaient et apparaissaient 3 nouveaux éléments : un autre champ dans lequel le joueur entrait le nom de la "salle" qu'il souhaitait rejoindre si elle existait ou créer sinon. Il pouvait donc cliquer sur deux boutons : "Create" qui allait créer une "salle" avec le nom donné ou "Search" qui allait faire rejoindre la "salle" avec le nom donné si elle existe.

   Le joueur pouvait donc arriver dans une "salle d'attente" où il y avait trois nouveaux éléments : un bouton "Leave Game" pour retourner au choix des "salles", un bouton "Join Game" pour rentrer enfin dans la partie et un élément de défilement qui permettait d'afficher les noms des joueurs présents que l'on pouvait faire défiler de haut en bas.
   Pour cela on regardait chaque joueur présent dans la liste donnée par Photon qui contient les joueurs de la "salle" et on les ajoutait tous dans la liste affichée. Sinon il s'agissait d'une mise à jour de la liste affichée en ajoutant le joueur en tant que dernier ou en regardant l'index du joueur quittant la salle dans la liste pour le retirer de la liste affichée.

   Enfin tout le long de ce processus il y a un bouton "Menu" situé en bas qui permet de revenir au menu principal. Finalement lorsque le joueur rentre dans la partie il y a une barre qui montre le chargement de la partie (il s'agit d'un curseur qui évolue selon le chargement d'une nouvelle scène de façon asynchrone).

(c) L'inventaire :

— Explication visuelle : J'ai implémenté un inventaire pour chaque joueur pour qu'il suive la position de celui-ci. Il y a donc encore une fois un élément de défilement afin de pouvoir faire défiler les différents éléments de l'inventaire. Ces éléments sont constitués d'un élément comprenant une image et un bouton servant d'icône pour l'objet, d'un bouton pour enlever l'objet de l'inventaire et d'un texte qui permet de voir le nombre d'objets dans un emplacement de l'inventaire. Si le nombre d'objets par emplacement dépasse celui autorisé, alors l'objet sera ajouté dans un nouvel emplacement et ainsi de suite jusqu'à arriver au maximum de capacité de l'inventaire. L'inventaire de chaque joueur sélectionné est affiché lorsque le joueur appuie sur la touche E et disparaît s'il appuie de nouveau sur cette touche .
Ces objets, stockés dans l'inventaire sont utilisables selon leurs fonctionnalités. Par exemple une pomme pourra être mangée. Donc lorsque le joueur cliquera sur la pomme dans l'inventaire elle sera enlevée de l'inventaire s'il n'en contenait qu'une seule ou décrémentée sinon et la faim du joueur diminuera, les informations du joueur étant mis à jour.

— Explication fonctionnelle : Pour effectuer l'ajout dans l'inventaire d'un objet, pour cette première soutenance nous avions deux manières de le faire. Soit nous pouvions interagir directement avec l'objet selon une certaine de zone de rencontre définie par un rayon autour de cet objet. L'objet possédait un script dérivant d'une classe "Interactable" pour que ce dernier puisse avoir un objet différent et appelle alors les différentes fonctions pour ajouter l'objet dans l'inventaire. L'autre manière de faire était une manière pour tester l'inventaire. Le joueur pouvait appuyer sur la touche I et un objet au hasard était choisi et ajouté dans l'inventaire parmi une liste d'objets.
Dans les deux cas, l'appel de l'ajout à l'inventaire vérifiait dans un dictionnaire si l'objet pouvait être ajouté, c'est-à-dire si l'inventaire n'était pas déjà plein, sinon l'objet soit était ajouté dans le dictionnaire s'il n'en faisait pas partie, soit il incrémentait sa valeur associée. Si l'objet avait pu être ajouté correctement au dictionnaire, l'inventaire était mis à jour par des fonctions qui géraient les différents éléments de chaque emplacement. On affichait chaque emplacement et permettait l'interaction avec celui-ci en gérant son incrémentation ou sa décrémentation.

<u>Pour la seconde soutenance :</u>

— Dans le menu : Si l'utilisateur clique sur "Solo" alors il n'arrive plus directement sur la prochaine scène mais il passe d'abord par un choix. J'ai implémenté différents "panels" qui contiennent les images de chaque mode de jeu disponible, les modes de jeu correspondants aux événements qui se dérouleront durant la partie (ex : un mode seul, un mode grand froid, etc). Chaque "panel" est donc composé d'un bouton afin de choisir le mode de jeu. Après avoir appuyé sur le bouton d'un des "panels", on charge la prochaine scène en solo et on garde l'information du mode de jeu choisi

grâce à la classe "Mode" sur laquelle on utilise aussi la fonction "DontDestroyOnLoad".

— Dans le multijoueur : Tout d'abord j'ai ajouté un composant de texte en haut lors de la recherche de salle qui affiche le pseudo avec lequel le joueur s'est connecté au serveur Photon. A coté j'ai implémenté la possibilité de modifier ce pseudo. Il y a donc un bouton "Edit" qui permet de faire apparaître un champ dans lequel le joueur peut taper un nouveau pseudo et appuyer sur le bouton "Valid". Ce nouveau pseudo remplace alors l'ancien, est affiché et est sauvegardé pour le reste de la partie et sera gardé en mémoire lors de prochains lancements du jeu.

Il y a ensuite le bouton pour créer une nouvelle salle et la liste des salles qui se trouve dans un composant de défilement et qui donc permet de faire défiler les différentes salles si elles existent. Les salles sont ajoutées les unes en-dessous des autres dans ce composant de défilement qui est mis à jour à chaque fois qu'une salle est créée, fermée ou détruite. Chaque salle affichée possède un composant bouton qui permet de la rejoindre. Si la salle est déclarée comme privée, il faut alors rentrer un mot de passe. Un "panel" s'affiche alors dans lequel se trouve un champ où l'utilisateur doit rentrer le mot de passe de la salle. Si le texte rentré dans le champ correspond bien au mot de passe, le joueur peut alors rejoindre la salle.

Si on décide de créer une salle, on arrive dans la partie de création des paramètres de la salle. Il y a alors un champ où renseigner le nom que l'on souhaite assigner à la salle. Viennent ensuite les différents paramètres optionnels tels que la possibilité d'un mot de passe, la valeur maximale de joueurs dans la salle, la valeur maximale de joueurs dans chaque équipe et la taille souhaitée pour la génération du monde. Chaque paramètre possède plusieurs composants : un interrupteur (un "toggle") qui permet d'activer ou de désactiver le paramètre et un champ où l'utilisateur associe la valeur souhaitée au paramètre si celui-ci est activé par le "toogle". Un dernier paramètre est présent, il s'agit du choix du nombre de personnages qui vont apparaître dès que le chef d'une équipe va rentrer dans la partie. Ce paramètre est associé à un curseur qui permet donc de choisir entre 0 et 5 le nombre de personnages à faire apparaître.

Lorsque l'utilisateur appuiera sur le bouton "Create Room" il va alors créer une salle en lui ajoutant donc les différents paramètres personnalisés. Si un paramètre est désactivé ou n'a pas de valeur associée dans le champ, une valeur par défaut lui sera attribuée lors de la création de la salle. Il peut sinon appuyer sur le bouton "Cancel" qui le ramènera à la recherche de salles.

Si le joueur a créé une salle ou en rejoint une, il arrive alors dans la salle d'attente où il peut choisir de rester sans équipe, d'en créer une nouvelle ou d'en rejoindre une déjà existante. Pour ce faire j'ai implémenté deux composants de défilement : un qui est la liste des joueurs sans équipes, l'autre correspondant à la liste des équipes et des joueurs présents dans ces équipes.

Pour créer une équipe il y a un un champ dans lequel on peut mettre le nom de l'équipe à créer et un bouton pour procéder à sa création. Une équipe peut être créée s'il n'existe aucune autre équipe avec le même nom. La liste des équipes est alors mise à jour pour tous les joueurs avec celle nouvellement créée. Pour rejoindre une équipe il suffit simplement de cliquer sur le nom de l'équipe dans la liste. Si un joueur sans équipe en rejoint une, la liste des joueurs sans équipes sera alors mise à jour sans ce dernier. S'il était déjà dans une équipe le joueur peut en créer une nouvelle ou en rejoindre une. L'affichage sera alors mis à jour en mettant le joueur dans sa nouvelle équipe. De plus si l'équipe qu'il vient de quitter ne contenait pas d'autres joueurs elle sera effacée de la liste des équipes dans l'affichage.

Chaque joueur a la possibilité de montrer qu'il est prêt à lancer la partie. Un bouton "Ready" est en effet présent en bas et a pour action d'afficher à côté du nom du joueur une image montrant qu'il est prêt et cette image sera visible par tous les joueurs. On peut aussi voir un onglet dans lequel sont affichées les différentes informations par rapport aux paramètres de la salle.

Il existe aussi des fonctionnalités disponibles uniquement pour certains joueurs. Le joueur ayant créé une équipe possède le droit de détruire son équipe et/ou de retirer des joueurs de l'équipe via un bouton rouge affiché à droite du nom de l'équipe et/ou du joueur. Ce bouton n'est donc visible que pour le joueur en ayant les permissions. Le joueur ayant créé la salle a quant à lui les mêmes permissions que celles citées précédemment mais il peut le faire de manière générale, c'est-à-dire qu'il peut détruire n'importe quelle équipe, retirer n'importe quel joueur d'une équipe et même retirer un joueur de la salle s'il n'avait pas d'équipe. Une mise à jour de l'affichage des équipes et des joueurs sans équipes est à chaque action de ces utilisateurs effectuée. De plus, ce joueur possède deux boutons de plus que les autres : un bouton "Launch" qui permet de lancer la partie si tous les joueurs sont prêts (et ont donc l'image associée affichée) et un bouton "Force Launch" qui lance la partie même si certains ne sont toujours pas prêts.

— Dans le jeu : J'ai implémenté de nouvelles interfaces utilisateurs telles qu'un menu de pause (le jeu continue quand même en arrière-plan) qui nous permet soit de retourner dans le jeu, soit de gérer les options du jeu, soit de retourner au menu ou bien de quitter complètement le jeu. Ensuite j'ai repris la boîte d'informations qui avait été faite par Tanguy et j'en ai modifié l'affichage. Cette boîte d'informations possède aussi un bouton qui permet d'afficher plus d'informations sur les différentes caractéristiques et capacités du personnages.

J'ai aussi ajouté un espace en haut de l'écran qui affiche les références de chaque personnage que nous pouvons contrôler. Pour le moment il s'agit d'images avec le nom du personnage et possèdent un composant bouton qui permet de sélectionner et de contrôler un joueur et de ramener la caméra au-dessus de celui-ci. Ensuite le système d'affichage d'inventaires a été complètement refait : j'ai implémenté à droite un composant de défilement dans lequel sont ajoutés les différents inventaires

des meubles et interfaces des personnages dès lors qu'ils sont sélectionnés par l'utilisateur. Pour afficher cette liste d'interfaces il suffit d'appuyer la touche E si au moins un personnage est sélectionné.

Il existe deux types d'interfaces : tout d'abord l'interface du joueur composée de différents onglets cliquables et donc répartie en différentes parties : il y a en premier lieu l'inventaire du personnage sélectionné dans lequel on peut voir les différents objets qu'il possède et avec lesquels on peut interagir comme à la soutenance précédente avec l'ajout en plus d'un bouton qui affiche la description de l'objet souhaité.

Le deuxième onglet représente les équipements du joueur. Lorsque le joueur possède des objets dans son inventaire qui sont de types "Equipable" ou "Wearable" et qu'il clique dessus ils seront enlevés de l'inventaire et mis dans cette onglet selon les parties du corps auxquels ils doivent être mis, dans les emplacements adéquats).

Le troisième onglet est le système d'artisanat où le joueur pouvait choisir de fusionner deux objets pour en former un nouveau. Pour cela il devait posséder le bon nombre d'objets pour faire le nouvel objet. Si c'était le cas, la case correspondant au nouvel objet possédait un carré vert. Il y avait des conditions pour pouvoir procéder à la création de nouveaux d'objets. Si le joueur ne remplissait pas les conditions pour certains, l'objet récupérable était grisé et le joueur ne pouvait pas interagir avec.

Enfin le dernier onglet est l'onglet santé dans lequel sont affichées les différentes statistiques de santé du joueur ainsi que ses blessures.

Le deuxième type d'interface que l'on peut retrouver dans la liste des interfaces sont les inventaires des meubles par exemple. Une interaction est alors possible entre l'inventaire d'un meuble et les interfaces des personnages sélectionnés. Lorsque le joueur va cliquer sur l'objet présent dans l'inventaire du meuble cela va alors le supprimer de celui-ci et l'ajouter dans l'inventaire du premier personnage sélectionné.

Finalement, j'ai implémenté une barre de construction dans laquelle se trouve différents boutons qui permettent de lancer le mode construction implémenté par Tanguy avec le bon élément à construire.

Pour la troisième soutenance :

Pour cette dernière soutenance le but principal au niveau des menus et interfaces utilisateurs était de peaufiner le rendu visuel pour avoir quelque chose de plus attirant visuellement.

Je me suis donc occupé d'améliorer les polices d'écritures, les couleurs et les formes des différents éléments de menus ou d'interfaces utilisateurs.

J'ai aussi implémenté de nouveaux éléments de menus comme les options qui sont maintenant disponibles à partir du menu de pause dans lequel on peut ainsi attribuer des touches pour différentes actions, par exemple pour se déplacer, pour faire apparaître les interfaces du joueur ou le mode construction. Si le joueur est dans le mode "Solo", alors avant de lancer la partie il aura la possibilité d'appuyer sur un bouton pour charger une sauvegarde antérieur et au moment de quitter la partie il aura la possibilité entre deux boutons, un pour sauvegarder sa partie et un pour quitter complètement le jeu.

Du côté des interfaces utilisateurs j'ai amélioré le système d'interaction du personnage avec d'autres personnages ou avec des objets. Maintenant lorsque le joueur clique, en ayant au moins un personnage sélectionné, sur un objet ou un autre personnage, une liste d'actions sous formes de boutons apparaît au-dessus de la tête du personnage. Ensuite avec l'implémentation d'un canal de communication j'ai ajouté une boîte de dialogue dans laquelle se trouve un champ pour écrire le message, un bouton pour envoyer le message et un champ dans lequel tous les messages sont affichés.

De plus, j'ai ajouté un guide d'apprentissage dans le mode "Solo" qui donc fait apparaître des messages à l'écran au moment où certaines actions sont faites avec des messages adaptés à ces moments.

2. Travail de Tanguy (suppléant) :

Afin de commencer à différencier différents personnages, j'ai implémenté lors de la première soutenance un système d'informations du personnage. En haut à gauche de l'écran on va avoir un cadran d'informations avec le nom du personnage et ses différents attributs que sont : la Force, l'Intelligence, la Perception, le Mental, le Social qui vont de 0 à 100, une barre de faim temporaire a aussi été mise pour tester les interactions avec l'inventaire. Cette boîte d'informations apparaît lorsqu'un personnage est sélectionné. Si plusieurs personnages sont sélectionnés, c'est la boîte d'informations du dernier personnage sélectionné qui apparaîtra.

De nombreux facteurs vont faire évoluer un personnage. Le seul mis en place pour le moment est l'inventaire. Certains objets sont utilisable dans l'inventaire, par exemple le joueur peut cliquer sur un emplacement représentant de la nourriture pour modifier la faim d'un personnage, le cadran d'informations sera alors mis à jour.

Pour la seconde soutenance mon travail fut de conceptualiser les interfaces et les fonctionnalités du menu, de la création de salle, de la salle d'attente ainsi que les interfaces des personnages (inventaire, équipement, artisanat et santé). Ces interfaces et fonctionnalités ont été implémentées et décrites par Yanis dans ses parties : dans "Multijoueur" et "Menus & Interfaces utilisateurs".

Pour cette soutenance finale, j'ai amélioré la visibilité des actions dans le menu d'actions décrit par Yanis dans la partie précédente. J'ai donc rajouté des noms d'actions dynamiques directement influencées par la spécificité d'une action par exemple le nom de l'action de l'attaque d'un personnage avec un marteau n'est plus "Attack Melee 1" (Attaque de mêlée 1) mais "Hit with SledgeHammer" (Frapper avec un marteau), le nom s'adapte ici en fonction du type de dégât de l'arme et son nom. L'apparition des actions dans la liste est aussi dynamique, prenons en exemple l'action de mettre un bandage. Si le personnage ne saigne pas, l'action n'apparaît pas dans la liste d'action, sinon l'action apparaît mais est grisée si le personnage n'a pas de bandage. Ces changements permettent d'augmenter la visibilité du joueur vis-à-vis de la liste des actions, bien que petits ils font partie de l'opération de polissage effectuée pour la soutenance finale.

3.7 Textures, Modèles & Animations

Travail de Marc-Emmanuel (responsable) :

Pour la première soutenance :

1. Modèles :

   Comme prévu tous les modèles faits pour le jeu l'ont été en voxel. Le travail de modélisation a commencé avec des essais d'importation sous Unity avec le premier modèle fait pour le jeu, un réfrigérateur . Ce modèle a été suivi par beaucoup d'autres qui avaient pour but de supporter la génération procédurale et le "pathfinding". Ainsi ont suivi des modèles de routes et de structures de bâtiments . Au vue de l'avancement prévu, la priorité était de trouver une génération viable avant de se lancer dans la conception globale. Une fois la génération finie j'ai pu me lancer dans la modélisation de bâtiments de différentes tailles et formes ainsi que d'objets qui seront interactifs.

2. Animation :

   J'ai débuté le travail d'animation entre la conception des premiers modèles de test et le début de conception des modèles finaux. Il a fallu dans un premier temps créer nos personnages, premières cibles de notre animation. Initialement les modèles pensés pour les personnages devaient se faire sans membres animés mais ont vite évolués pour profiter des options que propose Unity. Ainsi le premier modèle en "T-POSE", posture standard pour l'animation où le personnage est debout, les bras levés sur le coté (comme un T), a vu le jour. Il a fallu le munir d'un squelette puis j'ai pu utiliser la banque d'animation humanoïde de Mixamo pour tester les animations. Après quelques essais infructueux avec différents modèles arrivés à ce stade j'ai enfin pu trouver le style de modèles qui ont le meilleur rendu en animation. Il a ensuite simplement fallu me constituer une banque d'animation pour commencer à travailler sous Unity.

Pour la seconde soutenance :

1. Textures :

   - Le premier travail sur les textures du jeu entre la première et la seconde soutenance a été de faire les images présentes dans les menus. L'idée était d'avoir un environnement rappelant le contexte apocalyptique du jeu sans pour autant que l'endroit soit reconnaissable. En conséquence ces images sont floutées et c'est la raison pour laquelle des images réalistes et non du pixel art ou des scènes voxel ont été choisies, les deux dernières options essayées collant plus avec le style général voulu mais rendant beaucoup moins bien.

- La seconde tâche, plus majeure, a été de commencer à faire toutes les icônes dont nous nous servons dans le jeu. Déjà les icônes d'interface pour le menu par exemple les boutons de validation et d'exclusion du menu multijoueur mais aussi toutes les icônes d'objet visibles dans les inventaires du jeu, on trouve en vrac les armes, les meubles et les composants de construction qui ont chacun besoin de leur texture facilement identifiable.

2. Modèles :

   Le travail sur les modèles du jeu s'est fait dans la continuité de ce qui avait été fait pour la première soutenance mais à un rythme bien plus soutenu.
   Au niveau modèles nécessaires à la conception des villes j'ai jusqu'à présent pu terminer onze bâtiments différents. Pour chacun d'entre eux j'ai d'abord fait un modèle générique avec le découpage des pièces, des escaliers et de quelques éléments dont l'interaction n'est pas possible dans le jeu final comme des sonnettes à l'entrée des immeubles qui seraient hors service dans le cadre d'une apocalypse. J'ai ensuite fait pour chaque plusieurs versions des différents étages des bâtiments, ajoutant des détails comme de la salissure ou encore des effondrements de façade. Cette partie de la conception a pour but de ne pas avoir un environnement trop redondant. La plupart de ces modèles sont déjà ajoutés au jeu mais une partie est simplement encore en conception en attente de formes détaillées alternatives.

   Pour ce qui est des modèles de meubles et autres éléments qui constituent la génération des pièces mais aussi des rues avec des poubelles ou encore des 'espaces de nature' avec une végétation variée le travail a là aussi suivit son chemin. J'ai pu finaliser beaucoup de modèles (lit, établis, poubelles, meubles de stockage en tout genre, éléments plus spécifiques à certains bâtiments comme des rayons pour les espaces de vente) qui ne sont pas encore présents dans le jeu car il faudra d'abord définir toutes leurs positions possibles, action indispensable pour la génération de ville. La raison pour laquelle cela n'est pas encore fait a été une question de temps mais c'est surtout qu'il sera préférable de le faire une fois tous les modèles en main pour éviter de revenir sur la position de la génération pour ajouter de nouvelles choses et tout faire efficacement en une fois.

3. Animations :

   L'animation n'a pas beaucoup évoluée entre cette première et seconde soutenance, nos objectifs initiaux ayant déjà été remplis dès la première. Il y a néanmoins eu un travail de recherche sur l'animation de plusieurs modèles assemblés entre eux et aussi le début d'une constitution d'une banque d'animation personnalisée pour les zombies qui doivent avoir leurs propres manières de se déplacer.

4. Résumé :

   Les textures et les modèles étaient en bon avancement, les avancements étant surtout théoriques et le travail de la première soutenance étant voué à être réutilisé.

Pour la troisième soutenance :

1. Textures :

   - Pour finaliser le projet qu'est TotA au niveau des textures il a d'abord fallu finir toutes les icônes représentant les différents objets trouvables en jeu, le style cubique des ces icônes a été maintenu et on peut maintenant facilement identifier les armes, les objets nécessaires à la construction ou encore les meubles constructibles.

   - La seconde tâche nécessaire pour finir les textures du jeu a été de faire le rendu visuel des menus à savoir les boutons et zones de sélection étant donné que le fond avait déjà été fait et des interfaces trouvables en jeu comme l'inventaire, le menu de construction et les zones où l'on trouve les statistiques des personnages. C'est un des seuls éléments du jeu qui ne suis pas complètement le style cubique voulu mais là encore c'est justifié par le besoin de clarté dans les interfaces et surtout la volonté de faire une distinction claire entre le jeu en lui-même et les interfaces.

2. Modèles :

   Le travail sur les modèles du jeu s'est fait dans la continuité de ce qui avait été fait pour la première et la deuxième soutenance, il a fallu garder le même rythme de travail pour finir ce qui avait été déjà bien entamé.

   Au niveau modèles nécessaires à la conception des villes j'ai pu terminer les quinze bâtiments différents voulus pour le jeu. Ce nombre n'est cependant pas très représentatif étant donné que chaque chose désignée comme bâtiment n'est en fait qu'un modèle générique avec le découpage des pièces, des escaliers et de quelques éléments dont l'interaction n'est pas possible dans le jeu final comme des sonnettes à l'entrée des immeubles qui seraient hors service dans le cadre d'une apocalypse. le vrai nombre de bâtiments en jeu, si l'on ne compte pas les différents meubles qu'ils peuvent abriter approche plus de la cinquantaine étant donné que différentes versions détaillées, trois en général, de chaque étage de chaque bâtiment créé existent et permettent une diversité dans le jeu. Les détails n'ont pas été modifiés en vue de la dernière soutenance et les bâtiments se démarquent souvent par l'usure de leurs pièces et leur destruction partielle voir totale. J'ai ensuite fait pour chaque plusieurs versions des différents étages des bâtiments, ajoutant des détails comme de la salissure ou encore des effondrements de façade.

   Enfin la liste des modèles de meubles et autres éléments avec lesquels on peut interagir voulue pour le jeu a été terminée, ce qui correspond à tous les meubles de vie classiques que l'on peut trouver dans une maison comme les tables, chaises ou lit et ainsi de suite mais aussi nos établis, des objets de lieux publics comme des bancs, des voitures ou des poubelles et enfin les objets qui servent au système électrique du jeu. Le placement de ces objets en jeu dans les différents bâtiments a lui aussi été fait, suivant la méthode

programmée pour la deuxième soutenance. Ainsi ceux-ci apparaissent à des endroits prédéfinis avec une certaine probabilité ce qui ajoute encore de la diversité à l'univers de jeu.

3. Animations :

   - Aspect visuel majeur du jeu, l'animation des personnages et des objets a elle aussi été finalisée, dorénavant on peut par exemple ouvrir les portes et les personnages avancent et agissent selon nos propres animations, entre autre pour la marche et l'orientation ou encore la mort de ceux-ci.

3.8 Autres fonctionnalités jouables

1. Travail de Tanguy :

   Pour la première soutenance :

   Une fois la génération de ville terminée lors de la première soutenance, j'ai implémenté le déplacement de la caméra du joueur dans le monde. Dans TotA le joueur voit le monde en vue de dessus, le déplacement horizontal de la caméra se fait avec les touches directionnelles ou les touches WASD. Le déplacement vertical de la caméra dans le monde a été plus difficile à mettre en place, en effet le joueur doit pouvoir voir l'intérieur des bâtiments à travers le sol et le toit, pour ce faire, le joueur utilise la molette de la souris pour définir l'étage qu'il veut voir, à chaque fois qu'un nouvel étage est défini tous les étages supérieurs à celui choisi ne sont pas montrés à l'écran (évidemment cette disparition des bâtiments se fait localement et n'est pas transmise aux autres joueurs).

   Pour la deuxième soutenance :

   (a) Système de traits :

   Comme annoncé dans le cahier des charges, nous souhaitons donner une identité au personnages. Cet identité se fait principalement par le système de traits. Lors de la création d'un personnage, des traits lui sont attribués, ces traits modifient les caractéristiques du personnage, par exemple le trait "Strong" (fort) donne +10 à la force du personnage. Il existe plusieurs catégories de traits : physique, mental, professionnel et expérience apocalyptique. Les traits physiques regroupent l'ensemble des traits caractérisants le corps du personnage, les traits mentaux regroupent l'ensemble des traits concernant la condition mentale, les compétences sociales et les connaissances d'un personnage, les traits professionnels regroupent l'ensemble des traits concernant l'expérience professionnel acquise avant le début de l'apocalypse, enfin, les traits d'expérience apocalyptique regroupent l'ensemble des traits concernant l'expérience pendant l'apocalypse. Ce système de trait permettra au joueur de s'attacher plus facilement à ses personnages, ainsi que rendre ces personnages plus mémorables.

   (b) Système de santé et de combat :

   Pour une expérience de jeu plus intéressante, nous avons implémenté un système de santé relativement complexe. Chaque personnage à plusieurs parties du corps avec des spécificités différentes(tête, jambes, torses, bras, mains etc). Chaque partie du corps peut avoir des blessures. Chaque blessure pouvant avoir des effets différents (saignement, infection, fracture, etc).

Le système de combat existe en parallèle du système de santé. Chaque personnage peut s'équiper d'armes et d'armures. Les armes ont des caractéristiques nombreuses : à distance ou non, tranchante, dégât de l'arme, vitesse d'attaque, etc. Les armures en ont aussi : résistance au coupure, résistance à la température, etc.

Pour illustrer ces deux systèmes prenons un exemple, un personnage en attaque un autre avec un sabre, s'il réussit le personnage est coupé à la jambe, la blessure va saigner, il risque de mourir si son stock de sang se vide. Aussi, comme c'est la jambe qui a été touchée, sa vitesse de déplacement change grandement. De plus, si la coupure est trop importante la jambe est entièrement coupée, le personnage perd alors sa jambe et son pied. Enfin, la douleur provoquée par la blessure affecte les capacités du personnages et peut le faire s'évanouir.

(c) Système d'artisanat et de construction :

Les personnages peuvent créer des nouveaux objets à partir de leurs objets via l'onglet artisanat de leur inventaire. Il existe des recettes permettant d'établir quelles combinaisons d'objets donnent quels objets. De plus, les joueurs peuvent placer des plans de construction, les personnages pourront finir la construction de cette construction en utilisant des objets.

Pour la troisième soutenance :

(a) Temps d'action :

Contrairement à la soutenance précédente, les actions prennent maintenant souvent du temps pour se faire. Ce temps varie en fonction de l'action et du personnage qui la réalise. Par exemple, tirer avec une arme dépend des caractéristiques de l'arme et de la compétence "Marksman" (tireur) du personnage qui tire. Yanis s'est occupé de l'implémentation d'une représentation visuelle de ce temps d'action, représenté par un cercle qui se remplit au fil de l'action (inspiré par This War of Mine, voir cahier des charges). Ce système de temps d'action est vital au rythme du jeu, les actions ne sont pas instantanées, les joueurs investissent du temps pour progresser, ce système permet aussi de donner de l'importance à certaines actions (ex : un tir de sniper est un investissement important de temps, ce temps investi reflète ainsi la puissance de l'arme).

(b) Système de progression :

Avec ce système de temps d'action vient le système de progression. Les compétences des personnages peuvent progresser au fil des expériences. Certaines actions sont liées à une compétence, par exemple "Tirer" comme vu plus haut est lié à la compétence "Marksman" (tireur), d'autre non, par exemple ouvrir une porte. Les actions liées à des compétences permettent au personnage qui effectue l'action de

gagner de l'expérience dans cette compétence. L'expérience gagné est le temps passé à réaliser l'action, l'expérience requise pour monter de niveau de compétence est influencé par le niveau actuel de la compétence (plus ce niveau est élevé plus il faudra d'expérience pour passer au prochain niveau). Pour visualiser ce système prenons en exemple la fouille d'un meuble par un personnage. L'action "Fouiller" utilise la compétence "Scavenger" du personnage, l'action "Fouiller" sur une commode de base prend 5 secondes mais comme dit précédemment ce temps peut varier avec le niveau de "Scavenger" du personnage. Une fois l'action réalisée, le personnage va gagner 5 points d'expérience de "Scavenger", si son niveau augmente, les prochaines fouilles dureront moins de temps (mais rapporteront moins d'expérience). A noter que dans ce cas-là, une fois la commode fouillée, l'action pour l'ouvrir n'utilise aucune compétence et ne prend que 0.5 seconde.

(c) Système de santé et température :

Comme annoncé à la soutenance précédente, le système de santé a été complété par les notions de Soin, de Température et de Maladie. Les personnages peuvent soigner les saignements avec des bandages, ils peuvent aussi s'amputer les parties du corps infectées par une morsure de zombie. Ces actions utilisent la compétence "Doctor" (médecin) des personnages. Le système de température a aussi été implémenté, la température dépend du biome dans lequel se trouve le personnage, de la saison et de la présence de source de chaleur proche (feu de camp). Si la température environnant descend en dessous d'un seuil, ce seuil pouvant être augmenté par certains vêtements, le personnage subit des gelures ("Frostbite") aussi il a plus de chance de tomber malade. De même, lors de température trop élevée le personnage peut subir une insolation et risque de perdre connaissance.

(d) Artisanat :

Le système d'artisanat a été amélioré avec l'ajout des établis. Certaines recettes requièrent aux personnages d'interagir avec un certain type d'établi. Il existe pour l'instant 3 établis distincts, l'établi de bois, de métal et d'électronique. Certaines recettes demandent aussi un certain niveau de compétence. Finalement, chaque personnage a des connaissances qui lui permet d'utiliser certaines recettes, il peut étendre ses connaissances en lisant certains livres.

(e) Système de construction, de destruction et de recyclage :

Le système de construction est terminé. Le joueur peut poser des plans, les personnages iront alors déposer les objets nécessaires à la construction. Pour l'instant, les personnages peuvent construire des chaises, des tables, des lits, des établis, des feux de camp et des caissons de stockage. La plupart des meubles sont recyclables, c'est-à-dire qu'ils peuvent être démonté pour redonner au personnages certains objets qui les composent. D'autres constructions ne sont pas démontables, les murs et les portes sont plus résistants, ils possèdent une barre de vie et ne sont détruits que

si cette barre atteint 0. Ces constructions sont utilisés par le joueur pour construire sa base ou fortifier un endroit, il peut alors s'approprier le monde qui l'entoure.

(f) Porte :

Les portes peuvent être construites par les personnages et sont présentes naturellement dans le monde. Elles peuvent s'ouvrir des deux côtés et s'ouvrent automatiquement quand un personnage court à travers. Elles peuvent être cadenassées, alors déverrouillables seulement par les personnages d'une même équipe. Comme les murs, les portes sont résistantes et destructibles. La porte fut très intéressante à implémenter, la problématique étant comment compenser la taille fine (vu de haut) de la porte pour que le joueur puisse cliquer sur elle. Premièrement, la zone de détection de clique est plus large quand la porte est fermée. Ensuite, si le joueur clique sur un mur au-dessus d'une porte, la porte est sélectionnée. Finalement, l'ouverture automatique des portes permettent de réduire le nombre d'interaction avec les portes.

(g) Agriculture :

Les personnages peuvent planter des graines dans des pots. Les plantes ont différent stades permettant de visualiser leur croissances. A n'importe quelle stade la plante peut être détruite, utile pour saccager une base adverse. Quand la plante est mature, elle peut être récoltée. Pour l'instant nous avons seulement implémenté un plant de tomate comme indiqué dans le cahier des charges, mais le système est suffisamment flexible pour ajouter de nombreuses nouvelles plantes.

(h) Autre :

Les personnages peuvent suivre d'autre personnage, cela permet une meilleure gestion d'un grand nombre de personnage.

2. Travail de Yanis :

Pour la troisième soutenance, je me suis occupé de toute la partie "Solo" du jeu : j'ai donc du intégrer tous les éléments mis en oeuvre dans le multijoueur mais sans utiliser de réseau. J'ai donc adapter les différentes fonctions mises en places utilisant du réseau pour pouvoir se jouer de façon hors-ligne.

J'ai aussi implémenté un système de sauvegarde dans le mode "Solo". Le joueur peut donc décider au moment de choisir le mode de jeu de reprendre au niveau d'une ancienne sauvegarde ou de repartir de zéro et lorsqu'il décide de quitter le jeu il peut choisir de sauvegarder sa partie ou bien de quitter complètement. Ce système de sauvegarde garde en mémoire toutes les caractéristiques des différents personnages contrôlés par le joueur au moment où celui-ci a sauvegardé la partie. Ses caractéristiques sont : son

nom, ses traits de caractères, ses compétences et statistiques mais aussi son inventaire, ses équipements et sa santé.

Encore dans le mode "Solo" j'ai mis en place un petit guide d'apprentissage pour le joueur qui au lancement du jeu pourra voir des messages lui indiquant les principales fonctionnalités qui sont disponibles dans le jeu et les manières de les effectuer. Il aura ainsi la possibilité de comprendre comment rentrer dans le jeu, se déplacer, zoomer, faire apparaître des personnages, etc.

3. Travail de Jordan :

M'occupant déjà des sons et de l'artificielle intelligence, j'ai implémenté pour la troisième soutenance un système de visualisation des sons, de plus les zombies sont alertés par le son. Lorsqu'un son est détecté par un zombie, le zombie retient le niveau sonore du son qu'il a entendu, le joueur peut alors détourner son attention en créant du bruit avec un niveau sonore plus élevé que celui que le zombie a entendu (ou en apparaissant dans son champ de vision). Les sons passent à travers les murs, ils sont donc un élément primordial de jeu lorsque les zombies sont présents.

J'ai aussi rajouté un système de saison et de météo. Il peut faire beau, pleuvoir et même neiger. Lorsqu'il a neigé le sol se recouvre de blanc, simulant une couche de neige qui se serait déposée. La pluie et la neige utilise le système de particule d'Unity, j'ai du alors adapté ce système et créé le système qui décide de la météo. La neige au sol fut plus compliquée à implémenté car n'utilisant aucune fonctionnalité d'Unity vu jusque là, le rendu est tout de même très acceptable.

3.9 Son & Musique

Travail de Jordan :

Pour cette deuxième soutenance nous voulions implémenter les sons et les musiques que nous avions trouvé pendant la première soutenance pour donner un peu d'ambiance, ce qui est important dans notre jeux. Je me suis donc renseigné sur le fonctionnement de l'audio sur Unity qui est plutôt complet : les sons dans un environnement 3D, le mixage des sons et musiques, les sources de sons et le receveur. Pour gérer tout ça nous avons un "AudioManager" qui, comme son nom l'indique, gère toutes les sources audio. On y stock tous les sons et toutes les musiques et on peut y régler leur nom, leur volume, choisir si elles se répètent, etc. On peut ainsi faire jouer n'importe quel son ou musique n'importe quand et n'importe où dans un script en une ligne de commande. Pour l'instant, nous avons implémenté un cycle de musique en lien avec le cycle jour/nuit. La musique change lorsque la nuit tombe et revient au levée du soleil. Nous entendons aussi un son d'alerte lorsqu'un zombie nous repère. Pour éviter que le son se répète tant que nous sommes dans la vision du zombie, nous créons un nouvel objet quand le son a été joué pour montrer qu'il a déjà été joué une fois puis on le détruit si le joueur sort de la vision du zombie.

La majorité du travail avait déjà été faite en amont, nous avons récolté tous les sons et musiques nécessaires au jeu pour la deuxième soutenance. Pour cette soutenance finale, j'ai alors associé chaque action à un son et géré le mixage global en jeu.

- bruit de porte
- bruit de coffre
- bruit de coup
- bruit d'arme à feu
- bruit de construction
- bruit de zombie
- bruit de flamme

3.10 Site Web

1. Travail de Yanis :

   Pour la première soutenance, je me suis occupé de générer les différentes pages HTML :

   - Une page d'accueil avec un menu pour accéder aux autres pages, une brève introduction au projet et probablement un aperçu du jeu *Tales Of The Apocalypse*.
   - Une page de présentation du projet, de son origine et de son but.
   - Une page pour présenter chaque membre du groupe.
   - Une page concernant l'avancement du projet et la répartition des tâches.
   - Une page relatant les différents problèmes rencontrés et les solutions envisagées lors de ce projet.
   - Une page de téléchargements où il sera possible de télécharger le jeu mais aussi une version du rapport de soutenance finale et une version lite du projet.

— Une page d'annexes où seront données les références aux logiciels, images, sons, bibliothèques, applets et autres éléments que nous aurions pu utiliser.

Je me suis de plus occupé de faire l'hébergement du site grâce GitHub Pages avec le nom de domaine pris grâce à namecheap.com, pour que l'on puisse accéder au site par un lien ayant un nom nous représentant, nous Filiga.

Pour la seconde soutenance, je me suis occupé de la maintenance du site, c'est-à-dire que je me suis occupé de continuer à ajouter du contenu au site par rapport à ce que nous avons fait et ajouté à notre jeu avec l'ajout de nouvelles images et gifs par exemple. J'ai aussi commencé à travailler sur une version du site utilisant un serveur pour l'implémentation d'un site dynamique avec une base de données afin d'avoir des utilisateurs qui peuvent se connecter ou créer des comptes. Ils peuvent aussi envoyer des messages d'avis du jeu sur la page d'accueil ou encore consulter leur page de profil avec les différentes informations concernant leur compte.

Pour la troisième soutenance, j'ai fini de mettre en place la base de donnée pour notre site utilisée avec MySQL et phpmyadmin. Les utilisateurs du site peuvent donc se connecter grâce à leur compte ou en créer un. Ils peuvent envoyer des messages sur la page d'accueil pour donner leurs avis par rapport au jeu ou au site mais aussi pour communiquer avec les autres utilisateurs. Chaque utilisateur a aussi la possibilité d'accéder à ses informations personnelles (celles données lors de l'inscription) et de les modifier. J'ai aussi ajouté du contenu par rapport à ce nous avons fait pour cette dernière soutenance comme les images montrant les dernières avancées dans le jeu.

2. Travail de Jordan :

Mon travail lors de la première soutenance a alors été de faire la mise en page de ces différentes pages et autres fonctionnalités du site :
— Menu
— Banderole
— Bouton d'accueil et bouton pour revenir en haut
— Animation des boutons
— Uniformisation du site
— Mise en place des liens
— Interface d'affichage des profils en JavaScript

Pour cela, pratiquement l'intégralité du site a été réalisée en CSS sauf l'interface d'affichage des profils qui a été faite en JavaScript. Ne voyant pas le moyen de faire une telle interface en CSS et n'ayant aucune connaissance en JavaScript, je me suis aidé d'un modèle que j'ai modifié pour obtenir ce que je recherchais. Il faut d'abord commencer par découper chaque page en différentes parties selon ce qu'on souhaite afficher. Pour cette division on utilise des blocs "div" auxquels on ajoute soit un "id" soit une "class" pour y faire appel dans le CSS de la page. On peut alors effectuer

de nombreux éléments de mise en page tels que l'alignement du texte, la position de certains éléments par rapport à d'autres, la marge intérieur et extérieur des différents blocs, gérer la couleur de fond, mettre des images en arrière-plan etc.

Nous avions de l'avance sur le site pendant la première soutenance, la majorité de la mise en page étant déjà faite, il ne me restait donc plus qu'à faire que le site soit adaptatif au petits périphériques pour la deuxième soutenance. Je n'avais jamais fait de site dont la mise en page était adaptée à l'appareil utilisé auparavant, donc j'ai dû me renseigner en amont. Globalement, si la vue du site devient trop petite le site s'adapte :
— le menu horizontal s'affiche en verticale,
— le bouton animé de l'accueil disparaît,
— la taille des polices du site diminuent,
— le contenu occupe 90% de la largeur de l'écran au lieu d'avoir une taille fixe,
— les tableaux et les gifs sont affiché les uns en dessous des autres.

Pour la troisième soutenance, il ne nous restait pratiquement rien à faire sur le site web. En effet, tous les éléments du site sont déjà présents ainsi que la mise en page générale. Il ne me restait plus qu'à bien placer les éléments ajoutés par le PHP, notamment la page "Connexion", la page "Compte" et la boite de commentaires.

4 Accord avec le planning

Pour cette soutenance nous sommes dans les temps que nous avions prévu lors de notre cahier des charges.

Soutenance / Tâche	**Soutenance 1**	**Soutenance 2**	**Soutenance 3**
Programmation			
Génération de la ville	85%	100%	100%
Génération du monde	20%	80%	100%
Pathfinding	100%	100%	100%
IA	5%	50%	100%
Multijoueur	60%	80%	100%
Menus & Interface utilisateur	20%	40%	100%
Art			
Textures	5%	30%	100%
Modèle & Animations	20%	40%	100%
Son & Musique	5%	20%	100%
Site Web			
HTML	65%	90%	100%
CSS	65%	90%	100%

Tâche en avance par rapport au pourcentage prévu Tâche effectuée

5 Réalisation

5.1 Fonctionnalités jouables

5.1.1 Lors de la première soutenance

Le joueur peut créer une salle d'attente, d'autre joueurs peuvent le rejoindre s'ils connaissent le nom de la salle. Une fois que tous les joueurs sont là, le joueur qui a créé la salle lance la génération de la ville. Une fois la ville générée, les joueurs peuvent rejoindre la partie. Ils ont chacun leurs équipes et peuvent faire apparaître un personnage qui rejoint leur équipe en appuyant sur la touche espace. Ils peuvent déplacer leur personnage (et n'ont pas le contrôle des autres personnages). Pour faire survivre leurs personnages, ils doivent leur faire ramasser de la nourriture trouvable dans certains bâtiments. Une fois ramassé, l'objet est ajouté dans l'inventaire et est utilisable pour nourrir le personnage.

5.1.2 Lors de la deuxième soutenance

Pour cette deuxième soutenance, les joueurs peuvent créer des salles avec des réglages spéciaux, d'autres joueurs peuvent les rejoindre, ils peuvent se mettre ensemble ou non. La taille de la carte où ils jouent étant décidée par celui qui a créé la salle. Le monde dans lequel les joueurs évoluent est un monde ouvert avec un réseau de villes et de villages inter-connectés avec des biomes différents impactant sur le type de végétation générée. Les bâtiments générés dans les villes ont des tailles et des formes diverses. Dans ces bâtiments se génèrent des meubles et des décorations possiblement propres aux bâtiments. Les personnages peuvent fouiller ces bâtiments pour trouver des objets à fonctions variées comme de la nourriture, des armes, des vêtements et des matériaux. Pour trouver ces objets, les personnages fouillent dans des conteneurs qui peuvent stocker les objets. Ces objets peuvent aussi être utilisés pour créer d'autre objets via de l'artisanat, les joueurs peuvent aussi placer des plans de construction que les personnages pourront construire. Chaque personnage a des traits qui le caractérisent, son état de santé est défini par l'état de ses différentes parties du corps. Les personnages peuvent se battre entre eux, ils peuvent mourir de saignements et d'infections. Leur état de santé impact leurs mouvements et leurs caractéristiques. Les joueurs peuvent aussi être mordus par des zombies. Ils peuvent aussi attraper des rats qui fuient à leur approche.

5.1.3 Fonctionnalités jouables actuelles

Dans un premier temps, le joueur arrive dans un menu où il peut choisir entre le mode de jeu solo, le mode de jeu en ligne, les options ou le fait de quitter le jeu. Selon son choix différents éléments seront proposés :

1. Le mode solo : le joueur peut alors choisir entre les différents modes de jeu disponibles tels que l'histoire solo, le mode grand froid ou encore le mode zombie...

2. Le mode en ligne : dans lequel le joueur va pouvoir se connecter au serveur Photon et créer ou rejoindre une salle dans laquelle il pourra créer ou rejoindre une équipe.

3. Les options : dans lesquels le joueur peut régler la taille de fenêtre de jeu ou encore le volume de la musique.

Ensuite le joueur arrive dans le jeu à proprement parler. Il peut alors faire apparaître des personnages qui vont être ajoutés dans les personnages contrôlables par les joueurs de l'équipe. Chaque personnage possède un système de traits physiques, mentaux, professionnels, d'expériences face à l'apocalypse. Il possède aussi des compétences sociales et des connaissances qui représentent son intelligence pour pouvoir produire certaines actions.

Les personnages sont aussi composés d'un système de santé avec les différentes parties du corps qui ont des spécificités et les blessures subies par les personnages ont des répercussions sur ces parties du corps (saignements, infections, fractures, etc). Ainsi si le personnage est blessé à la jambe, sa vitesse de déplacement sera réduite, la douleur provoquée affectera les capacités du personnage et si la blessure devient trop importante, il y a la possibilité que le joueur s'évanouisse ou meure.

Un système de combat est aussi présent pour permettre aux personnages de se défendre en cas de besoin. Ils peuvent s'armer ou s'équiper d'armures qui possèdent des caractéristiques de poids, de dégâts, de résistance...

Il existe aussi un système d'artisanat et de construction. Les personnages peuvent créer des nouveaux objets à partir de leurs objets via l'onglet artisanat de leur inventaire. Il existe des recettes permettant d'établir quelles combinaisons d'objets donnent quels objets. De plus, les joueurs peuvent placer des plans de construction, les personnages pourront finir la construction de ce plan en utilisant des objets.

Le monde dans lequel les joueurs évoluent est un monde ouvert avec un réseau de villes et de villages inter-connectés avec des biomes différents impactant sur le type de végétation générée. Les bâtiments générés dans les villes ont des tailles et des formes diverses. Dans ces bâtiments se génèrent des meubles et des décorations possiblement propres aux bâtiments. Les personnages peuvent fouiller ces bâtiments pour trouver des objets à fonctions variées comme de la nourriture, des armes, des vêtements et des matériaux. Pour trouver ces objets, les personnages fouillent dans des conteneurs qui peuvent stocker les objets. Ces objets peuvent aussi être utilisés pour créer d'autre objets via de l'artisanat. Les joueurs peuvent aussi être mordus par des zombies. Ils peuvent aussi attraper des rats qui fuient à leur approche.

5.2 Possibles améliorations

Tales of the Apocalypse est désormais fini par rapport à ce que nous avions prévu de faire. Cependant le développement de notre jeu pourra continuer, en améliorant tous les éléments que nous avons jusque là implémentés dans le jeu mais aussi en rajoutant de nouvelles fonctionnalités. Nous pourrons ainsi amélioré toutes les interfaces utilisateurs avec de nouveaux éléments afin que le joueur puisse plus interagir avec des éléments du monde et qu'il se sente encore plus imprégné dans le jeu.

Nous avons aussi d'autres idées pour ajouter de nouveaux contenus dans notre jeu, par exemple avec l'implémentation de nouveaux modes de jeu ou de nouveaux événements dans le jeu.

Nous pourrons aussi ajouter des nouveaux éléments dans le jeu tels que de nouveaux bâtiments mais aussi implémentés des voitures afin de se déplacer plus mieux rapidement dans le monde ou encore amélioré le système de personnage non-jouable afin qu'ils puissent mieux interagir avec nos personnages pour les aider dans la recherche de nourriture, d'armes, ou autres ou bien qu'ils aient des comportements plus aléatoires comme par exemple attaquer nos personnages pour aucune raison.

Une autre amélioration du jeu pourrait être d'améliorer le système de construction afin de pouvoir construire au fur et à mesure de vrais bâtiments et de pouvoir améliorer les bâtiments avec des éléments trouvables en fouillant de le monde pour permettre à ces bâtiments par exemple de mieux s'isoler thermiquement.

5.3 Logiciels utilisés

Discord : Logiciel gratuit qui nous a permis de communiquer entre nous et d'échanger sur l'avancement de nos parties tout au long du projet.

GitHub : GitHub est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git que nous avons utilisé pour nous échanger nos différents fichiers ainsi que pour l'hébergement de notre site web grâce au GitHub Pages.

Sublime Text : Éditeur de texte pour le développement du site internet. Il est riche en fonctionnalités, ce qui permet une grande flexibilité ainsi que des outils adaptés aux pour les langages de programmation utilisés.

Unity : Il s'agit d'un logiciel d'édition de scènes, et moteur de jeu puissant. Ce fut notre arme principale pour l'élaboration de ce projet. C'est le point central vers lequel convergent tout les scripts et les accessoires (textures, audio, vidéo, etc...).

Visual Studio : Environnement de développement intégré pour le développement en C#. Il nous a permis d'écrire les scripts pour Unity.

MiMind : Logiciel qui permet de poser ses idées sous forme d'arbre de réflexion pour mieux voir les liens entre chaque idées.

MagicaVoxel : Logiciel qui permet de modéliser des personnages, des éléments en 3D, ce qui nous a été utile pour nos graphismes du jeu.

5.4 Coûts matériels et logiciels

Composants PC	Yanis Chaabane	Tanguy Desgouttes	Jordan Failloux	Marc-Emmanuel
Processeur	Intel Core i5 (4 coeurs)	Intel Core i5 (4 coeurs)	Intel Core i5 (4 coeurs)	Intel Core i5 (4 coeurs)
Carte Graphique	Nvidia GeForce MX150	NVidia GeForce GTX 1050TI	Nvidia GeForce GTX 1050	Nvidia GeForce GTX 1070
RAM (en Go)	8	8	8	8
Coût	680 €	1100 €	700 €	1200 €

	Coûts
Ordinateurs	3680 €
Licenses Adobe	299.95 €
Total	3979.95 €

5.5 Modèle économique

Nous avons choisi un modèle P2P (Pay To Play) comme modèle économique, le joueur doit payer pour jouer au jeu , une fois une somme d'argent payée le joueur possède le jeu. Le jeu sera disponible sur Windows, la vente passera par la plate-forme Steam. Bien que Steam prenne des parts sur les ventes, la visibilité gagnée y est importante. Nous nous sommes longtemps posés la question du prix du jeu, bien que Steam propose de la visibilité supplémentaire pour les jeux dont les prix sont inférieurs à 10 € et à 5 €, une études des données disponibles sur SteamSpy montre que les jeux dont le prix est compris entre 15 € et 20 € font plus de ventes moyenne que ceux compris entre 10 € et 15 €. Nous sommes suffisamment confiant en notre jeu pour ne pas tromper les attentes d'un joueur en le vendant, dans un premier temps, à 15,99 €. En effet, notre développement suit un modèle en cascade où nous serons à l'écoute permanente des joueurs, c'est la phase de "Early Access", à la fin de cette phase, lors de la sortie du jeu, nous augmenterons le jeu à 19,99 €, pratique courante dans le milieu (exemple récent de *Slay The Spire*) permettant de fidéliser les anciens joueurs et de marquer la sortie de l'Early Access afin d'attirer de nouveaux joueurs. Une fois le jeu sorti, nous pourrons envisager la création de contenus supplémentaires payants (DLC) rajoutant des objets, des animaux ou des scénarios.

5.6 Futur de *Tales of the Apocalypse*

Comme annoncé plus haut, nous avons choisi un modèle de développement en cascade, cela veut dire que si le jeu venait à sortir, il sortirait en "Early Access" (accès anticipé). Nous pourrons alors recueillir le retour des joueurs, tout en continuant un développement constant. Le genre de *Tales of the Apocalypse* étant extrêmement niche, l'avis des joueurs est d'autant plus important. Le développement continu du jeu serait permis par nos implémentations flexibles des différents systèmes et serait justifié par une conséquence intéressante de l'utilisation de génération procédurale étant le fait que la qualité du jeu est grandement lié à la quantité de contenu jouable dans le jeu. L'avis des joueurs nous permettraient alors de créer du contenu de qualité en quantité.

5.7 Problèmes & Solutions

Jusqu'alors quelques problèmes ont été rencontrés. Ces problèmes n'ont pas été si importants que cela aurait pu être. Voici une liste de nos quelques problèmes :

1. L'utilisation de git en équipe : Nous avions tous été formés à git à travers nos nombreux travaux pratiques de programmation mais l'utilisation de git en équipe de plusieurs personnes est évidemment différente. Il a donc fallu faire des branches pour que chacun puisse travailler sur les différentes parties qui lui incombaient. Cependant il fallait très souvent gérer les nombreux conflits lorsque l'on voulait "merge" (faire une mise en commun dans la branche principale). Ces nombreux conflits étant relatifs à de petites différences lorsqu'on travaillait sur les mêmes scènes Unity étaient très dures à gérer car ces fichiers ne sont pas facilement manipulables comme peuvent l'être les scripts en C# dans lesquels même si des conflits sont présents, on comprend assez aisément comment les résoudre. Nous avons donc décidé de mieux gérer ces conflits en gardant l'idée que chacun travaille sur une branche et en même temps nous faisons des copies des éléments à potentiels conflits. Par exemple pour ne pas avoir de conflits sur un" prefab", Tanguy et Yanis vont faire une copie du "prefab" et travailler chacun sur leur copie. Au moment de la mise en commun via le "merge" de git, il n'y a donc pas de conflits sur ce "prefab", et il faut faire les quelques changements apportés par chacun sur le "prefab" principal.

2. Un autre problème est apparu avec la génération procédurale. A un moment du développement, le système s'est heurté à un mur, Photon ne pouvait pas supporter la transmission d'autant d'éléments en multijoueur, en effet jusqu'alors il transmettait l'entièreté des routes et des bâtiments aux autres joueurs. Il fallait donc trouver un moyen de n'envoyer que les informations essentielles aux autres joueurs. Les fonctions RPC ("Remote Procedure Call") permettent d'envoyer des informations de certains types aux autres joueurs (comme des nombres ou des chaînes de caractères) à travers l'appel de fonctions spéciales. Il a donc fallu traduire les informations de génération en une suite de nombre (rangée dans une liste), les envoyer aux autres joueurs puis les retraduire en informations de génération. Ainsi dans la version actuelle du projet, la génération est décidée par un seul joueur, dit "Master" (celui qui a créée la "room")

puis les informations de génération sont compactées dans une liste de nombres puis décompactées par les autres joueurs, la génération des bâtiments se fait donc localement.

3. Problème lors de l'adaptation au modèle 3D du personnage et son animation. Avant d'importer ces derniers éléments on utilisait des cubes pour imiter des joueurs, on pouvait alors les faire se déplacer. Mais à partir du moment où l'on a remplacé ces cubes par les modèles 3D des personnages et leurs animations cela ne marchait plus correctement. Le problème ne venait pas des animations en elles-mêmes ni de l'animator mais des informations envoyées par le navmesh. L'agent ne détectait simplement pas bien le sol et le personnage flottait au dessus, ce qui perturbait le programme. Il a simplement fallu changer quelques paramètres pour qu'il détecte bien le sol.

De par notre expérience et nos méthodes acquises pour la première soutenance, nous n'avons pas eu de problèmes trop importants lors de la deuxième soutenance. Le seul problème rencontré était l'envoi de donnée en masse lors de la génération d'un très grand monde provoquant la déconnexion des joueurs du serveur de Photon. La solution étant d'optimiser la génération pour ne pas envoyer toutes les données en même temps.

6 Synthèse des expériences individuelles

6.1 Tanguy

L'expérience du projet jusqu'à présent a pour moi été utile et révélatrice sur plusieurs plans. Tout d'abord sur le plan de la programmation j'ai pu grandement m'améliorer et me familiariser avec les codes à respecter pour travailler en équipe. Le fait d'apprendre en autodidacte à utiliser Unity m'a aussi permis de changer mes méthodes d'apprentissage, une rigueur plus importante étant nécessaire. Sur le plan du travail d'équipe, ayant été désigné chef de projet, j'ai pu expérimenter les meilleurs moyens de répartir le travail au sein du groupe et de garder ce dernier soudé et motivé, expérience et pratiques qui me seront à coup sûr utiles dans le futur.

6.2 Yanis

Après 6 mois de travail acharné sur ce projet, je peux dire que celui-ci m'a apporté de nombreuses choses. En effet il m'a permis de travailler sur le coté programmation et de m'améliorer dans ce domaine mais aussi de me familiariser avec de nouveaux éléments tel que Unity ou encore du côté du site Web.

Cet apport de connaissances n'est pas négligeable puisque ces connaissances pourront nous resservir à tout moment le long de notre cursus scolaire ou même plus tard en tant qu'ingénieur.

Ce projet m'a aussi apporté beaucoup pour mon futur métier d'ingénieur avec la réalisation du projet en groupe, nous apprenant donc à mieux nous organiser et à mieux gérer la division du travail et de l'échange d'informations entre membres afin de toujours être au courant des avancées des autres.

J'ai aussi pu apprendre à toujours chercher des solutions aux problèmes que nous rencontrons et de persévérer jusqu'à la réussite et même si des fois, psychologiquement, c'est dur de rester bloquer pendant des heures sur un même problème, j'ai appris à ne pas lâcher car c'est un travail de groupe et je ne peux pas juste abandonner et laisser les autres se débrouiller avec les problèmes que je leur ai laissé.

6.3 Jordan

En travaillant sur ce projet, j'ai pu apprendre beaucoup de nouvelles choses. Tout d'abord de nouvelles notions de programmation en C# mais aussi de développement web, notamment en CSS avec la réalisation de site adaptatif. Depuis ma première expérience de création de jeu sur Pygame en ISN, j'ai toujours voulu utiliser un moteur de jeu tel que Unity pour explorer et utiliser toutes ces fonctionnalités que Pygame ne possède pas. Travailler en groupe de quatre m'a également beaucoup appris. En effet, c'est très différent du travail en groupe de deux que j'ai pu expérimenter en ISN. Grâce à cela, j'ai pu me rendre compte de l'importance de la répartition des tâches et de la communication dans un groupe.

6.4 Marc-Emmanuel

M'étant occupé majoritairement de la modélisation, de l'animation et des aspects graphiques pendant ces six premiers mois c'est surtout dans ce domaine que je me suis amélioré. Voir ma propre progression au fil du travail a été un facteur de motivation majeur et les tâches que j'effectuais en une heure au début de ces quatre mois ne me prennent souvent plus que vingt à trente minutes maintenant. Au delà de ça j'ai pu progresser un peu en programmation et ai appris beaucoup de fonctionnalités de Unity en suivant les autres travaux en cours du projet, notamment en lisant les différents scripts du jeu pour savoir ce que j'allais devoir faire de mon coté.

Enfin la chose la plus importante que j'ai pu expérimenter et pratiquer durant ces six mois est le travail en équipe, les sentiments de devoir envers le groupe sont très positifs et m'ont permis d'avancer même quand il m'est arrivé de craquer un peu après de longues sessions de travail. C'est cette dernière partie de l'expérience qu'est le projet qui me sera sans doute la plus bénéfique pour le futur et que j'ai le plus appréciée jusqu'à maintenant.

7 Conclusion

On arrive à la fin de ce rapport et nous sommes fiers du travail que nous avons réussi à produire. Ce fut long et souvent semé d'embûches mais nous avons su garder le cap grâce à notre détermination, notre cohésion et la bonne ambiance au sein du groupe. Nous sommes heureux de pouvoir rendre un jeu fini comme nous le prévoyions et qui répond à toutes nos attentes. Ce projet aura été bénéfique pour chaque membre du groupe nous apportant de nombreuses connaissances tant sur le plan technique de la programmation avec Unity mais aussi sur l'approche professionnelle de ce projet qui nous aura appris d'ores et déjà à gérer les projets en groupe dans un délai déterminé et à respecter un cahier des charges comme nous aurons l'occasion de le faire dans nos futurs métiers.

8 Annexes

8.1 Sources

- https ://doc.photonengine.com/en-us/pun/v1/demos-and-tutorials/pun-basics-tutorial/intro
- https ://docs.Unity3d.com/Manual/index.html
- https ://Unity3d.com/fr/learn/tutorials
- https ://pages.github.com/
- https ://nc.me/
- https ://www.w3schools.com/
- https ://www.udemy.com
- https ://www.youtube.com/channel/UCYbK_tjZ2OrIZFBvU6CCMiA
- https ://www.youtube.com/channel/UCGIF1XekJqHYIafvE7l0c2A
- https ://www.youtube.com/user/Cercopithecan
- https ://soundcloud.com/thescrapboy/tracks

8.2 Annexes

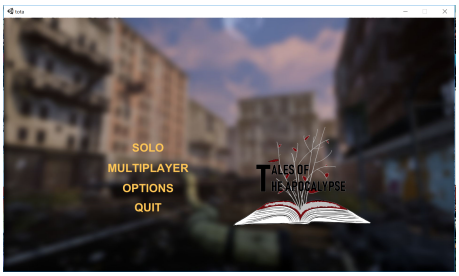

FIGURE 3 – Menu principal

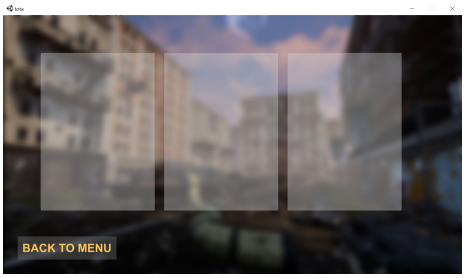

FIGURE 4 – Choix du mode solo

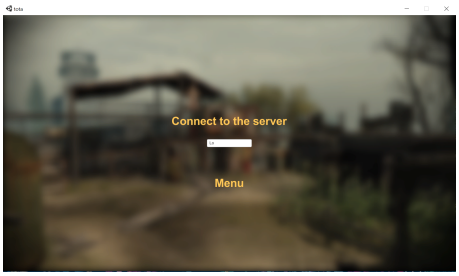

FIGURE 5 – Connexion au serveur

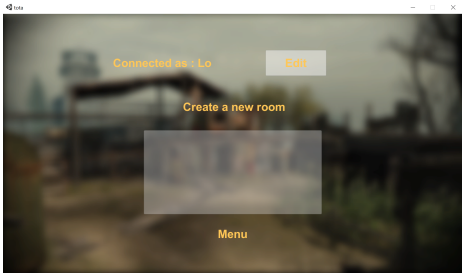

FIGURE 6 – Choix de la salle

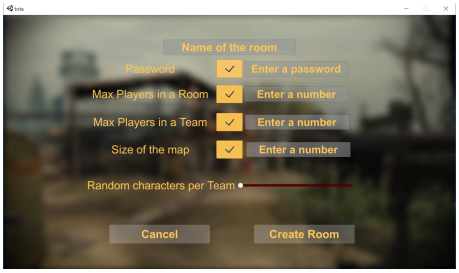

FIGURE 7 – Création des paramètres

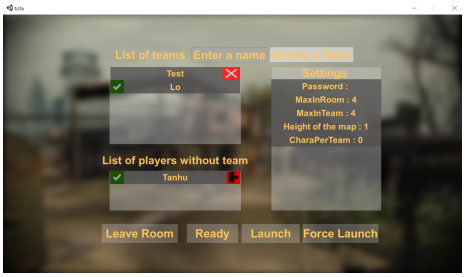

FIGURE 8 – Attente dans la salle

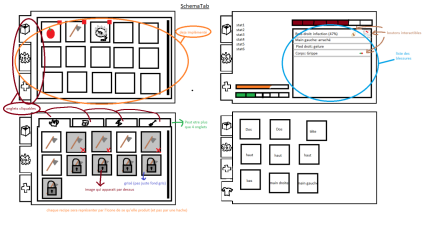

FIGURE 9 – Idée d'interface

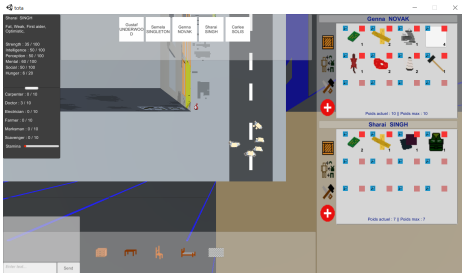

FIGURE 10 – Interfaces du jeu

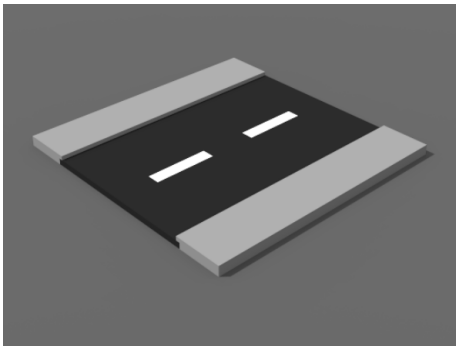

FIGURE 11 – Route

FIGURE 12 – Réfrigérateur

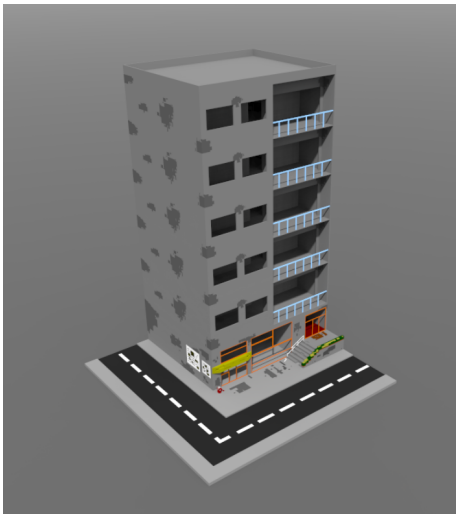

FIGURE 13 – Bâtiment

FIGURE 14 – Ancien modèle de personnage

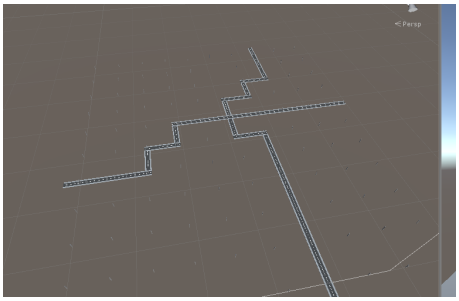

FIGURE 15 – Première génération

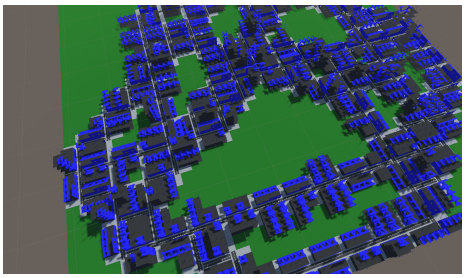

FIGURE 16 – Deuxième génération

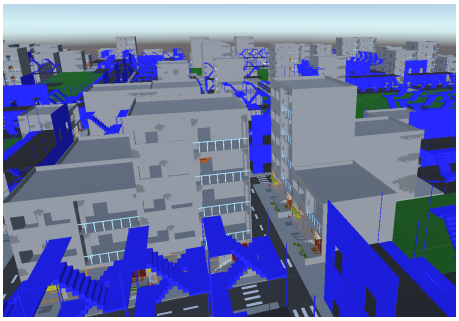

FIGURE 17 – Vision de près

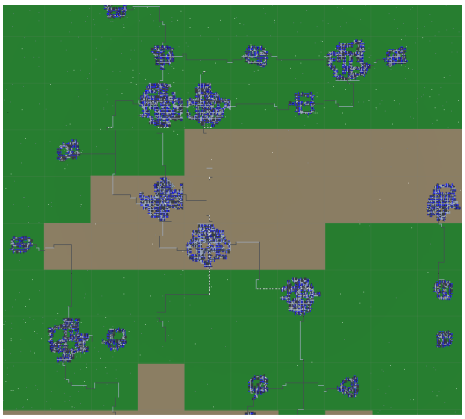

FIGURE 18 – Génération finale

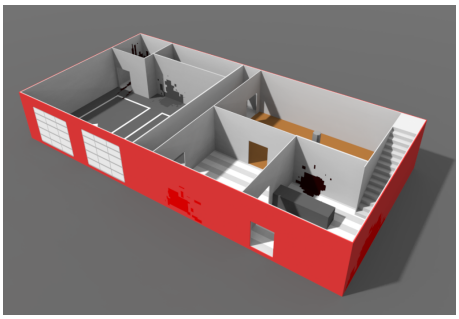

FIGURE 19 – Bâtiment

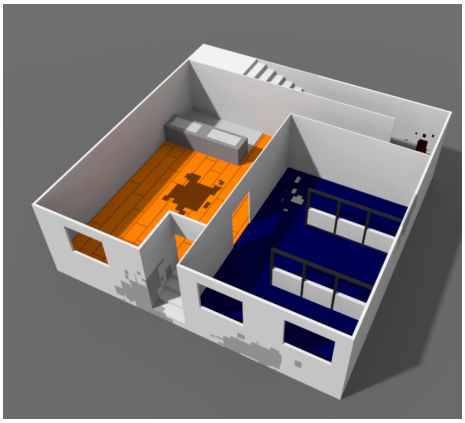

FIGURE 20 – Bâtiment

FIGURE 21 – Bâtiment

FIGURE 22 – Bâtiment

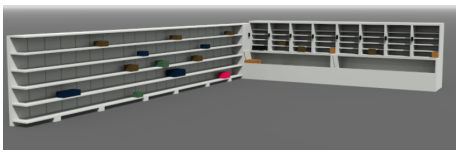

FIGURE 23 – Bâtiment

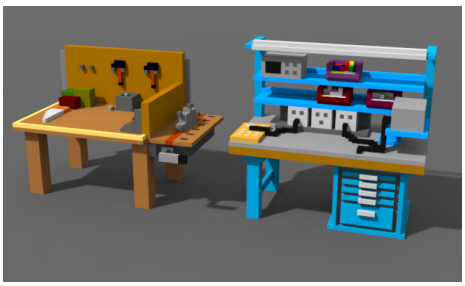

FIGURE 24 – Bâtiment

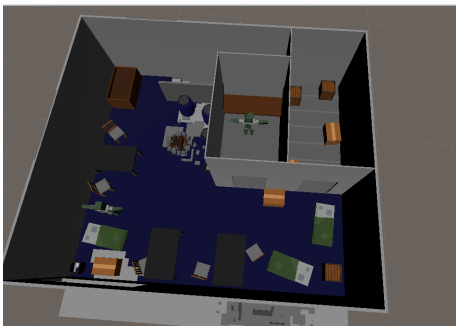

FIGURE 25 – Bâtiment

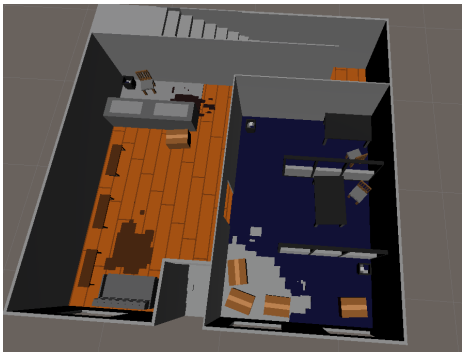

FIGURE 26 – Bâtiment

FIGURE 27 – Bâtiment

FIGURE 28 – Bâtiment

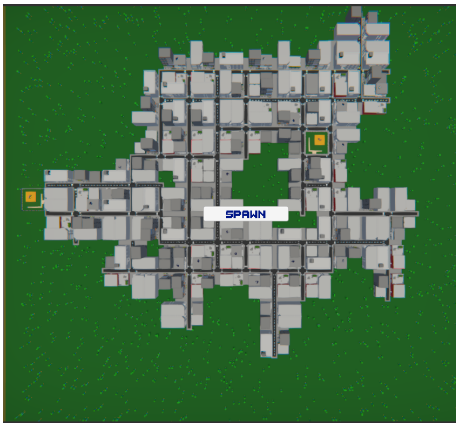

FIGURE 29 – Génération du monde

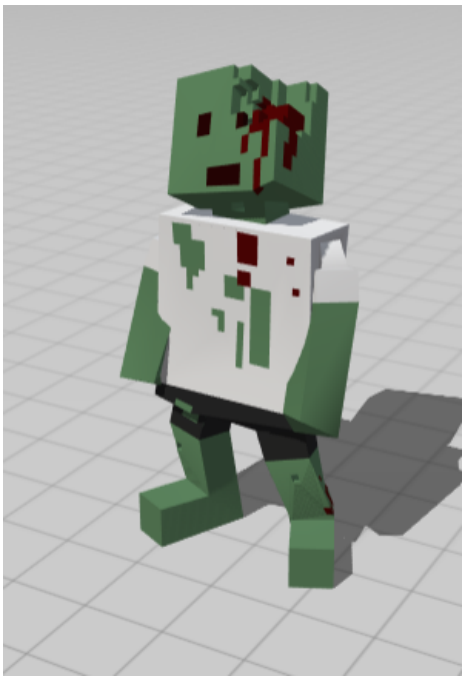

FIGURE 30 – Zombie

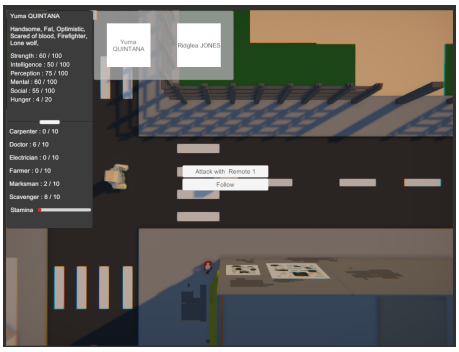

FIGURE 31 – Actions

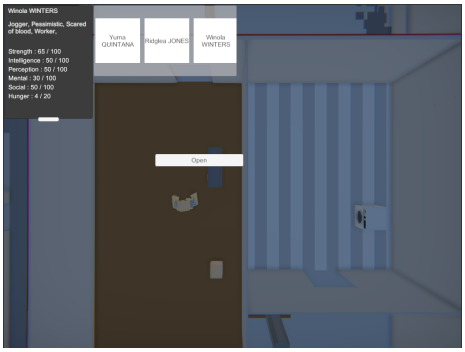

FIGURE 32 – Actions

FIGURE 33 – Actions de construction

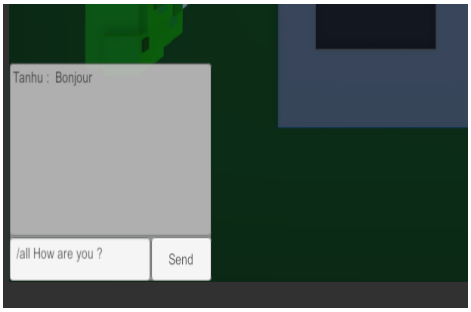

FIGURE 34 – Canal de communication