Rapport De Soutenance 2 - Year Zero

Bearth Studio

21 avril 2019

timothee.ribes nicola.brankovic

enguerrand.vie axel.ribon

Table des matières

1 Introduction

Year Zero est un jeu de stratégie se déroulant dans l'espace et dans lequel plusieurs équipes s'affrontent pour la suprématie de l'univers. On y contrôle des bâtiments qui produisent des unités sur une carte aux limites fixées.
Le moteur de jeu choisi est Unity et le langage utilisé est le CSharp (C). La direction artistique est fantaisiste et non réaliste afin de donner un ton sombre mais non choquant. L'objectif étant de rendre le jeu tout public. L'aspect cartoon a été préféré au réalisme afin de rendre l'ambiance du jeu assez légère mais sans perdre l'aspect stratégie. Cependant nous avons tenu à garder un certain degré de réalisme afin de garder un côté dramatique au vu de l'histoire du jeu. D'un point de vue sonore le jeu est assez épique avec de grands thèmes entraînants. Mais il jongle aussi avec des thème plus calmes et mélancoliques pour rappeler la condition désespérée de l'humanité. L'aspect épique étant purement ludique et l'aspect dramatique servant à renforcer les intentions du scénario.

2 Développement sur les tâches réparties

2.1 Timothée

Mon travail en tant que chef de projet se situe toujours entre les différents domaines que nous utilisons, tout en étant focalisé sur les aspects sonores,

2.1.1 Mission

La mission fait partie de l'expérience solo que propose Year Zero et se place dans la continuité du tutoriel qui doit être terminé pour y accéder, et va présenter la mécanique la plus attractive dans un jeu de stratégie en temps réel, les combats dont le fonctionnement et l'implémentation sont détaillés dans le travail d'Axel. L'ennemi ne possède donc pas encore de bâtiments économiques ou de production, et le joueur peut se concentrer sur le simple fait de se débarrasser des unités qui vont attaquer sa base en suivant les instructions introduites par la narrateur. D'autre part, un travail d'ajustement devra se faire avec les rééquilibrages du jeu pour que cette partie reste assez simple à exécuter pour le joueur novice. Enfin cette mission a aussi pour but d'introduire le scénario en effectuant le premier contact entre lui et l'antagoniste qui envoie des troupes pour le détruire. De plus, un autre mode qui est sans fin à été ajouté, c'est un mode que nous avions prévu d'ajouter et qui est similaire à cette mission dans lequel le joueur va donc encore une fois faire face à des vagues qui ne lui laisserons pas de repos et se répéteront à l'infini en étant toujours plus fortes pour constituer un challenge pour atteindre les vagues les plus avancées.

2.1.2 Sons

En parallèle il a fallu continuer à développer l'ambiance sonore du jeu, déjà pour les voix des personnages qui arrivent au fur et à mesure de l'histoire, ce qui nous a forcé à faire du montage audio à l'aide du logiciel Audacity et pour lesquelles nous avons choisi d'utiliser nos voix malgré notre absence d'expérience dans le domaine du doublage.Nous avons réalisé la diversité des métiers impliqués dans un jeu. Nous espérons donc pouvoir être sérieux sur ce domaine mais laissant tout de même notre trace à travers nos voix aussi modifiées qu'elles soient. On prend donc soin de préparer les répliques avec une certaine idée du ton qu'elle doivent avoir, puis on effectue des essais jusqu'à avoir une version pertinente dont on enlève ensuite le bruit et à laquelle on ajoute des effets de hauteur pour coller au personnage qui doit être incarné. Aussi, je me suis assuré que l'aspect musical soit fonctionnel dans le jeu, c'est à dire qu'on puisse lancer les musiques en fonction du moment, qu'elles ne se superposent pas, ni

ne se redémarrent ou se coupent en fonction des changements d'environnement comme rejoindre une partie ou changer de menu. De plus pour que la musique dans ce jeu soit une partie intégrante de l'expérience de jeu j'ai pensé à, mieux que faire une liste de sons aléatoire qui se lisent au fur et à mesure d'une partie, allier leur ambiance à l'état de la partie. En effet des musiques plutôt calmes seront jouées en début de partie, puis au fur et à mesure que le joueur débloque des unités ou des technologies, les sons seront de plus en plus énergiques ayant pour but d'aboutir à un cadre épique pour les grandes batailles de fin de partie qui mettront en jeu beaucoup d'unités.

2.1.3 Equilibrage

L'équilibre d'un jeu de stratégie, même dans sa version la moins poussée est un concept principal, de façon à ce que les différentes unités forment un tout. Il permet que chaque vaisseau de notre jeu ait sa place et son utilité soit en ayant des statistiques avantageuses pour un coût plus élevé, ou plus faibles mais avec des sorts plus puissants. Atteindre un équilibre parfait nécessiterait une analyse de données sur un nombre énorme de parties et de joueurs, c'est pourquoi nous nous sommes appuyés sur des séances d'essais personnels du groupe qui devront être poussées plus loin pour la prochaine soutenance afin de continuer à affiner l'équilibre et une bonne expérience intéressante. Pour cela nous avons créé un tableau rassemblant toutes les unités auxquelles nous avons assignées les diverses statistiques : vitesse d'attaque, dégâts, portée, points de vie, vitesse et leur coût dans les trois ressources du jeu. A cela nous avons ajouté des coefficients à chacune de ses valeurs et les avons regroupés dans une valeur propre à l'unité. Notre but était alors que toute les unités aient la même valeur tout en modulant leur différentes caractéristiques, mais nous avons abouti à quelque chose où les valeurs tendaient à croître en fonction de la fin de partie car nous avions négligé les recherches à faire à travers l'arbre de compétence nécessaire pour obtenir ces unités qui sera lui équilibré pour la prochaine soutenance avec le projet d'avoir un déroulement de partie complet jouable et agréable.

2.2 Nicola

Non donné à temps pour impression.

2.3 Enguerrand

Entre la première soutenance de ce projet et la seconde, l'essentiel de mon travail s'est concentré sur l'amélioration de détails graphiques, principalement sur l'interface, ainsi que l'ajout d'une partie des modèles 3D des unités ainsi que des bâtiments. Une idée directrice pour codifier la partie graphique de Year Zero est de ne rien laisser par défaut, il faut que nous puissions mettre notre empreinte dans tous les aspects de ce que Unity nous propose comme personnalisation ? C'est à dire remplir tous les menus avec nos images conçues pour cela, l'ajout du curseur personnalisé et bien d'autres.

2.3.1 Ajouts de ressources graphiques

Ajout d'un curseur personnalisé

A l'issue de la première soutenance, je me suis dit qu'il fallait supprimer tout ce qui, jusqu'alors était *par défaut*. C'est dans cette démarche que j'ai eu l'idée de concevoir un curseur qui viendrait remplacer celui que *Windows* possède.
Il fallait alors que ce curseur puisse s'intégrer dans la charte graphique du jeu sans pour autant perdre en visibilité. Après plusieurs essais non concluants, souvent parce que les couleurs choisies le rendaient moins visible, je suis enfin parvenu à la version actuelle qui est intégrée au jeu.

FIGURE 1 – Curseur présent dans Year Zero

Amélioration de la visibilité
Un jeu dont l'environnement dominant est l'espace comme Year Zero, a souvent des couleurs sombres. Les textes doivent être présentés dans des couleurs vives pour qu'ils soient bien vu de la personne devant l'écran. Ainsi la plupart des messages auront des couleurs comme le rouge ou le jaune-doré. Pour certains ils pourraient être encore plus visible et c'est dans cette optique que je leur ai appliqué un fond noir en dégradé de sorte qu'ils soient encore plus marquants parmi tout ce qui est affiché à l'écran.
Ainsi les messages du tutoriel expliquant comment jouer seront un peu plus visibles ainsi que les messages d'aide sur les boutons d'actions des unités, qui ne comportaient d'ailleurs pas de message d'aide auparavant.

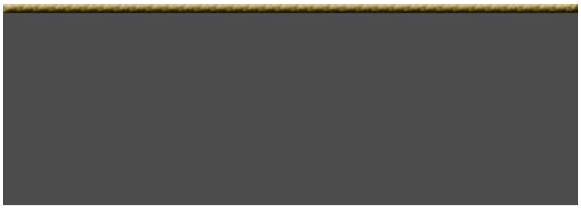


FIGURE 2 – Bandeau de fond pour les textes

FIGURE 3 – Aide pour les actions des unités

Par ailleurs, certains menus ont étés améliorés avec quelques détails graphiques pour les rendre plus dynamique. Il s'agit de l'écran de connexion, et du *lobby*, la salle d'attente avant que la partie ne se lance en multijoueur.
J'ai aussi appliqué les créations graphiques de Year Zero au menu que l'on peut trouver en jeu, lors d'une partie. On y accède en appuyant sur le raccourcis écrit sur les boutons de navigation en haut de l'écran ou en appuyant sur la touche *echap*.

FIGURE 4 – Menu principal dans une partie

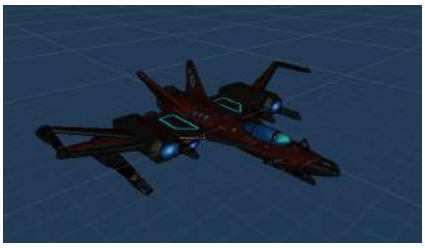
2.3.2 Modèles 3D

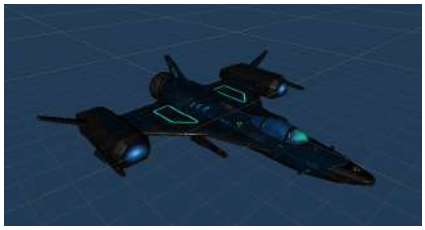
Les unités

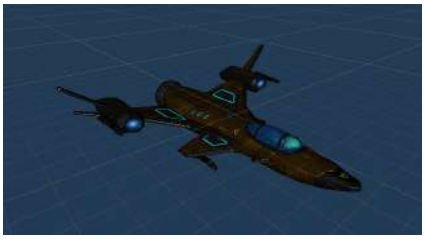
Year Zero, comme tout bon jeu de stratégie, offre de nombreuses unités. Il faut alors les habiller afin qu'on puisse les différencier. Après de nombreuses recherches de modèles 3D pouvant correspondre à nos besoins, j'ai trouvé un assortiment de ressources qui vont pouvoir constituer une majeure partie des modèles associés aux unités, pour ceux qui joueront les humains tout du moins. C'est ainsi qu'on retrouve pas moins d'une quinzaine de vaisseaux différents, chacun pouvant être décliné en plusieurs teintes, qui à terme viendront marquer les différentes équipes avec la couleur choisie en salle d'attente de partie.

FIGURE 5 – Vaisseaux de Year Zero en plusieurs couleurs

Certains des vaisseaux présents ne feront pas partie du jeu, nous ne les avons pas encore tous finalisés, néanmoins la variété de couleurs et de type de texture différents qui sont à notre disposition nous donnent de nombreuses possibilités pour le jeu en version finale afin de proposer des modèles s'inscrivant dans la charte graphique de Year Zero.
Vous pouvez voir ici des images approchées de certains des vaisseaux de Year Zero.







Un aspect important des modèles de vaisseaux, c'est qu'ils sont modulaires. C'est à dire qu'ils sont décomposable en plusieurs parties constituantes. Ainsi il nous sera plus aisé de matérialiser une destruction voire même une animation de construction de ces vaisseaux.

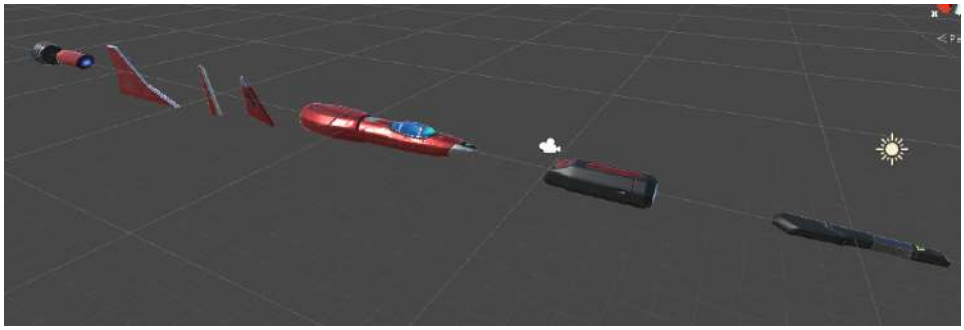
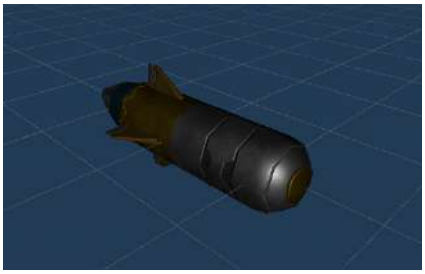
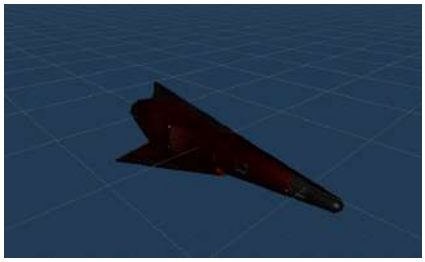


FIGURE 6 – Modularité d'un vaisseau

Outre ces modèles correspondant aux unités même, il y a tout un aspect des unités dans la 3D, qui est souvent oublié, mais nous ne pouvons passer à côté. Il s'agit de l'ensemble des projectiles qui seront tirés par les unités. Cela va du simple laser aux missiles, en passant par les bombes et autres. Outre le laser qui n'est pas encore dans le jeu, puisqu'il consiste en une animation de particule et que cette dernière n'est pas encore faite, on ne le retrouve pas dans cette version du jeu. De même pour les autres projectiles puisqu'ils seront associés à des effets de particule, des réacteurs pour les missiles, des effets d'explosion etc. Tout ceci devra être présent dans la version du jeu que nous rendrons pour la troisième et dernière soutenance qui marquera la fin du projet. Nous sommes dans l'obligation de fournir un jeu complet pour cet évènement.
Ci-dessous, quelques exemples des modèles de projectiles de tous types qui viendront rejoindre les modèles 3D de Year Zero.

Les bâtiments

Dans Year Zero il n'y a pas moins d'une dizaine de bâtiments différents. La plupart sont du même type, fixes et relativement ressemblant et quelques autres singuliers comme le radar ou encore la tourelle de défense. Donner des modèles à ces bâtiments n'est pas une tâche simple puisqu'ils doivent refléter l'atmosphère du jeu, ne pas briser l'unité d'un environnement spatial que l'on retrouve dans tous les aspects de ce que le jeu nous donne à voir sur l'écran. Ainsi, j'ai ajouté des ressources 3D correspondant à des modules d'un type futuriste sinon spatial qui, mis les uns avec les autres devraient pouvoir nous offrir assez de possibilités pour nos nombreux bâtiments, pour que chacun ait un modèle unique, se distinguant des autres. La plupart ne sont pas encore assignés puisque je suis encore à la recherche de modèles qui pourraient mieux convenir. Une chose à marquer sur la liste des objectifs de la troisième soutenance.

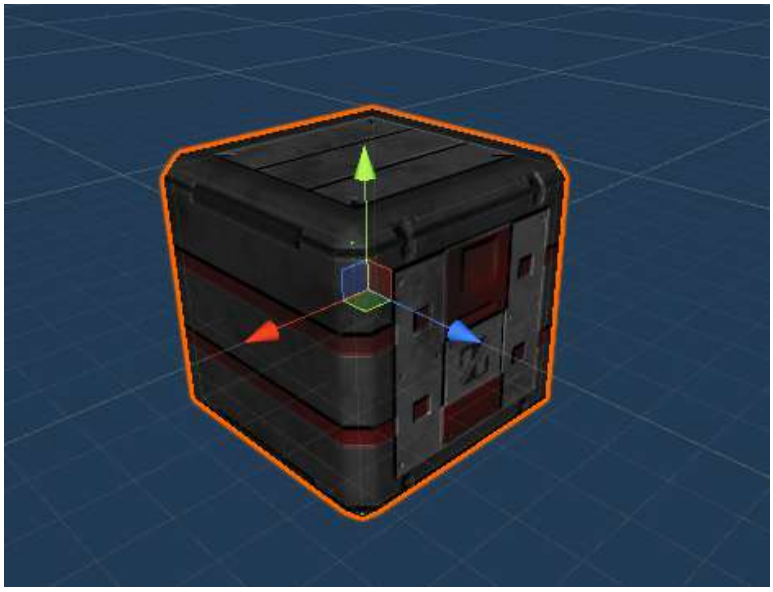


FIGURE 7 – Exemple d'un modèle modulaire

Modèles additionnels

Un environnement spatial a pour vocation d'être vide, mais il est apparu que remplir ce vide au moins par quelques éléments graphiques additionnels serait une bonne idée et rendrait l'environnement encore plus prenant. C'est pourquoi j'ai rajouté quelques astéroïdes qui vont venir prendre place dans la carte du jeu, dans une partie. Ils ne sont que purement décoratifs, on ne peut interagir avec eux, notamment pour récolter des ressources. Ils sont placés plus ou moins haut en fonction du niveau de référence qui est celui des unités et des bâtiments de sorte que l'effet d'environnement spatial soit conservé au maximum et améliorer l'immersion du joueur. Les astéroïdes sont déclinés en plusieurs tailles et possèdent différentes textures que l'on va leur appliquer afin de choisir l'aspect qu'ils ont en fonction du type d'environnement spatial de la partie. Par exemple on pourra appliquer la texture de glace aux astéroïdes lors d'une partie avec comme carte choisie un espace de glace. C'est une des fonctionnalités que je prévois d'ajouter au jeu final, présenté lors de la troisième soutenance.

FIGURE 8 – Petite taille

FIGURE 9 – Taille moyenne

FIGURE 10 – Plusieurs astéroïdes

Toujours dans le soucis de l'amélioration de l'immersion et du rendu 3D du jeu, j'ai aussi ajouté des ressources 3D de débris spatiaux. On y retrouve nombreux débris divers, de bâtiments ou même de vaisseaux. En fonction de la carte on pourra retrouver nombre de ces débris comme élément de décor au même titre que les astéroïdes.
Il est important de noter également que certains de ces débris nous servirons lorsqu'une unité ou un bâtiment sera détruit, nous afficherons alors le modèle 3D de débris correspondant.

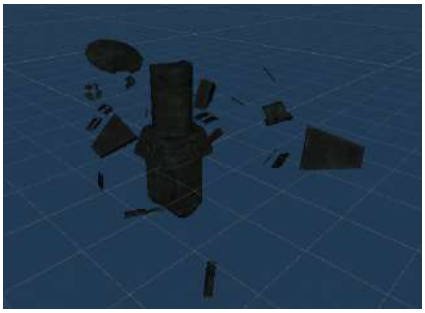


FIGURE 11 – Modèle de débris

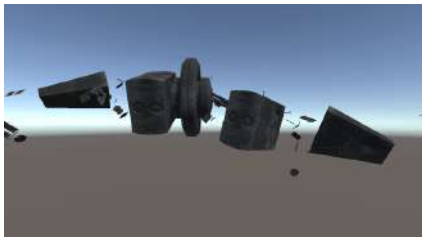


FIGURE 12 – Autres débris

2.4 Axel

Pour cette deuxième phase de développement je me suis totalement focalisé sur l'intelligence artificielle, le reste du jeu étant quasiment fini. Cela étant dit, plusieurs sessions de débogage ont eu lieu.

2.4.1 Préparation à l'IA

Avant de commencer le cœur de l'IA, j'ai d'abord dû préparer le jeu à son arrivée. N'ayant jamais développé d'IA pour ce type de jeu mes algorithmes n'étaient pas prévu pour cela. En effet jusque-alors l'identification du joueur auquel appartient une unité passait par le système implémenté pour le multi-joueur, or il n'est pas possible de créer des joueurs « factices » correspondants aux IA. L'hôte de la partie (celui qui l'a créé), doit ainsi contenir toutes les IA localement. J'ai dû alors implémenter des méthodes pour identifier si une unité appartient à un joueur ou une IA et réagir en fonction.
Ensuite, j'ai dû implémenter des outils pour pouvoir permettre à l'IA de contrôler son jeu. En effet, des fonctionnalités comme l'instanciation d'unité ou la construction de bâtiment ont été conçu à la base pour le joueur qui est équipé d'une souris et d'un clavier notamment. J'ai alors développé différents module pour chaque type d'action :
-L'Armée : Cet objet va permettre de savoir si l'armée de l'IA est prête pour l'attaque ou non, et ainsi l'envoyer attaquer.
-Les Constructeurs : Ce module permet de récupérer des constructeurs en fonction de leurs actions en cours (ne fait rien, mine ou construit) ainsi que les répartir équitablement entre toutes les tâches possibles.
-La Construction : Permet à l'IA de lancer simplement l'ordre de construire un bâtiment. Pour l'heure la position du bâtiment généré en forme d'escargot.

C'est-à-dire que les bâtiments sont construits autour de la Station Spatiale initiale pour former un carré qui deviendra toujours plus grand au fur et à mesure de la construction de bâtiment. Cependant, l'IA vérifie avant de construire que le terrain est apte. Ainsi, si un minerai ou un bâtiment est déjà présent elle passera automatiquement à la position suivante.
- L'Instanciation : Tandis que le joueur doit sélectionner un bâtiment pour lui demander d'instancier une unité, l'IA, elle, va demander une unité, et ensuite, elle déduira dans quel bâtiment il est idéal d'initialiser la tâche.
-Le Minage : Permet d'envoyer un constructeur miner. L'algorithme déterminera alors l'astéroïde ou la ferme optimale sur laquelle envoyer la troupe. Pour la déterminer, l'algorithme va attribuer un score à chaque bâtiment de ressource compatible et prendre celui qui à le score le moins élevé. Ce score est établi grâce à la distance entre la Station spatiale et le bâtiment de ressource ainsi que par le nombre de constructeurs déjà en train de miner dessus.

Le but de ces outils est de permettre à l'IA de contrôler sa partie, on peut imager ces outils comme le corps du Bot, et l'IA comme étant sa tête. Une fois ces outils terminés j'ai pu passer au développement de l'IA c'est à dire, les utiliser intelligemment.

2.4.2 IA

Avant de parler du fonctionnement de l'IA je vais m'attarder sur les réflexions et le chemin qui m'a mené à cette vision de l'intelligence artificielle.

Histoire :

N'ayant aucune expérience en intelligence artificielle, et face à la difficulté qu'est de développer une IA pour un jeu de stratégie nous avons voulu en faire une assez modeste en termes de faculté de raisonnement. En effet dans un jeu de stratégie l'IA doit savoir gérer beaucoup de chose, elle doit anéantir les équipes adverses mais pour cela elle doit d'abord développer sa base, gérer ses ressources, créer assez de constructeurs, découvrir de nouvelles ressources et construire tel ou tel bâtiment pour arriver à ses fins.
Ma première idée fut de mélanger scripts et réactions, en effet je n'avais aucune idée de la manière pour faire en sorte que l'IA gère bien ses ressources et fasse les bons choix. Concrètement cela devait se traduire par une suite d'action que l'IA allait devoir effectuer de manière scriptée avec quelques petites libertés. Les grandes lignes de son comportement devaient être scriptées, notamment la construction de tel ou tel bâtiment, tandis que d'autres actions comme la

création de nouveau constructeurs devait être automatiquement géré par l'IA. Le premier défaut de cette méthode est de faire durer cette IA dans le temps, tant qu'elle n'a pas gagné elle doit continuer d'améliorer sa base ainsi que de former des armées, mais si l'IA suit des actions scriptées elle se serait arrêtée tôt ou tard. Pour remédier à cela j'ai décidé que la suite d'action que devait effectuer l'IA serait très générique et qu'une fois que toutes les actions auraient été effectuées, l'IA allait recommencer à suivre les même actions (à quelques exceptions près). Cependant un défaut majeur persistait, il nous aurait fallu créer cette liste d'action. Cela est possible mais très compliqué, elle se devait d'être très bien pensée pour ne pas brider l'IA. Pour pallier cette difficulté, j'ai voulu donner plus d'autonomie à l'IA, lorsqu'elle aurait manqué de ressources, qu'elle détermine si elle a besoin de créer de nouvelle unité ou bien d'attendre ou bien de mieux les répartir, et lorsque ce cas se serait présenté, de reporter la tâche actuelle à la fin de la liste. Pour pousser ce principe encore plus loin j'ai voulu faire en sorte que, si l'IA voulait créer une unité, elle devait avoir construit le bâtiment adéquat. Cela est aisément automatisable et c'est alors que j'ai compris où cette automatisation allait me mener. En réalité il est assez simple pour l'IA de déduire d'elle-même ce dont elle a besoin, c'est ainsi que j'ai changé ma manière de voir les choses pour arriver à l'IA que je m'apprête à présenter.

IA :

Voici le fonctionnement de l'IA : en début de partie, un objectif lui est donné, celui de former une armée prédéfinie. L'IA va alors utiliser tous les moyens en sa possession pour accomplir cet objectif ; Si elle ne possède pas les bâtiments nécessaires elle les construira, mais pour cela elle a besoin d'un constructeur qu'elle créera alors. Si elle manque de ressources elle pourra ou non demander de nouveau constructeurs, ou bien mieux les répartir, ou encore attendre.
Le fonctionnement est simple, chaque tâche que l'IA doit effectuer est stocké dans une pile. Lorsque l'IA accomplie une tâche elle dépile pour effectuer celle qui sort de la pile. Si elle ne peut pas encore l'effectuer pour tel ou telle raison, elle va empiler de nouveau la tâche puis en créer une nouvelle qui répondra aux besoins de la tâche initiale.
Chacune de ces tâches se met en quelque sorte en erreur lorsqu'elles ne sont pas effectuables, cela est détecté par l'IA et, en fonction de l'erreur, une tâches adéquate sera créée. Une fois que l'IA aura terminé son armée, elle va détecter quelle est la base ennemie la plus proche et l'attaquer. Une fois l'assaut lancé, seule la mort des troupes l'arrêtera. En effet, si une armée détruit une base adverse elle ne rentrera pas mais attaquera la suivante jusqu'à la victoire ou

la mort. Pendant ce temps, dans la base, l'IA est déjà en train de préparer sa nouvelle armée au fur et à mesure que la précédente voit son nombre de troupes diminuer.

Pour l'heure les seuls éléments prédéfinis sont :
- Les armées : Nous définissons pour l'instant nous-même ce que l'IA devra créer comme troupes. Lorsqu'une armée est terminée, l'IA passe à la suivante, et une fois que la dernière a été créée, l'IA créera à nouveau cette même armée jusqu'à la fin de la partie. L'un des objectifs pour la soutenance finale est notamment de faire en sorte que les nouvelles armées de l'IA soit proportionnellement plus grande que la précédente tout en gardant les mêmes proportions d'unités. Ainsi nous n'auront qu'à définir ces proportions uniquement.
- Le nombres de bâtiment maximum : En effet, pour éviter que l'IA ne construise le maximum de maison en début de partie, ou bien qu'elle construise une nouvelle station de combat à chaque fois que les autres sont pleines, j'ai défini un nombre maximum pour chaque bâtiment, et ce nombre va augmenter à chaque que l'IA termine une armée. Cela permet de faire en sorte que la base de l'IA grossisse continuellement et régulièrement.

2.4.3 Combat

Pour cette soutenance je me suis aussi occupé du système de combat. Le principe est simple, mais deux difficultés majeures sont à prendre en compte, le comportement ainsi que le multijoueur.
Etant donné que les troupes sont des vaisseaux, il est normal que ceux-ci se défendent en se tirant dessus avec des projectiles. Chaque unité de combat observe dans un certain rayon autour d'elle, si une unité ennemie entre dans son périmètre elle l'attaquera si elle ne fait rien d'autre. Le comportement des unités est délicat lorsqu'une ennemie approche, doit-elle l'attaquer ? continuer ce qu'elle fait ? fuir ? Pour l'heure elle se contentera de riposter si elle ne fait rien et sinon continuera ce qu'elle fait.
Lorsqu'une unité en attaque une autre elle va instancier à intervalle régulier des projectiles vers l'ennemi qui lui feront des dégâts s'il le touche. Les troupes restent toujours face à leur ennemi et en cas de fuite le suivent tant que la distance entre les deux est suffisamment petite. La fréquence de tir, ainsi que les dégâts par projectile varient selon les unités, et selon l'arbre de compétence. Une de mes difficultés a été de gérer cela en multijoueur, tout d'abord il n'est pas très optimisé d'instancier les projectiles de chaque troupe en multijoueur et de suivre leurs positions, notamment lors de grandes batailles. Ainsi, étant donné que les balles ne se déplacent que dans une direction, et qu'elles n'en

changent jamais, j'ai fait en sorte que lorsqu'une balle est instanciée chez un joueur elle le soit chez les autres mais que le joueur propriétaire de la balle communique aussi les force et la direction de la balle aux autres joueurs qui la feront alors se déplacer localement chez chacun. Ceci n'a qu'un but purement visuel puisque c'est le joueur propriétaire qui détruira ou non la balle pour tous les autres lorsque celle-ci touche un joueur ennemi. Le désavantage de cette méthode à l'inverse de celle qui consiste à synchroniser les positions en ligne et que chez les autres membres de la partie est que cette balle peut ne toucher aucune unité visuellement mais pourtant faire des dégâts à cause de la latence en multijoueur. Cependant, si la position de la balle était synchronisée en ligne, cela consommerait une telle bande-passante que la latence serait encore plus élevée.

2.4.4 Débogage

Une très grande partie de mon temps de travail a été dédiée à la correction de bogues notamment avec l'arrivé de l'IA et du système de combat et d'autant plus lorsqu'il s'agit du multijoueur.

2.4.5 Améliorations diverses

Beaucoup de détails ont été améliorés ou ajoutés. Ils ne sont ni indispensable ni très compliqués, il s'agit en général de messages d'aide dans les menus au cas où le joueur n'aurait par exemple pas mis de nom pour sa partie ou bien s'il ouvre le panneau des alliés, ou encore un petit message « *waitting for players...* » dans la salle d'attente pour rappeler au joueur qu'il est bien dans la salle d'attente ainsi que son but.

2.4.6 Travail en groupe

Nos tâches ayant été bien réparties, nous pouvons travailler un maximum chacun de notre côté sans avoir besoin des uns ou des autres. Nous organisons régulièrement des réunions pour faire des points sur ce qui a été fait et reste à faire ainsi que pour discuter de la direction que va prendre le jeu, on débat de la manière d'implémenter certaines fonctionnalités, en somme, on définit ce que l'on veut que le jeu soit.
Concrètement cela fonctionne plutôt bien il n'y a jamais eu de différents majeurs et nos différents domaines d'expertise nous permettent de nous faire confiance les uns les autres.
D'un point de vue technique nous échangeons sur le logiciel Discord et nous avons initialisé un dépôt sur GitHub.

2.4.7 Prévision pour la Soutenance Finale

Quasiment toutes les fonctionnalités du jeu ont été implémenté, ainsi pour la prochaine soutenance je prévois surtout de la correction de bogue et diverses petites améliorations qui tenteront d'élever ce jeu au niveau des standards actuelle du secteur. L'idée est que si quelqu'un lance le jeu, il ne puisse pas se douter qu'il s'agit d'un jeu indépendant développé par 4 étudiants pendant 6 mois seulement dans le cadre d'un projet éducatif.

3 Conclusion

Au long de la période qui a séparé la première soutenance de la seconde, les majeurs ajouts concernent l'arrivée de l'intelligence artificielle dans la directions des *bots*, ces personnages non joueurs gérés par ordinateur, ainsi que celle des modèles 3D des unités. De très nombreux ajouts autres que ceux cités sont aussi désormais dans Year Zero. Ils sont plus difficilement visibles considérant qu'ils relèvent du détail ou de l'optimisation mais contribuent tout autant à l'atmosphère que nous souhaitons créer dans notre jeu. Ainsi nous sommes à jour du point de vue du planning mais il reste quand même beaucoup à faire pour avoir une version jouable avec deux camps distincts. Nous nous emploierons à finir ce que nous avons prévu lors de la rédaction du cahier des charges pour la troisième et dernière soutenance afin que nous puissions fournir une version stable, jouable et agréable.