ID du projet : 14

Projet cassiopée - Rapport

Projet Rupture

3 JUIN 2024

ROUX Basile

LANTIGNY Valentin

SCHROEDEL Adrien

BENNANI Tawfik

Encadré par :

Michel SIMATIC

Table des matières

Introduction

Quatre élèves appréciant les jeux vidéos se lancent le défi d'en faire un engagé sur la thématique de l'environnement avec deux contraintes supplémentaires; inclure dans ce dernier les trois étapes d'une révolution : ridicule, dangereuse puis évidente, et le réaliser sur Unreal Engine.

Avec ce projet Cassiopée, nous souhaitons montrer que le jeu vidéo à encore sa place dans un futur durable, tant par son rôle évident de divertissement que par son rôle, peut-être plus méconnu, de vecteur d'émotion et de sensibilisation.

Premièrement, nous sommes tous très peu familiarisé avec l'Unreal Engine et son langage de programmation, le C++. Nous sommes cependant convaincus que ce moteur de jeu est un excellent choix pour notre jeu car il facilite grandement l'établissement de paysages réalistes qui favorisent l'immersion et donc l'impact de message que nous souhaitons faire passer à travers notre jeu, comme nous y reviendront.

Ensuite, les contraintes imposées sur notre jeu : la durée du jeu d'abord, imposée à une quinzaine de minutes maximum. Durant ce laps de temps nous devons nous assurer que le joueur soit suffisamment investi dans le jeu, ce qui implique une durée de contextualisation, mais néanmoins réceptif à un message qui se glisserait dans le jeu. Les trois étapes d'une révolution apparente ensuite, qui ont nécessité une recherche en amont pour bien les appréhender, puis une élaboration scénaristique nous permettant de les faire apparaître dans le produit final (tout en gardant toujours en tête la contrainte temporelle). Enfin, le temps de développement se totalisant à 800h peut paraître léger pour développer un jeu vidéo, particulièrement en tenant compte des formidables outils fournis par Unreal Engine qui peuvent encourager à être trop ambitieux (vouloir trop bien faire, apprendre et utiliser des technologies qui, bien que magnifique, prendrait du temps que nous n'avons pas etc…). Nous avons choisi de faire un jeu à scénario avec un gameplay simple et une mise en situation rapide, avec un travail particulier sur l'environnement du joueur, les dialogues entre personnages et les cinématiques.

Enfin, ce projet est un projet de groupe, ce qui impose l'utilisation d'un outil de contrôle de version. Nous nous sommes alors tourné très naturellement vers git et github. Cependant nous avons rencontré des problèmes nous laissant penser que ce n'était peut-être pas le meilleur outil pour travailler avec Unreal Engine.

Unreal Engine

Choix d'Unreal Engine 5 pour le développement de notre jeu

Dans le cadre du développement de "Rupture", nous voulions un jeu qui se distingue par son réalisme et ses décors quasi réels. Le but étant d'obtenir une expérience immersive pour faire passer notre message, mais aussi d'impressionner les futurs joueurs qui sauront que ce n'est qu'un projet étudiant. Cette décision a été motivée par plusieurs avantages clés qu'offre Unreal Engine 5, notamment pour une équipe ayant une expérience limitée dans la création de jeu aussi ambitieux graphiquement.

Les Mégascans : Avec une bibliothèque de plus de 15000 Mégascans, Unreal Engine 5 met à disposition une vaste gamme de textures et de modèles 3D photoréalistes. Ces ressources sont essentielles pour créer des environnements immersifs et détaillés qui renforcent l'expérience du joueur.

La Lumière : Le système d'éclairage dynamique d'Unreal Engine 5 permet de simuler des conditions d'éclairage naturelles avec une précision remarquable. La lumière joue un rôle crucial dans l'atmosphère du jeu et contribue à la crédibilité des scènes.

Le Système d'Eau : Le moteur offre des outils avancés pour la simulation de l'eau, permettant de créer des plans d'eau réalistes, des océans dynamiques et des interactions fluides avec les éléments du jeu.

Les Assets mensuels gratuits : Unreal Engine 5 enrichit régulièrement sa bibliothèque avec des assets gratuits chaque mois. Cela offre aux développeurs de nouvelles ressources pour expérimenter et peaufiner leurs créations sans coût supplémentaire.

Sculpter et Peindre des Landscapes : Une des forces d'Unreal Engine 5 réside dans sa capacité à permettre aux développeurs de construire rapidement des terrains gigantesques et variés, qui impressionnent par leur étendue et leur diversité. Grâce aux outils de sculpture et de peinture de paysages intégrés, nous avons pu créer des environnements vastes et détaillés en un temps record. Les textures réalistes, qui s'appliquent avec une facilité déconcertante, ajoutent une couche supplémentaire de réalisme, rendant chaque scène plus vivante et crédible. Une manière simple d'en mettre plein la vue aux joueurs.

Le Système de Foliage : Unreal Engine 5 intègre un système de foliage robuste qui facilite l'ajout et la gestion de la végétation. Ce système est indispensable pour donner vie à des environnements naturels denses et diversifiés.

Les Classes d'Unreal Engine 5 : L'un des atouts majeurs d'Unreal Engine 5 est l'ensemble de classes prédéfinies, qui offrent une base solide pour le développement rapide de personnages interactifs. Ces classes viennent avec une multitude de comportements préprogrammés, ce qui permet aux développeurs de se concentrer sur la personnalisation plutôt que sur la construction de fonctionnalités de base. Que ce soit pour des mouvements fluides, des interactions complexes ou des réactions environnementales, ces classes facilitent grandement la création de personnages dynamiques et crédibles, enrichissant ainsi l'expérience de jeu.

MetaHuman : Enfin, Unreal Engine 5 inclut MetaHuman, un outil révolutionnaire pour la création de personnages humains numériques hautement réalistes. Cela permet de générer des personnages expressifs et détaillés, renforçant ainsi l'immersion narrative de notre jeu.

En somme, Unreal Engine 5 s'est imposé comme le choix idéal pour notre projet, offrant une suite complète d'outils et de fonctionnalités qui nous ont permis de réaliser notre vision créative avec un niveau de réalisme et d'immersion exceptionnel.

L'apprentissage d'Unreal Engine

Comme vu précédemment, l'apprentissage d'Unreal Engine était un des éléments phares de notre projet. Nous avons tous commencé par appréhender le moteur et la base du développement avec Unreal en suivant l'excellent cours "Udemy - Unreal Engine 5 C++ The Ultimate Game Developer Course". Il est apparu que, au vu de l'ambition de notre projet, nous prenions du retard sur ce dernier en ne faisant que se former et, aux alentours de Février, nous avons réellement commencé à nous concentrer sur le développement de "Rupture".

Avec les solides bases que nous venions d'acquérir, nous avons pu avancer efficacement dans nos tâches respectives. Nous formant au besoin à l'aide des nombreuses ressources disponibles sur internet (la documentation d'Unreal Engine, youtube, autre…).

L'apprentissage, bien que long, n'a pas posé davantage de contraintes dans l'avancement du projet.

Points techniques rencontrés

Élaboration du Scénario : Un Message Écologique au Cœur de l'Aventure

La conception du scénario de notre jeu a été guidée par la volonté de transmettre un message écologique intelligent et engageant. Nous avons souhaité que ce message soit intégré de manière intrinsèque à l'expérience ludique, plutôt que d'être un simple vernis thématique. Pour ce faire, nous avons structuré la trame narrative autour des trois étapes classiques d'une révolution : initialement perçue comme ridicule, elle devient ensuite menaçante, avant d'être finalement acceptée comme évidente.

Dans notre quête d'équilibre, nous avons choisi de plonger les joueurs dans un monde futuriste proche, où ils suivent le quotidien d'un couple dépendant de la technologie. Cette proximité narrative permet aux joueurs de s'identifier facilement aux protagonistes et de s'immerger dans leur évolution. Le scénario a été méticuleusement réécrit pour affiner les dialogues et les situations, éliminant toute artificialité et assurant une progression crédible et cohérente.

Le message central, qui met en lumière notre dépendance excessive aux technologies, est distillé avec subtilité tout au long du jeu. Il ne s'agit pas de sermonner, mais d'amener le joueur à une prise de conscience personnelle à travers les événements du récit. Les protagonistes évoluent lentement, permettant aux joueurs de s'engager émotionnellement et d'assimiler naturellement les enjeux écologiques.

La mise en scène et l'esthétique des décors jouent également un rôle crucial dans la transmission de ce message. Des croquis préparatoires ont été élaborés pour définir l'apparence et l'atmosphère des lieux, garantissant que chaque environnement renforce le propos du jeu. Cette phase de conception scénaristique, bien que plus longue que prévue, a été fondamentale pour établir une base solide sur laquelle l'ensemble du jeu a été construit.

Modélisation des personnages : film de vrais acteurs + animation pour le corps

L'animation des personnages 3D se décompose en deux parties, l'animation facile et l'animation du corps. Sur Unreal Engine, les Metahumans ont par ailleurs séparé les deux Rigs (c'est-à-dire les deux "armatures" ou "structures osseuses") pour chaque personnage afin d'économiser en ressource.

Pour l'animation des personnages, nous avons eu recours à des banques d'animations et de motion capture gratuites, notamment Mixamo, que nous avons mixé pour correspondre aux phrases des personnages. Nous avons voulu faire de la motion capture également à l'aide d'outils gratuits tels que Move AI mais les résultats sont trop peu fiables. Il a été finalement plus rapide de chercher des animations plus ou moins appropriées que de nettoyer les résultats approximatifs fournis par ces outils. Cependant, l'animation facile par motion capture est plus accessible. Les iPhone récents disposent tous de l'outil LiveFace qui permet de s'enregistrer et de faire correspondre ses expressions à un personnage 3D et les Metahumans disposent d'un lien direct avec LiveFace.

Gestion de GIT (+ LFS)

L'utilisation de Git ou d'autres logiciels de contrôle de version est indispensable dans un projet en groupe. Ces outils permettent le travail simultané sur un même système de fichiers et leur maîtrise au sein de l'équipe constitue un véritable vecteur de productivité. La gestion de ces outils a donc été une priorité tout au long du projet. Nous avons également dû faire face à d'autres restrictions, notamment concernant la taille des fichiers avec lesquels nous travaillions.

Versionnage de fichiers lourds

Les mini-projets que nous avons réalisé en école jusque là (TP, DM et autres devoirs où l'usage de Git était nécessaire ou demandée), étaient en majorité légés en nombre d'octets total, c'est à dire inférieur à 1 GB. Le projet RUPTURE va alors créer une *rupture* de cette habitude.

Le poids des fichiers dont le projet a besoin pour être développé, environne les 40 GB en somme. Pour le Git classique que l'on connaît, le quota est largement dépassé : GitHub recommande une taille du répertoire maximum de 5 GB. Dans le cas contraire, la mise en ligne des fichiers peut être annulée, et un avertissement est reçu. Ils proposent cependant le service de Git LFS (pour Large File Storage) qui

étend cette limite à 50 GB pour 5€ par mois (après arrêt de l’abonnement, tout commit ajoutant de la taille au dessus des 5 GB classiques est refusé) cette somme est donc dépensée uniquement pendant les mois de développement.

La mise en place de ce service nécessitait une installation particulière qui nous a posé problème pendant un certain temps. Les fichiers volumineux sont en réalité stockés sous forme de pointeurs (pointant vers les serveurs de Git-LFS), et leur format n'était pas uniformisé sur nos machines. Cela était dû à l'installation de Git-LFS en cours de route, au lieu de l'avoir fait dès le départ. Une fois les incompatibilités résolues, début avril, nous n'avons plus rencontré de problèmes avec Git-LFS.

Conflits de fusion

Les conflits de fusion, plus connus sous le nom de *merge conflicts* en anglais, ont été particulièrement embêtants durant notre projet. Unreal Engine utilise majoritairement des Blueprints (codage visuel à la type “Scratch”), qui ne sont pas facilement traduisibles en texte. Cela rend les conflits de fusion impossibles à résoudre sans perte de données lorsqu'ils surviennent. Cette contrainte supplémentaire nous a poussés à travailler systématiquement sur des fichiers différents à chaque commit. Cela a nécessité beaucoup de communication une élimination rigoureuse de l'effet tunnel, qui aurait pu être dévastateur.

Retours Personnel de projet

Basile

Mon ressenti sur Unreal est partagé, j'ai été véritablement impressionné par la puissance de certaines features, pouvoir faire en quelques clics des paysages aussi réalistes, avoir en quelques clics un personnage fonctionnel, tester rapidement des bouts de codes en blueprint, nul doute qu'Unreal est un outil de travail de choix dans un cadre professionnel et pour des jeux d'une certaine ambition. Néanmoins, pour des projets plus modestes, l'expérience fut des plus désagréables:

- Problème de performance, Unreal se nourrit littéralement de la ram et de la carte graphique.
- Travail en équipe, Unreal est immensément moins pratique que d'autres moteurs tel qu'Unity en ce qui concerne le travail en équipe. Un commit de travers, j'entends par là avec l'éditeur encore ouvert, peut potentiellement corrompre l'entièreté d'une branche.
- En réalité, Unreal ne semble pas pouvoir s'intégrer proprement avec git, les blueprint par exemple, assez fondamental dans Unreal Engine, n'ont aucun moyen de se merge. Pire encore, un changement dans un blueprint n'est pas nécessairement détecté par Unreal et on se retrouve alors avec des blueprints différents selon le membre du projet, problème réglable uniquement en s'envoyant les dit blueprints d'une autre manière.De plus, nous avons rapidement atteint un stade où le projet dépassait les 40 GO et avons dû payer entre 30 et 40 euros de bande passante github. Je ne parlerai pas non plus de la difficulté à comprendre l'outil git L(arge) F(ile) S(ystem) qui a allongé de plusieurs heures mes différents commits et merge.

Pour toutes ces raisons, et d'autres que je pourrai oublier, je ne pense pas pousser plus loin mes usages d'Unreal Engine dans un avenir proche. S'il me faut faire un jeu en tant que projet personnel, j'irai inévitablement me tourner vers la concurrence (Unity …).

Pour finir sur une note positive sur mon ressenti, je suis extrêmement fier de ce que nous avons accompli dans ce projet, nous étions, je pense, ambitieux sur ce que nous voulions faire, mais (sans doute grâce à Unreal Engine qui a accélérer beaucoups de choses grâce à toutes ses fonctionnalités) nous avons pu répondre à un large majorité de nos objectifs et de notre cahier des charges dans un temps si limité.

Valentin

J’ai personnellement été très impliqué dans le processus de scénarisation du jeu. De la création de la trame globale du jeu et du message, à la rédaction de la plupart des dialogues. Je me suis également occupé de trouver et de faire enregistrer les acteurs. Il fallait notamment trouver la bonne configuration pour enregistrer le son, l’image (pour obtenir les expressions faciales), pouvoir lire le texte sans loucher ce qui évite de l’apprendre, et avoir une bonne lumière.

Nous avons essayé de boucler ce processus le plus tôt possible dans le projet. En effet, le jeu repose entièrement sur son scénario et ses dialogues. Une fois écrits et enregistrés, le chemin est tout défini sur ce qu’il reste à faire. Il faut simplement développer le jeu pour y intégrer les dialogues, et construire les décors pour servir le message de l’histoire.

L’autre tâche qui m’a bien occupé sur ce projet est la construction de la partie “Open World” de notre jeu. Dès que nous avons su que nous utiliserons Unreal Engine 5, nous avons directement voulu faire un jeu beaucoup plus avancé techniquement que ce que nous avions pu faire jusque là. Comme la plupart des membres du groupe, mon expérience en création de jeu ne s’aventurait pas plus loin que des petits projets sans prétention sur quelques semaines, ou des jeux de Game Jam. Mais Unreal Engine 5 offre des outils qui permettent d’obtenir un résultat très impressionnant en très peu de temps. Il est conçu pour le grandiose. Nous avons donc pris le parti de changer nos habitudes et d’embrasser pleinement les ressources de ce moteur graphique, en faisant un jeu scénarisé, et en ayant une partie “vitrine technique” : un monde dans lequel le joueur peut se balader librement, et qui exploite toutes les ressources d’Unreal pour donner un résultat magnifique. Cela a pris du temps car il faut expérimenter et paramétrer chacune des fonctionnalités. On peut potentiellement passer une quantité infini de temps pour obtenir le résultat parfait. Mais à la fin le résultat est réellement impressionnant.

Adrien

Rédigeant ceci vers la fin du projet, on s'aperçoit du chemin parcouru par l’équipe sur ce semestre. Au départ tous motivés pour produire un jeu vidéo travaillé et de qualité, nous avons pu au travers de ce projet comprendre le processus classique que nécessite la création d’un jeu.

En quelques mots : j’ai adoré contribuer à la réalisation de RUPTURE et en avoir profité pour grandir dans la maîtrise en Game Design et en compétences techniques dans Unreal Engine et l’usage de GIT.

Nous sommes également tombé dans le piège classique : le projet a mis du temps à démarrer pendant la période d’auto-formation à Unreal Engine, mais par moment, l’équipe se voyait entrer dans des pics d’efficacité où le projet avançait rapidement. Un des points les plus bloquants a été la gestion de GIT (et GIT-LFS comme expliqué dans une section plus haut). Nos travaux simultanés ne s'imbriquaient parfois pas correctement, ce qui conduisait à des pertes de temps non négligeables. En effet, il nous arrivait de devoir reproduire un travail perdu en raison d'une négligence dans l'utilisation de GIT. Heureusement, à force de remédier à ces problèmes, nous trouvions des solutions de plus en plus convenables, jusqu’à trouver une solution que l’on pense aujourd’hui être la plus utile et rapide à exécuter pour récupérer le travail perdu.

Concernant la partie programmation, j'ai particulièrement apprécié la création de schémas de logique (le “diagramme de classe”) pour illustrer la logique du jeu. Bien que j'aie été initialement sceptique comparé à la manière de coder classique, j'ai trouvé que l'utilisation des Blueprints dans Unreal Engine (à la style “*Scratch*” mais en bien plus complexe) était à la fois intuitive et puissante, ce qui a grandement facilité le développement et m'a permis d'exprimer pleinement ma passion pour la programmation. Par exemple le système d’intéraction du robot avec les autres objets qui a été conçu pour être hérité et réutilisé, a été réfléchi pour que le même code soit utile à plusieurs endroits différents.

L'ajout de l'interface utilisateur (UI) m'a également beaucoup plu. En effet, j'accorde une grande importance à la fluidité de l'expérience de jeu, et l'UI joue un rôle crucial dans cet aspect. C'est pourquoi je n'ai pas hésité à consacrer du temps à l'élaboration des animations de l'interface, afin de rendre le jeu plus agréable et immersif pour les joueurs.

Tawfik

J'étais très enthousiasmé dès le départ par la possibilité de pouvoir consacrer une année scolaire à la création d'un jeu-vidéo car je sentais que c'était un domaine qui avait le potentiel de me passionner et mes attentes ont été plus que satisfaites. En investissant du temps dans la recherche de ressources pédagogiques j'ai pu découvrir que la communauté de Game Dev et de 3D en général est très généreuse et offre tout ce qu'il faut pour pouvoir atteindre des résultats au niveau des standards d'aujourd'hui tout en laissant libre cours à son imagination.

Les cours de Stephen Ulibarri sont par exemple extrêmement riches et montrent non seulement comment créer de multiples systèmes de mécaniques fonctionnels mais aussi comment les structurer d'un point de vue ingénierie logicielle pour correspondre aux attentes d'un studio professionnel. En dehors de la programmation, Internet regorge de multiples professionnels de l'industrie qui partagent leur savoir sur la création et l'animation de personnages, la réalisation d'environnements et leur optimisation, les FX, etc…

J'ai d'ailleurs passé deux mois à me former à Houdini, un logiciel de 3D spécialisé dans la simulation physique et la modélisation procédurale pour ne finalement quasiment pas l'utiliser. L'intégration de ce que j'avais en tête dans notre jeu Unreal était trop compliqué et m'aurait demandé quelques mois de recherches supplémentaires, bien qu'Epic Games et SideFX (la boîte à la tête de Houdini) soient en collaboration depuis quelques années pour rendre cette intégration plus simple.

Ce projet a donc été l'occasion pour moi de tester un maximum de choses et de me mettre au défi en réalisant un projet qui devait avoir un message clair et cela sans trop s'éparpiller.