Rapport De Soutenance 1 - Year Zero

Bearth Studio

13 mars 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 sera 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 sera assez épique avec de grands thèmes entraînants. Mais il va aussi jongler avec des thème plus calme et mélancolique 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.

Pour cette première soutenance nous avons tenu à ce que d'importantes bases du jeu soient prêtes afin de ne laisser que les parties artistiques et l'intelligence artificielle pour les autres soutenances.

2 Développement sur les tâches réparties

2.1 Timothée

Mon travail sur ce projet se répartit sur plusieurs domaines du jeu qui sont divers. Il s'agit majoritairement du tutoriel du jeu, des aspects sonores et l'aide à la conception du site internet.

2.1.1 Tutoriel

Le tutoriel consiste en une partie scénarisée dans laquelle le joueur doit accomplir les actions demandées pour avancer tout en comprenant et apprenant les mécaniques du jeu. La première difficulté qui se pose dans ce genre d'exercice est de se mettre à la place du joueur et pour cela oublier le plus possible ses connaissances et habitudes par rapport au jeu sur lequel on joue, voir même par rapport au genre tout entier. De même il faut arriver à lui faire comprendre les actions qu'il doit effectuer et pour cela nous avons décider d'afficher des messages et de mettre en pause la partie le temps que le joueur lise et puisse avancer à son rythme avec sa souris. Nous avons choisi de faire un tutoriel simple qui explique seulement les bases concises du jeu afin de laisser une certaine expérience de découverte et de faire quelque choses de très léger qui puisse au besoin être rejoué très vite pour se remémorer les fonctionnements principaux des unités et bâtiments. Du côté technique cela a consisté principalement en une utilisation des outils mis en place pour le fonctionnement du jeu et à la façon de les relier entre eux par un script qui exécute chaque étape à effectuer. Ce qui nécessite de la communication avec Axel pour la compréhension et parfois l'adaptation du code, mais aussi les ajustements graphiques d'Enguerrand qui permettent au jeu de rester toujours cohérent esthétiquement. C'est donc notre capacité à travailler à quatre sur les mêmes éléments qui est mise en jeu à ce moment là.

2.1.2 Site internet

De mon côté je n'ai jamais pris part à la conception d'un site internet, j'ai donc particulièrement travaillé sur l'aspect du rendu et des fonctionnalités avec Nicola qui possédait déjà des compétences dans ce domaine.

2.1.3 Doublage

L'ensemble de musiques et de sons qui seront parties intégrantes du jeu notamment pendant les parties et les différentes interactions du joueur et les

actions des unités elles mêmes devront être développées pour la prochaine soutenance en même temps que les graphismes internes du jeu. De plus les unités posséderont des voix qui répondent aux ordres donnés. Par exemple lors d'un ordre d'attaque donné à un vaisseau on entendra "à l'attaque". De plus les textes et dialogues de la campagne seront doublés pour vraiment créer un récit avec une narration correspondant au scénario de Year Zero.

2.2 Nicola

Je vais ici présenter mon travail réalisé sur ce projet, là première chose qui ma été donnée de faire à été le scénario, en effet même si le thème et l'ambiance du jeu avait déjà été décidée en amont, le scénario permet de développer le jeu dans une seule direction et d'obtenir une certaine harmonie entre le style général choisi,la campagne et le site.

2.2.1 Site internet

N'étant pas spécialisé dans la création de sites Internet, j'ai décidé de me servir de l'outil Wix afin de fournir un résultat propre. J'ai tenu à garder le thème du jeu dans le site internet, j'ai donc réutilisé une des polices dominantes de notre jeu et l'image de fond du menu. J'ai ensuite suivi le plan proposé dans le Dossier Projet, et donc incorporé les liens de téléchargement du Cahier des charges, du rapport de soutenance et de la version "lite" du jeu, et une présentation complète de membres de Bearth Studio. Compte tenu du fait que le site n'est obligatoire qu'a partir de la soutenance 2, ce n'est ici qu'une esquisse du site, il se verra sûrement remodelé par la suite.

2.2.2 Tutoriel

Le tutoriel à pour but de présenter les mécaniques de bases du jeu, il pose l'ambiance du jeu, et donne les principales clés de victoire d'un RTS. Pendant celui-ci, le joueur se verra présenter comment sélectionner les unités/bâtiments mais également les bâtiments de base et leurs utilités, l'utilisation des Builder qui est l'unité de construction/récolte, les unités de combat et comment les créer, comment récolter des ressources, comment les utiliser et enfin le fonctionnement du SkillTree, élément original au centre du gameplay de Year Zero. Afin de forcer le joueur à écouter les consignes du tutoriel, nous avons du limiter les actions que celui-ci peut réaliser, et vérifier que les demandes du tutoriel sont satisfaites avant de continuer. Le tutoriel, étant l'une des premières expérience du jeu que nous avons pu avoir, cela à permis de faire ressortir certains bugs ou mécaniques

de jeu incomplètes voir absentes, mais également certaines implémentations qui seront nécessaire pour continuer la campagne.

2.2.3 Scénario

Dans un avenir quoique lointain, l'humanité a enfin su surmonter les difficultés qu'elle a rencontrées ; guerre, pollution, crise énergétique, surpopulation... Le monde vivait en paix. Mais rien n'est éternel, et la soif de l'homme pour la conquête spatiale et la découverte de vie extraterrestre s'est retournée contre lui. En retournant d'une longue mission aux abords du système solaire et après avoir visité nombres de lunes et de planètes, une équipe d'astronaute aurait ramené un parasite extraterrestre, prenant contrôle du corps de son hôte, n'en faisant plus qu'une coquille vide, exempte de tout désir, ou plus précisément, que tout désir qui aurait pu les animer précédemment. Les personnes infectées ne pensais plus qu'à répandre le parasite dans le plus d'hôte possible, et une fois leur nombre suffisant, ils ont pris les armes. Il ne faut pas croire que l'humanité est restée les bras ballants devant leurs camarades malades. Mais le parasite se propageait très vite, et les ressources sont rapidement devenus limitées. Et le seul remède qui a pu être développé était un vaccin, empêchant le parasite de s'implanter dans l'hôte mais impuissant face aux infectés. Un grand pas un avant face à cette menace, mais un peu tardive, une grande partie de l'humanité était déjà sous l'influence du parasite, il ne restait plus qu'une poignée de résistants, prêt à tout pour récupérer leurs terres, pour lutter contre leurs frères, qui aujourd'hui se comportaient comme de vulgaires fourmis. Mais chaque jour l'humanité reculait, jusqu'au jour où ils ont été forcé de quitté la terre, obligés de survivre dans les derniers vaisseaux spatiaux de l'humanité. Aujourd'hui, l'humanité est mise à genoux et les habitants actuels de la terre les pourchassent et recréent peu à peu leur arsenal militaire détruit pendant la guerre. L'humanité ne peux plus se laisser faire, c'est, armé jusqu'aux dents, dans des vaisseaux dignes des meilleurs science-fictions, qu'une poignée d'humains, vont devoir se battre, pour un jour, retourner sur terre et marquer le début d'une nouvelle ère, Year Zero.

2.3 Enguerrand

Mon travail jusqu'à présent n'a été essentiellement fait que sur Photoshop. Considérant que l'aspect graphique et artistique du jeu constitue la majeure partie de mon travail sur Year Zero, je me suis donné les moyens de réaliser de belles choses, s'intégrant au mieux dans l'univers que nous avons créé. Outre la création artistique sur Phosothop, il m'a fallu également ordonner tous les

éléments de notre jeu au sein de Unity afin que le rendu visuel soit agréable et plaisant, ainsi qu'ergonomique et pratique. Pour la prochaine soutenance, je m'attarderai plus en détails sur l'importations des modèles 3D des diverses unités et bâtiments, ainsi que sur la génération de la map et le système de combat.

2.3.1 Création des logos

Responsable de la partie visuelle du jeu, je me suis penché en premier, au moment de rédiger le cahier des charges, sur la conception d'un logo pour le jeu, ainsi que pour le groupe. Une sorte de studio de développement et d'édition de jeu vidéo. Déjà présentés lors du rendu du cahier des charges, les logos de Year Zero et de *Bearth Studio* contiennent majoritairement les couleurs rouge, bleu et violet dans diverses teintes. Voulant donner un aspect spatial pour se rapporter au thème principal du jeu, les logos sont devenus référence dans toute la conception future dans ce qui concerne le design du jeu.

2.3.2 Conception de l'identité graphique

Hormis les logos, qui ne sont qu'une pièce moindrement visible que l'interface du jeu par exemple, il a fallu déterminer les lignes directrices de la direction artistique de Year Zero afin de conserver les mêmes couleurs, formes etc. Le tout, dans la prétention de vouloir créer une atmosphère entraînante et immersive. C'est donc dans des tons de couleurs chauds, se rapprochant du rouge, du violet et du bleu que Year Zero prendra forme. La police que nous utiliseront majoritairement est *Orbitron Black* pour la plupart des textes et pour certains titres nous utiliserons Space Age.

FIGURE 1 – Police Orbitron Black

FIGURE 2 – Police Space Age

2.3.3 Menu Principal

C'est vers le menu principal que je me suis penché en premier. Au menu déjà existant, constitué de boutons sans texture, et du fond par défaut de Unity, j'ai ajouté un fond qui est une image dans l'espace où l'on voit des nébuleuses. J'ai ajouté en bas a gauche et à droite respectivement les logos de Year Zero et de Bearth Studio. Enfin j'ai créer sur Photoshop le logo titre Year Zero que j'ai importé dans Unity afin de le mettre au dessus des boutons.

FIGURE 3 – Menu principal

N'ayant pas d'idées précises de comment j'allais designer les boutons, j'ai tenté quelques versions avec diverses couleurs, toujours en créant une image de bouton sur Photoshop qui allaient se superposer au bouton Unity.

FIGURE 4 – Test de certains boutons

Au final, j'ai créé une nouvelle texture pour les boutons que j'ai importé. Il s'est avéré qu'elle collait bien au fond et avec le reste, ainsi on l'a gardée.

J'ai décliné cette même texture en plusieurs versions afin de s'intégrer au mieux dans toutes les scènes de notre projet Unity et dans toutes les situations ou nous allions en avoir besoin.

FIGURE 5 – Texture du bouton

FIGURE 6 – Bouton pour une barre de remplissage

FIGURE 7 – Bouton carré aux bords arrondis

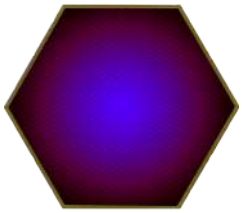

FIGURE 8 – Bouton en forme d'hexagone

FIGURE 9 – Bouton rectangle aux bords arrondis

Finalement, c'est vers des boutons en forme d'hexagone que je me penche, voulant innover un peu de sorte que nous n'ayons pas des boutons qui semblent être seulement par défaut ceux de Unity. Au lieu de marquer où ces boutons redirigent, je me suis alors dit que c'était probablement plus explicite et aussi plus agréable visuellement de mettre des icônes au lieu du texte. Nous avions alors un menu qui commençait à prendre forme.

FIGURE 10 – Menu principal

J'ai aussi rajouté pour un effet d'immersion au sein du menu, deux couches d'étoiles matérialisées par deux images contenant du transparent et des étoiles. Je leur ai ajouté une animation de sorte qu'elles tournent sur elles-mêmes dans s'arrêter. J'ai aussi créer via un petit script, un effet de parallax (que l'on retrouvera plus tard). La parallax est une façon de rendre de la profondeur et de la distance. Ici, il s'agit de faire suivre les mouvements de la souris dans le menu principal à l'image de fond mais avec un léger coefficient. L'effet est très léger ici, il n'est que visuel.

2.3.4 Écran de chargement

Outre la scène du menu principal, il y a une scène qui n'est que rarement visible puisqu'elle se passe très vite. C'est la scène de l'écran de chargement. Elle est simple dans la mesure ou elle n'est constituée que du même fond que le menu principal, ainsi qu'une barre de progression et d'un message qui affiche *LOADING...*

2.3.5 Animations des boutons

Rajouter des animations aux boutons était nécessaire pour créer du dynamisme dans la navigation au sein des menus. L'animation est la même pour tous les boutons du menu principal et de ses sous-menus. Elle consiste au grossissement du bouton. Ainsi quand la souris passe sur le bouton, ce dernier voit

sa taille augmenter légèrement.

2.3.6 Écran de jeu

Environnement global

Lorsqu'on arrive dans une partie, on est dans l'espace. Il est nécessaire d'avoir un environnement adapté. Ainsi le sol d'où bougent les unités est englobé dans une *Skybox* ce qui correspond à une sphère dont la texture est une image, ici de l'espace. Le rendu fait en sorte de respecter une sorte d'immersion dans cette sphère et donne l'impression que la profondeur est infini.

Effet de parallax

Lorsqu'on déplace la caméra dans la partie, on n'a d'impression de mouvement que si on voit défiler des unités. C'est pourquoi j'ai rajouté plus de quatre couches d'étoiles en dessous du sol, toutes à des hauteurs différentes de sorte que lorsqu'on se déplace, les étoiles suivent plus ou moins le mouvement en fonction de si elles sont loin du sol ou pas.

Inteface

Le plus gros de mon travail jusqu'à la soutenance présente a été la conception de l'interface dans une partie. Tout a été fait sur Photoshop puis importé dans Unity. J'ai réalisé en une image qui constituera l'interface globale dans le jeu.

FIGURE 11 – Image de l'interface

FIGURE 12 – Interface en jeu

De la même manière que pour le menu principal, les boutons d'actions des unités ont, au lieu d'un texte, une icône, ainsi que la texture d'un bouton carré (*cf : Figure 7*).

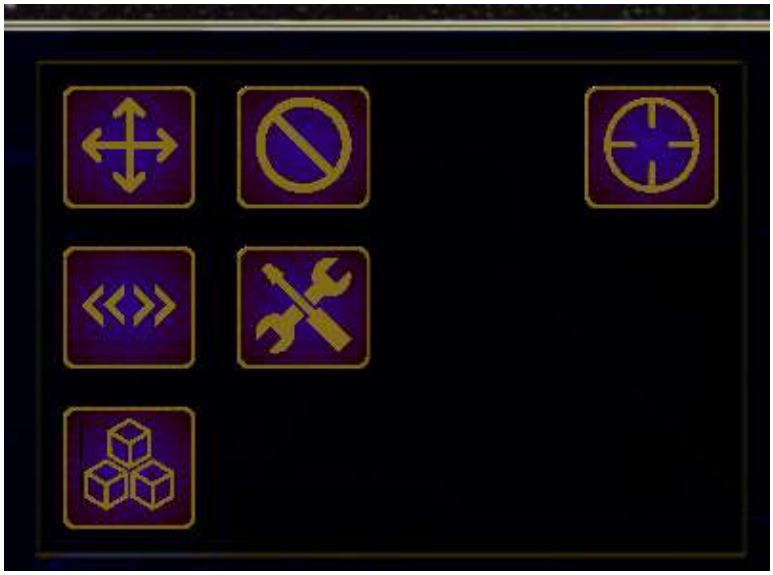

FIGURE 13 – Boutons avec icônes

On retrouve également les boutons de navigation au dessus de l'écran. La texture qui leur est associée est le bouton rectangulaire à bords arrondis. Le texte a été laissé pour une meilleure compréhension de leur action et la couleur a été mise sur un doré, jaune qui se porte bien au reste de l'interface.

FIGURE 14 – Boutons avec icônes

A leur droite, on retrouve un bouton pour l'arbre ce compétence, avec son icône associée. A sa droite également, on retrouve les indicateurs de ressource. A l'origine, ce sont des boutons mais on ne peut pas interagir avec eux, d'où leur couleur plus sombre que les autres. La couleur du texte et des chiffres a aussi été mise sur le doré, jaune.

FIGURE 15 – Boutons des ressources

Sont présents également les boutons de la minimap, formés eux aussi de leur texture et d'une icône, comme pour toutes les autres faite par mes soins sur

Photoshop. On retrouve aussi le bouton de constructeur inactif constitué de la même manière. Tous ces boutons possèdent la même animation, qui change leur couleur en fonction de leur état. Les quatre états distincts étant : *Highlighted* si la souris est sur le bouton, *Pressed* si la souris a cliqué sur le bouton, *Disabled* si on ne peut pas interagir avec le bouton ainsi que l'état par défaut.

FIGURE 16 – Bouton survolé par la souris

FIGURE 17 – Bouton cliqué

2.4 Axel

Je vais ici présenter chaque fonctionnalités que j'ai ajouté dans le jeu, mon ressenti, les problèmes survenus ainsi que mes prévisions pour les prochaines soutenances.
Tout d'abord afin de structurer correctement ma partie, je ne vais pas présenter mon travail selon un ordre chronologique mais plutôt par section. Les différents systèmes étant entremêlés les uns aux autres je n'ai pas pu tous les développer les uns à la suite des autres mais en construisant le tout petit à petit. Je vais cependant garder un ordre le plus chronologique possible.

2.4.1 Contrôles

Sélection

Au-delà de créer le projet sur Unity, ma première tâche fut de développer un système de sélection des unités. Le principe est un simple, un clic gauche sur une unité la sélectionne, mais à cela s'ajoute la sélection de plusieurs unités qui survient lorsque qu'on effectue un clic gauche sur la souris et qu'ensuite on la déplace pour former un rectangle permettant de sélectionner l'ensemble des unités dans ce rectangle. J'ai ensuite donné quelques spécificités à cet algorithme, par exemple, si on sélectionne un bâtiment on ne peut en sélectionner qu'un seul, si on effectue un double-clic gauche on sélectionne toutes les unités à l'écran qui sont du même type que celle sous le pointeur de la souris, ou encore, si on dessine un rectangle autour d'unités qui ne nous appartiennent pas on n'en sélectionnera finalement qu'une seule. Aussi, un clique droit sur une unité va exécuter l'interaction adéquat. Cet algorithme est un des plus complexe qu'il m'ait été donné de développer dans le cadre de ce projet tant il prend en compte de paramètres, j'ai dû le penser et le repenser continuellement au fur et à mesure que je lui ajoutais des fonctionnalités.
Ma première difficulté fut de savoir comment je pourrais sélectionner l'unité sous le pointeur de la souris. Après quelques recherches, j'ai découvert le principe du Raycast, cette fonctionnalité d'Unity permet d'envoyer un rayon depuis un point et suivant un vecteur direction défini et renvoie l'objet rencontré par le rayon. J'envoie donc un rayon depuis la position de la souris et dirigé vers le point juste en dessous de la souris pour accéder à l'unité et ainsi la sélectionner. Le système de multi-sélection est plus simple mais aussi beaucoup plus gourmand, il va parcourir la liste des unités et transposé leur coordonné 3D en 2D puis les sélectionner si elles sont dans le rectangle dessiné.

Player control system

Au fur et à mesure que j'ajoutais des fonctionnalités au jeu, il m'est apparu qu'il fallait faire un système pour gérer différents profiles de contrôles, par exemple lorsqu'on veut placer un marqueur sur la carte on appuie sur le bouton associé et on passe alors dans un nouveau profile de touche très simplifier dans lequel un clic gauche sur la minicarte va poser le marqueur et un clic en dehors de la minicarte ou bien un appuie sur la touche ECHAP va annuler le mode marqueur pour repasser aux contrôles classiques. Sans ce système il serait compliqué d'isoler certaines fonctionnalités et par exemple ouvrir le menu pause n'empêcherait pas de sélectionner des unités ou de déplacer la caméra ce qui pose un problème.

Formation System

J'ai développé cet algorithme assez récemment car il ne faisait pas partie de nos priorités mais son utilité est indiscutable tant d'un point de vue technique que pour le joueur. En effet il s'agit là d'un système permettant de déplacer ses unités en formation, pour l'instant seul la formation rectangulaire est implémentée. Ce système évite que toutes les unités essayent de rejoindre le même point lorsqu'on les fait se déplacer car techniquement une seule va l'atteindre avant les autres et va ainsi empêcher par se présence les autres de rejoindre ce point menant ces unités à toutes se bousculer à l'infini pour atteindre leur destination. L'algorithme est assez simple et défini pour chaque unité une destination différente. Le seul problème auquel j'ai dû faire face était le cas ou on ne sélectionnait pas un multiple de 5 unités, en effet les rangés étant de 5, la formation était relativement laide et surtout si on sélectionnait une seule unité elle n'allait pas au point désiré mais quelque centimètre au-dessus car elle commençait la formation. J'ai donc modifié l'algorithme de sorte qu'au lieu de former une ligne dans l'ordre 1 2 3 4 5, il la forme dans l'ordre 5 3 1 2 4.

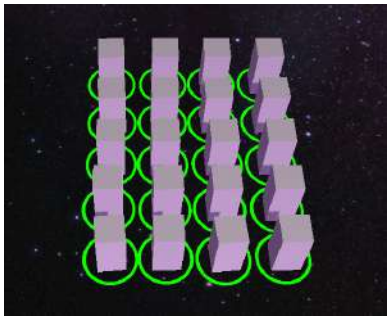

FIGURE 18 – Formation

Camera controller

Il est un des premiers algorithmes que j'ai implémentés car il est essentiel et simple. Par défaut la caméra est orientée à 60 ˚vers le bas pour avoir une vue assez générale de la partie mais offrant tout de même une certaine profondeur. La suite est simple, si on appuie sur les classiques touches Z Q S D ou bien que l'on place la souris sur les bords de l'écran on va pouvoir déplacer la caméra. Une autre fonctionnalité implémentée plus tardivement permet grâce la molette de la souris de zoomer en quelque sorte la caméra, en effet celle si va se rapprocher du sol et réduire son angle rotation de sorte à se retrouver quasiment à l'horizontale. Cette vue est plus esthétique qu'autre chose car elle n'est pas très pratique mais offre une vue plus cinématographique et épique sur la partie.

2.4.2 Managers

J'ai nommé manager toutes les classes comportant un singleton. Ce singleton permet de limiter le nombre d'instance de cette classe à une seule et de la rendre accessible par absolument tous les autres scripts simplement. Ces algorithmes sont en général primordiaux et demande un accès récurrent de la part d'autres script.

Chat Manager

Ce programme implémenté assez tardivement car non essentiel permet de gérer de manière générale tous les messages envoyés par les joueurs. La partie visuelle de cet algorithme sera détaillé plus loin dans le rapport. Concrètement, à chaque fois qu'un joueur envoie un message il sera stocké ici grâce à 2 informations, le joueur qui l'envoie et son contenu. Puis il sera ensuite envoyé à tous les autres joueurs de la partie grâce à un RPC (voir partie sur le multijoueur).

Instance Manager

Celui-ci permet de faire le lien entre diverses actions locales et des informations multijoueur. En début de partie, il va récupérer les informations du joueurs actuelle grâce à ses custom properties (voir multijoueur) définies dans la salle d'attentes (équipe, race, couleur, coordonnés de départ). Puis il va instancier les troupes de départ aux bons coordonnés. Ce manager gérera ensuite la notion d'échec qui surviendra lorsqu'un joueur n'a plus d'unité ou bien le cas ou un joueur se déconnecte entrainant la déconnexion de tous les autres.

Player Manager

Ce manager va gérer tous ce qui attrait aux ressources, c'est en effet ici qu'on ajoute ou retire de la ressource et qu'on trouve les fonctions permettant de payer des constructions ou de la production d'unités. Il va aussi gérer les différentes stations spatiales de sorte que les unités comme les Constructeurs puissent ramener leurs ressources à la station spatiale la plus proche.

2.4.3 Jeu

Gameresource

Une Gameresource est tout simplement une ressource, les minerais, l'énergie et la nourriture en sont. Elle contient simplement son nom, sa quantité et diverses méthodes pour modifier ses valeurs.

Population

Cette GameResource particulière gère la population. Elle est constituée d'une population maximum actuelle et d'une population actuelle. Si la population actuelle est égale à la population maximum actuelle alors on ne peut plus créer d'unité mobile. Pour augmenter ce maximum il faut construire des maisons.

Placement
Il s'agit là du système de placement de bâtiments. Il est décomposé en 3 sous systèmes :

Raycaster :
Un Raycaster va envoyer un Raycast vers le bas et dire s'il rencontre ou non le sol, si non ou bien si le sol est soit trop proche soit trop loin alors une variable booléenne va signaler que la case sur laquelle se trouve le Raycaster n'est pas disponible pour la construction.

DetectionCell :
Celle-ci est composé de 5 Raycaster disposé comme les points du cinq sur un dé de sorte à pouvoir tester la case assez précisément. Une DetectionCell représente une case de la Carte et va observer chacun de ses Raycaster, si au moins un seul d'entre se présente comme non disponible à la construction, la case au départ de couleur verte va passer rouge.

PlacementGrid :
Une PlacementGrid va être généré lorsqu'on veut poser un bâtiment, elle va alors obtenir la taille du bâtiment en case et va générer autant de DetectionCell que le bâtiment prend de case. Si au moins une seule des DetectionCell est au rouge alors la PlacementGrid empêchera le joueur de poser son bâtiment. Le centre de la PlacementGrid correspond au pointeur de la souris mais avec le système de case.

Ce système a demandé une certaine réflexion avant d'être développé, j'aurais pu me contenter d'un algorithme empêchant simplement de poser deux bâtiments l'un dans l'autre mais j'avais plus d'ambition et voulais un système qui montre au joueur quelle(s) case(s) pose(nt) problèmes. Étant dans l'espace la distance entre le sol et le Raycaster est inutile mais le système est là et bien fonctionnel. D'autre part le système de case est un artifice. Il n'y a pas de case à proprement parler mais simplement un calcul qui va créer un modulo des coordonnés sur lesquels on peut placer un bâtiment. Par exemple si le modulo est de 5, et que le joueur place son pointeur en $x = 12$ alors au lieu que le centre de la PlacementGrid soit en $x = 12$ il sera en $x = 10$, si le pointeur est en $x = 13$ celui de la PlacementGrid sera en $x = 15$ et ainsi de suite.

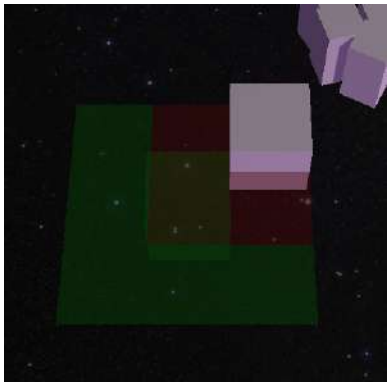

FIGURE 19 – Système de construction

Tasks

Le système de Tasks a été développé spécialement pour les bâtiments afin de leur donner la possibilité d'exécuter des tâches comme son nom l'indique. Au départ je voulais créer plusieurs types de tâches, celles de production et celles d'amélioration. En fin de compte seule la première a été implémenté, l'autre étant devenu l'arbre de compétences. Une tâche de production permet simplement de produire une unité. Chaque tâche se paye en ressource et prend un certain temps pour s'effectuer.
Bien que simple à expliquer, cet algorithme m'a pris pas mal de temps car il mêle beaucoup de système précédemment développer, il a fallu corréler le tout et évitant de créer des bugs. Le tout en mettant bien à jour l'interface afin de lui donner les bonnes informations.

2.4.4 menus

Main Menu et In-Game Menu

Une fois le cœur du jeu bien amorcé j'ai voulu incorporer le multijoueur rapidement, mais il me fallait d'abord créer des menus pour cela. Étant donné leur nombre conséquent j'ai dû créer un algorithme très général me permettant de simplifier au maximum l'implémentation de nouveau menu. En effet, au-delà de faire les choses correctement pour le joueur, il faut aussi créer des outils assez puissant et intelligent pour faciliter le travail du développeur.

Menu principal :
Permet de lancer le mode solo, multijoueur, d'aller dans les options, de voir les crédits, ou encore de quitter le jeu. Menu solo :
Permet de choisir entre le mode campagne (encore absent) et le mode partie rapide.

Menu multijoueur :
Permet de choisir entre rejoindre une partie ou en créer une. Pour rejoindre une partie il suffit de taper son nom et de cliquer sur Join.

Menu de création de partie :
Permet de créer une partie en choisissant la carte, et si le joueur créer une partie multijoueur de donner un nom à la partie ainsi que définir le nombre maximum de joueur (IA comprise).

Menu Options :
Permet de régler certains paramètres. Il est trié en plusieurs catégories :
-Gameplay : Permet de définir la vitesse de défilement de la caméra avec le clavier et la souris, de désactiver le défilement à la souris et d'activer ou non les bulles d'aides. -Pseudo : Permet de choisir son pseudo
-Vidéo : Permet de choisir la résolution, d'activer ou non le plein écran, et de choisir le niveau de qualité graphique du jeu.
-Son : Permet d'ajuster le volume du jeu

J'ai reproduit peu ou prou le même menu durant la partie, ils manquent seulement quelques options comme le changement de pseudo parce qu'il est logique qu'il soit impossible de le changer durant une partie.

AlliesMenu
Le menu des allies permet d'envoyer des ressources à ses alliées. Il utilise le système de RPC (voir multijoueur).

loading times
Les temps de chargement ont été ajouté récemment et demeurent bien inutile pour l'instant. En effet, le jeu n'étant constitué que de polygones simples et de textures unies, les temps de chargement sont à peine perceptibles, mais le système est bien là et opérationnel. Il utilise la faculté d'Unity à pouvoir charger une scène tout en maintenant l'actuelle utilisable, et pour ce qui est de la barre de progression du chargement, Unity nous fournit une variable entre 0 et 1 donnant cette progression.

2.4.5 Interface

Dans cette sous-partie je vais expliquer tout ce qui attrait à l'interface durant la partie.

Cards

J'ai nommé Carte les petites fiches de chaque personnage sélectionné. Le panneau qui les contient permet alors d'affiche pour chaque unité une image de son modèle ainsi que sa jauge de vie pour avoir une vue d'ensemble sur les troupes sélectionnées. Son utilité n'est pas seulement visuelle puisque cliquer sur une de ces cartes permet de sous-sélectionner une unité. Cela signifie que les actions disponibles s'effectueront sur l'unité sous-sélectionné et non l'ensemble des troupes.

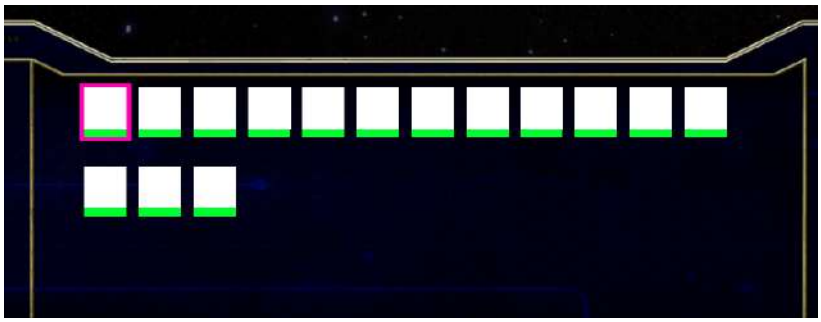

FIGURE 20 – Cards

Chat

La partie visuelle du chat est découpé en 2 panneaux : Le premier ne propose aucune interaction, il est situé sur la gauche de l'écran de jeu et affiche les messages envoyés mais ne les gardes qu'un temps, au bout de quelques secondes les messages disparaissent. Le second est accessible depuis un menu et affiche quant à lui tous les messages envoyés sans jamais les supprimer. Il offre une zone de texte pour pouvoir envoyer des messages.
L'autre méthode pour envoyer des messages est d'appuyer sur entrée dans le jeu ce qui fait apparaître une zone de texte pour écrire son message.
Ces 2 panneaux utilisent le Chat Manager qui sert de banque de message en quelque sortes.

advancement bar

Il s'agit simplement d'une barre de progression qui survient lorsqu'on sélectionne un bâtiment en construction et qui donne l'avancement de celle-ci.

description panel

Apparait lorsqu'on passe la souris sur un bouton pour construire un bâtiment ou instancier une unité. Il contient le nom de l'objet à créer, son coût en ressource ainsi qu'une petite description de l'objet.

FIGURE 21 – Description Panel

Help

Il s'agit d'un petit panneau qui va afficher l'utilité de certains boutons lorsqu'on passe le pointeur de la souris dessus car certains boutons utilisent des icônes qui peuvent prêter à confusion. Ce panneau est désactivable dans les options. Il survient notamment lorsqu'on passe la souris sur les icônes de la minicarte.

FIGURE 22 – Bulle d'aide

jobless button

Ce bouton sert à indiquer si oui ou non certains constructeurs son inoccupés, et si oui alors un clic sur ce bouton sélectionne cette unité et déplace la caméra au-dessus de celle-ci.

Monodescr

Ce panneau survient lorsqu'on sélectionne une unité et n'affiche pour l'instant que son nom, le but final étant d'afficher d'autres informations spécifiques à cette unité. Pour l'heure on ne voit que la quantité et le type de ressource transporté par un constructeur.

Portrait

Le portrait affiche la vie de l'unité sous-sélectionné et ainsi que son modèle. Il ne s'agit pas là de l'affichage d'un modèle issue des Assets du jeu mais d'une caméra qui va se place devant l'unité sous-sélectionnée, le portrait affiche tout simplement ce que voit cette caméra.

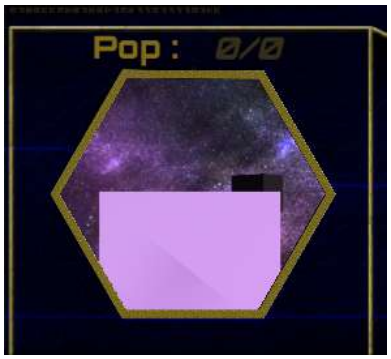

FIGURE 23 – Portrait

Resources

Le panneau de ressource va générer au lancement du jeu autant de panneau affichant le nom et la quantité de ressources que de ressources ajoutées au jeu.

Tools

Le panneau des outils et une des fonctionnalités les plus importantes du jeu. Un outil (tools) désigne ici bouton associé à une unité et qui va effectuer une action avec cette unité. Par exemple : Celui pour déplacer l'unité, pour l'arrêter, pour afficher le menu de construction, pour instancier une unité, pour aller réparer un bâtiment et ainsi de suite.
Ce panneau allant servir très souvent il se devait d'être bien codé, pour être assez puissant mais aussi facile à utiliser pour le développeur. Dans les faits, pour certains outils particuliers il faut créer des scripts particuliers mais pour les unités il suffit de glisser déposer le préfabriqué de cette unité dans la liste de tools d'une autre unité pour qu'elle puisse générer un tools qui créer une Task

d'instanciation.

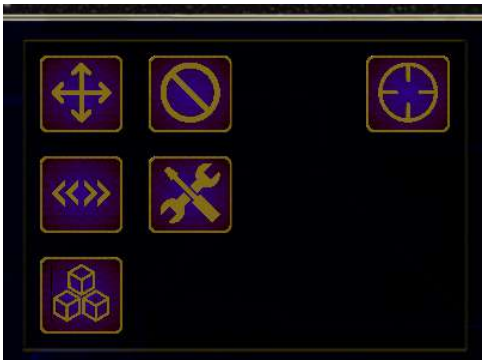

FIGURE 24 – Outils (Tools)

floating life bar
Pour chaque unité instanciée, une floating life bar va être généré et va faire apparaitre la vie d'une unité lorsqu'on passe la souris dessus.

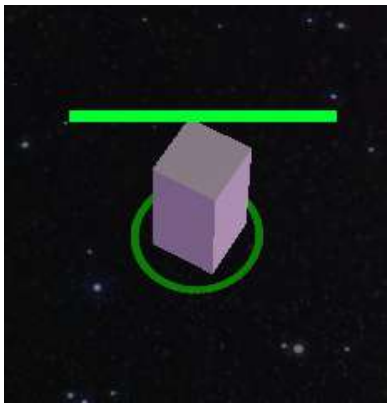

FIGURE 25 – Floating Life Bar

minimap La minimap (ou mini-carte) est en réalité une simple caméra placée au-dessus du jeu et qui en donne une vue d'ensemble. Chaque unité se voit attribué une icone qui ne sera affiché que sur la caméra. Elle contient par ailleurs d'autres fonctionnalités :
-Si une unité est dans le brouillard de guerre elle ne s'affichera pas sur la minimap

-Un clic gauche sur la minimap déplace la caméra de sorte qu'elle regarde l'endroit où le pointeur est
-Un clic droit déplace à l'endroit pointé sur la minimap
-Un rectangle indique la zone observée par la camera
-Un bouton permet de placer un marqueur sur la minimap, tous les joueurs de l'équipe le verrons aussi
-Un bouton permet d'afficher ou non le fond de la minimap
-Un bouton permet d'afficher ou non les unités
-Un bouton permet d'activer ou non le système de mise en formation des troupes
-Un bouton permet de changer la gestion de l'affichage des couleurs selon l'équipe.

FIGURE 26 – Mini-Carte

Selectionbox
Il s'agit du rectangle qui va montrer au joueur quel zone il est en train de sélectionner.

TemporaryMessage
Placée en haut droite de l'écran, cette zone va afficher des messages temporairement pour justifier au joueur pourquoi il ne peut pas encore effectuer une action. Par exemple s'il essaye de construire un bâtiment alors qu'il n'a pas assez de ressource.

2.4.6 Units

Dans cette sous-partie je vais expliquer tout ce qui attrait aux unités de manière générale, les unités plus particulières comme les bâtiments ou les unités mobile seront détaillé plus loin dans le rapport.

Interactable
Il s'agit de l'unité de base, elle ne possède aucune méthode ni attributs, elle définit seulement que l'on peut interagir avec cet objet en faisant un clic droit dessus.

Selectable
Elle hérite d'Interactable, et permet de rendre une unité sélectionnable. En effet cette classe définit la notion de Highlight (quand on passe la souris sur une unité sans la sélectionner, un cercle de couleur va s'activer pour montrer que la sélection est possible) et de sélection (le cercle est opaque). Ce cercle va avoir une couleur différente selon que l'unité appartient ou non au joueur, à son équipe ou aux équipes ennemies. La classe contient aussi le nom de l'objet ainsi que son coût, sa description et le temps requis pour le produire.
Cette classe permet aussi de générer le champ de vision, c'est-à-dire jusqu'à quel distance le brouillard de guerre (voir brouillard de guerre) est dissipé autour de cette unité.

FIGURE 27 – Normal

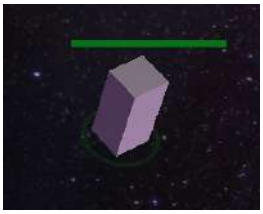

FIGURE 28 – Highlighted

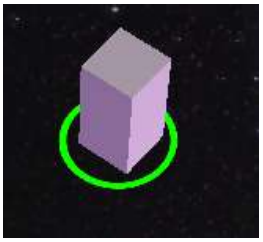

FIGURE 29 – Selected

DestructibleUnit
Elle hérite de Selectable, et permet de rendre une unité destructible. En effet cette classe définit la notion de jauge de vie et de mort. C'est ici qu'on définit la vie d'une unité, qu'on peut la soigner et la détruire.

Resource Unit

Elle hérite d'Interactable. Cette unité contient une certaine quantité d'un certain type de ressource et se détruit lorsqu'elle est vide.

2.4.7 Buildings

Constructedunit

Il hérite de DestructibleUnit, c'est le bâtiment de base. Il possède un Task System (voir Tasks) , et il se voit attacher un InConstructionUnit (voir InConstructionUnit) ainsi qu'un modèle du bâtiment mais tous vert et transparent (appelé Ghost, il sert à afficher le bâtiment lorsqu'on souhaite le placer). Un autre système lui est attaché, celui de la réparation, en effet si le bâtiment n'a pas son maximum de vie, alors on peut envoyer un ou plusieurs constructeurs le réparer.

Inconstructionunit

Il hérite aussi de DestructibleUnit, chaque ConstructedUnit a un InConstructionUnit, il s'agit du même bâtiment excepté qu'il est la version en construction, cela permet de gérer des scripts aux fonctionnements totalement différents de manière indépendante et donc plus simplement.
Il possède à peu près le même système de réparation que le ConstructedUnit mais pour la construction du bâtiment en lui-même.

production unit

Ce bâtiment va produire des unités, sa spécificité et des posséder une bannière de ralliement, c'est-à-dire que le joueur va placer une bannière attachée à ce bâtiment et les unités produite vont automatiquement essayer de rejoindre la bannière une fois instancié.

House

Ce bâtiment augmente la population maximale.

Radar

Ce bâtiment est constructible un nombre limité de fois (ce nombre peut être augmenté dans l'arbre des compétences). Il ne fait rien de particulier si ce n'est prévenir le joueur quand une unité ennemie entre dans son champ de vision. Un champ de vision volontairement plus grand que les autres bâtiments.

Station Spatiale

C'est le bâtiment principal de la base, il est l'endroit où les mineurs vont déposer

leurs ressources. C'est lui qui permet de produire des constructeurs.

EnergyFarm
Ce bâtiment génère continuellement de l'Énergie.

Farm
Ce bâtiment produit de la nourriture qui doit ensuite être récupéré par les constructeurs et amené à la station spatiale.

Asteroid
Cette unité hérite de ResourceUnit, elle contient un nombre définit de minerais (une des ressources du jeu) et s'autodétruit lorsque tout a été miné.

Laboratoire
Ce bâtiment génère continuellement des points de technologie (pour l'arbre de compétence).

Movable unit
Cette classe hérite de DestructibleUnit et contrairement au bâtiment elle est caractérisée par la possibilité de se déplacer. Elle peut aussi se faire hacker (L'unité appartient désormais au joueur qui hack).
Chaque unité mobile est équipée du système de patrouille, grâce à un Tools, on peut dire à une unité mobile de faire des allés et retours continuels entre son point actuel et le lieu pointé par la souris.

Constructeur
Cette unité mobile a beaucoup de fonctionnalités :
-Récolte : Elle fait des allés et retours entre son lieu de récolte et la station spatiale la plus proche en ramenant des ressources à chaque fois
-Construction et Réparation : Elle peut construire un bâtiment ou bien le réparer (plus il y a de constructeur qui construisent/réparent un bâtiment et plus cela va vite)
Cette unité m'a demandé beaucoup de temps et de réflexion pour bien m'organiser et que les divers comportements associés n'influent pas les uns sur les autres au risque de créer des bugs. (Par exemple, si un constructeur se voyait ordonner la construction d'un bâtiment alors qu'il en réparait un autre, il ne fallait pas que les 2 comportements se mélange sinon l'unité aurait eu un comportement chaotique). Chaque module est simple mais gérer tous ces modules est plus complexe.

Mobile Medical Station
Cette unité soigner les unités mobile alentours.

Hacker
Cette unité peut par définition en hacker d'autre (cela les fait changer de camps), mais aussi plein d'autre fonctionnalité qui seront débloquable dans l'arbre mais qui ne sont pas encore incorporées.

Basic troop
Troupe de combat par défaut son attaque et sa défense sont équilibrés.

Light Troop
Troupe de combat, son attaque et sa défense sont faibles mais il est très rapide et consomme très peu de population.

Bombardier
Unité de combat lente mais robuste et qui envoie des missiles très puissant mais à une cadence faible.

Croiseur
Unité unique très lente, très résistante, mais qui améliore les unités alliées alentours et une fois la bonne compétence débloquée servir de relais pour les bâtiments d'instantiation de vaisseaux. C'est-à-dire que les vaisseaux crées apparaîtront au niveau du croiseur.

2.4.8 Graphics

Brouillard de Guerre
Le principe du brouillard de guerre est simple, la carte du jeu est recouverte d'un brouillard, chaque unité à un champs de vision sous forme d'un cercle plus ou moins grand et qui va dissiper ce brouillard. Ce système est en réalité composé de 2 systèmes l'un est purement graphique et l'autre uniquement orienté programmation.
Graphiquement il s'agit d'une couche qui assombrit le jeu et grâce au système de shader intégré à Unity, on peut simuler qu'une zone est éclairé. Techniquement, lorsqu'une unité ennemie n'est dans le champ de vision d'aucun unité alliée, alors ses textures, son modèle, son icône de minimap sont désactivée, l'unité est invisible.
Ce système m'a posé beaucoup de problème car il utilise les shaders que je

connais très peu et qui ne se codent pas en C# mais Cg HLSL que je ne connais pas. De plus très peu de forum et de tutoriels existent à propos des brouillards de guerre. Il a fallu du temps et des recherches pour toutes la parties visuelle, la partie code étant plus traditionnelle.

FIGURE 30 – Fog Of War

2.4.9 Waitting room

Cette partie va détailler le fonctionnement de la Waitting Room, c'est-à-dire la salle d'attente dans laquelle les joueurs attendent que d'autres les rejoignent ou que le créateur de la partie la lance. Cette salle va gérer indépendamment chaque joueur à partir du moment ou il entre dans la salle d'attente. Tout d'abord un panneau de paramètre visible pour tous les joueurs va s'afficher, seul le joueur qui en est le propriétaire peut le modifier, ce panneau va permettre au joueur de définir, sa classe, son équipe, sa race et sa couleur. Le créateur de la partie peut ajouter des Bots (Intelligence artificielle) et lancer la partie une fois que tous les joueurs ont coché la case ready. Au lancement de la partie, le Players Manager va répartir les coordonnées de départ de chaque joueur de sorte que les joueurs étant dans la même équipe se retrouve proche les uns des autres.

2.4.10 Arbre des Compétences

Au-delà grâce à la construction nous avons voulu qu'une notion de progression soit présente dans le jeu et qui plus est qui permette aux joueurs de se différencier de sorte que le vainqueur ne soit pas seulement déterminé par la stratégie en combat mais aussi par tout le déroulement de la partie. Nous avons

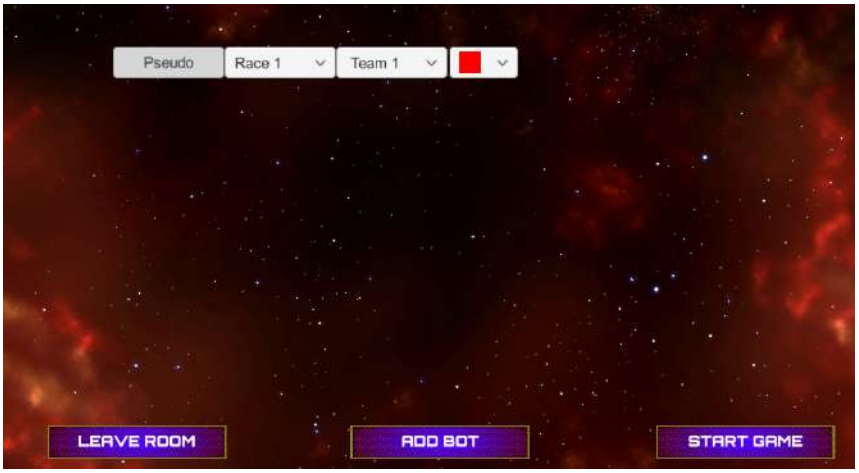

FIGURE 31 – Salle D'attente

donc opté pour un arbre de compétence. Celui-ci est découpé en plusieurs sous-arbre distincts dédiés chacun à un secteur particulier que sont :
- Attaque légère
- Attaque lourde (bombardier)
- Recherche et développement (Hacker)
- Croiseur
- Économie (Construction, ressources)
- Défense (tourelle)

Chacun de ces arbres permet donc de débloquer des unités ou des compétences mais aussi à améliorer certaines statistiques comme la vitesse de construction ou les dégâts par seconde d'une unité. De plus ces arbres sont restrictifs, c'est-à-dire que lorsque plusieurs compétences se trouve à la même hauteur de l'arbre, choisir l'une d'entre-elle va bloquer toutes les autres. Le but étant qu'en fin de partie, il n'y ai que très peu de chance que les joueurs aient fais les mêmes choix et se retrouve finalement avec la même base et la même armée. L'achat de compétence coûtent des points de technologies.

2.4.11 Online

Photon

Pour gérer le multijoueur dans notre jeu nous avons choisi d'utiliser l'asset Photon PUN 2 notamment pour son accessibilité mais surtout parce que l'outils nous met à disposition un serveur hébergé sur les serveurs Photon qui offrent

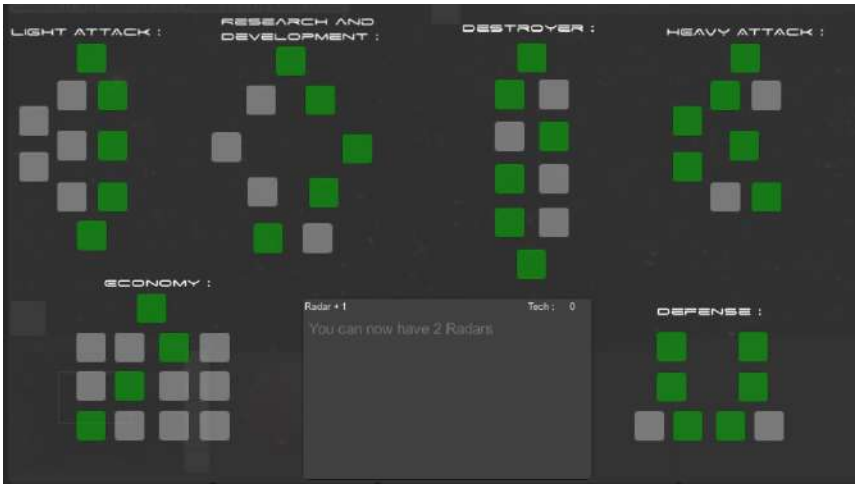

FIGURE 32 – Arbre des Compétences

une bien meilleure stabilité que si nous l'avions hébergé nous-même.
J'ai suivi les conseils des InfoSpé en implémentant très rapidement le multi-joueur. PUN offre plusieurs méthode pour synchroniser en ligne les divers éléments du jeu, tout d'abord on peut choisir d'envoyer un stream qui contient diverse variable à synchroniser, sinon on peut utiliser les RPC qui permettent d'exécuter une fonction sur toutes les instances d'un même objet.

2.4.12 Bilan et Perspective pour les prochaines soutenances

Je suis très satisfait de mon travail, comparativement au planning je suis en avance, une avance calculée car mon rythme de travail sur le projet devra baisser à l'approche des partiels car nous aurons beaucoup de travails. L'idée était d'avoir développé une très grande majorité des mécaniques de jeu pour cette soutenance afin de ne laisser que le système de combat et surtout l'intelligence artificielle pour les autres. Cette dernière va s'avérer être un très grand défi pour nous puisque nous n'avons aucune expérience ni connaissance en la matière. Ainsi pour la prochaine soutenance, je prévois d'avoir fait le système de combat ainsi qu'avoir bien commencé l'Intelligence Artificielle, au moins d'un point de vue théorique.

3 Conclusion

Pour résumer, les bases du jeu sont là et les prochaines soutenances verront arriver de l'intelligence artificielle ainsi que les textures, modèles et différents effets visuels du jeu. Cette avance a été prise à cause de la difficulté du second semestre plus élevée que celle de la première, nous avons donc profité du début du second semestre pour travailler un maximum de sorte à avoir moins de travail quand les autres cours se corseront.