ESPIMA BUSINESS SCHOOL

MÉMOIRE DE MASTER PROFESSIONEL

MANAGEMENT DE PROJET
PROJECT MANAGEMENT

AMÉLIORATION DU WORKFLOW D'UNE ÉQUIPE DE DÉVELOPPEURS DE JEUX VIDÉO

RÉALISÉ PAR

Hedi Mlika

SOUS LA DIRECTION DE

PROFESSEUR SAMIR BORGI, ENCADRANT ACADEMIQUE, EBS
M. JIHED JOUABI, ENCADRANT PROFESSIONNEL, LANTERNS

ANNÉE UNIVERSITAIRE 2020 / 2021

Remerciements

Avant tout développent, il parait opportun de commencer ce rapport de stage par des remerciements, J'adresse mes sincères remerciements aux personnes qui m'ont aidé tout au long de ces trois mois à la réalisation de ce stage.

En premier lieu, je tiens à remercier M. Samir Borgi, encadrant de l'Espima Business School, qui m'a donné de son temps ainsi que sa confiance pour mener ce stage à bien et à l'élaboration du rapport avec beaucoup de patience et de pédagogie et pour ses conseils qui m'ont été d'un grand soutient.
De même ; je tiens à remercier M. Nabil Brahem, COO et M. Oussama Ben Mariem, CTO, tous les deux co-fondateurs de Lanterns Studios qui m'ont donné de leurs temps et qui ont grandement contribuer à la réalisation des objectifs fixés.

Je tiens également à remercier tous les membres de Lanterns Studios et en particulier, M. Aberrahmen Brinis, M. Issam Farjaoui et Mme Myriam Jegham pour leurs conseils, aides et bonne humeur.
En tout particulier, je tiens à remercier M. Jihed Jaouabi, CEO et co-fondateur de Lanterns Studios qui m'a fait confiance, jugé mon travail avec un sens critique et avec pertinence du début à la fin et qui m'a donné de son temps.
De même, je tiens aussi à remercier tous les membres du jury qui m'ont honoré pour avoir accepté de prendre le temps de juger mon travail.

Enfin, je tiens à exprimer mon amitié et mon respect profonds envers ma famille, la direction d'EBS, mon amie et soutient morale d'outre-mer et à tous ceux et celles qui ont contribué de près ou de loin à la réussite et à la concrétisation de ce stage.

Je dédie ce rapport à mes parents, ma sœur et à tous ceux qui m'aiment.
À la mémoire de mes grands-parents

Abstract

Français

Ce stage consiste à la mise au point d'une charte illustrant le process de gestion des tâches et des livrables pour un projet dont le produit final est un jeu vidéo pour le compte de l'entreprise : Lanterns Studios.
Ce process appelé « pipeline » va notamment permettre aux équipes actuelles de visualiser tout le processus et les tâches à faire lors de la création d'un jeu vidéo en 3D mais aussi au top management d'avoir un moyen d'expliquer aux potentiels nouveaux membres, le déroulement du développement et situer leurs tâches par rapport aux autres tâches des autres membres/rôles. Ce pipeline est une mise à jour d'un pipeline déjà existante au tout début de l'entreprise mais prenant en compte les modifications et changements de leur workflow que l'entreprise a effectué jusqu'à aujourd'hui.
Le stage consiste aussi à trouver un moyen de gérer efficacement les ressources et fichiers partagés par toutes les équipes et leurs versions.
En d'autres termes, le stage consiste à améliorer et rendre plus rapide le workflow des équipes de développeurs de jeux vidéo.
Mots clés : Pipeline – Process – Jeux vidéo – Lanterns Studios – Versioning

English

This internship consists of the development of a chart that illustrates the process for managing tasks, workflow and deliverables for a project whose final product is a video game for the company: Lanterns Studios.
This process called "pipeline" will, in particular, allow the current teams to visualize the whole process and the tasks to be done during the creation of a 3D video game, but also, to the top management so they have the means of explaining to the potential new members, the development progress and situate their tasks in relation to the other tasks of other members / roles. This pipeline is an update of a pipeline that already existed at the very beginning of the company but takes into account the modifications and changes in their workflow that the company has made to date.
The internship also consists of finding a way to efficiently manage the resources and files shared by all the teams and their versions.
In other words, the internship consists of improving and making faster the workflow of video game developers / teams.
Keywords: Pipeline – Process – Video games – Lanterns Studios – Versioning

Table des Matières

Liste des Figures

Liste des Tableaux

Pour une meilleure expérience lors de la soutenance, veuillez, s'il vous plait, vous rendre à **l'utilitaire de la soutenance** à l'**Annexe 3**. Merci !

Bonne Lecture

Introduction générale

Un jeu vidéo (en anglais : video game) est un programme à but ludique qui se joue sur un appareil électronique doté d'une interface utilisateur permettant une interaction entre le joueur et un monde virtuel. Le joueur a en sa disposition des périphériques pour agir et réagir sur le jeu.

Le jeu vidéo se jouait au début sur des bornes d'arcade et il a évolué pour être jouable sur PCs, consoles et consoles portables et plus récemment sur smartphones et casques a réalité virtuelle.

Au fils de son histoire, le jeu vidéo a grandement contribué à l'évolution de la technologie et à la recherche scientifique ; notamment en 3D, en puissance de calcul, en réseaux, en supports de stockage et en simulations…etc. Ce domaine souffre malgré lui de fortes controverses, car le jeu vidéo -comme loisir et phénomène de masse- soulève des interrogations et des critiques. Le jeu vidéo est à l'échelle de l'histoire des sociétés humaines une activité récente, les parents d'enfants nés dans les années 90 n'ont pour la plupart jamais joué à ce type de jeu dans leur enfance ou adolescence. *[Voir ref ig1, Annexe 2, Page 63 : définitions sur les jeux vidéo].*

Un développeur de jeu vidéo est une personne physique ou morale comme une société ou une startup qui crée des jeux vidéo en rassemblant plusieurs domaines de compétences comme le game design, la programmation, la scénarisation, l'infographie, la musique, etc. Le développeur de jeux vidéo peut travailler au sein d'une entreprise, d'un collectif ou être indépendant. Un développeur de jeu vidéo utilise divers outils pour créer un jeu comme des langages de programmation, des logiciels de graphisme ou des moteurs dédiés à la création de jeux.

Un jeu vidéo est un projet de grande envergure qui nécessite divers domaines de connaissance, une organisation et un plan structuré quel que soit la taille du jeu vidéo ou de l'équipe. Un planning permet de fluidifier l'implémentation des diverses ressources dans le jeu vidéo, comme un personnage en 3D ou un élément de gameplay. C'est deux ressources

nécessitent deux compétences différentes, qu'un seul individue, deux individus, voire deux équipes peut avoir.

Différents types de systèmes sur lesquels le jeu vidéo se pratique coexistent, et de nombreux jeux sont dorénavant disponibles sur ces plates-formes. Les consoles de jeux, les bornes d'arcade et les ordinateurs, en sont les trois principaux vecteurs. Les ordinateurs sont des plates-formes informatiques hétérogènes qu'il est possible de trouver et de faire évoluer vers différentes puissances. Ils ne sont pas spécialement prévus pour jouer, mais de par leur modularité certaines configurations se prêtent aux jeux, parfois par l'adjonction de matériel dédié comme une carte graphique [1]ou un périphérique de contrôle particulier. *[Voir ref ig2, Annexe 2, Page 63 : plateformes de gaming et cartes graphiques]*

[1] Composant très complexe, très spécialisé et presque imbattable dans le rendu d'images en 3D. Très souvent obligatoire pour faire marcher un jeu vidéo. Plus le jeu est complexe et graphiquement détaillé, plus il est nécessaire d'avoir une carte graphique avec des performances élevés.

Chapitre **Premier**

Contexte et Cadre spatio-temporel du stage

Ouverture

Ce rapport est divisé en 3 chapitres, celui-ci, le premier chapitre, est consacré au cadre général du stage et la problématique à résoudre, c'est un chapitre introductif qui permet de contextualiser les chapitres suivants.

Le second chapitre est réservé au déroulement du stage en lui-même, des tâches et travaux effectués et les sous objectifs fixés pour atteindre au mieux un objectif final satisfaisant, ainsi que les différentes techniques et outils utilisés pour obtenir un résultat final, théorique ou expérimental.

Au troisième et dernier chapitre, nous analyserons les résultats obtenus en synthétisant et critiquant notre travail et prévoir des pistes d'améliorations si possible.

Vous trouverez toutes les sources, utilitaires et annexes dans les toutes dernières pages.

I. Contexte et thèmes à aborder

Le présent travail fait partie de la préparation du mémoire, dans le but d'obtenir un *Master en Management de projet* de l'enseignement supérieur de la prestigieuse école supérieure privée *EBS (Espima Business School).* De plus, un stage s'est déroulé dans une entreprise nommée *Lanterns Studios* pour une période de trois mois à compter du 22 Février 2021.

Le stage comprend l'intégration dans l'équipe, la recherche et l'analyse d'une problématique au sein de Lanterns Studios, que, dans notre cas, c'est la mise en œuvre d'un pipeline pour une équipe de développeurs de jeux vidéo, ainsi que, plus généralement, aider à fluidifier leur processus de travail.

Afin de mieux se lancer dans le vif du sujet, une halte sur quelques termes qui seront utilisés tout au long du rapport. Ces « termes » décrivent les thèmes généraux qui seront mentionnés à plusieurs reprises dans ce rapport et seront très utiles pour la compréhension des chapitres suivants dans le but de mieux les appréhendés.

1. Pipeline / Process

Dans n'importe quel projet, quelle que soit la vente de fruits, la construction d'un immeuble ou même le développement d'un jeu, le terme « pipeline » est utilisé pour

décrire une démarche à suivre vers un objectif clair. Cet objectif peut être à court et moyen terme, mais généralement, cette construction vise à créer une démarche dans le but de visualiser une route avec des étapes à suivre vers un objectif à long terme.

En effet, le pipeline est un terme anglais et est souvent utilisé pour décrire les progrès à travers une série d'étapes aboutissants à un objectif à long terme. Dans de nombreux cas, l'expression est utilisée pour décrire un processus en cours.

Processus *ou process en anglais* peut être, dans certains cas, un synonyme du mot *pipeline*. La différence est que, *pipeline* ne peut s'appliquer qu'à une vue globale du projet. Alors que processus peut aussi bien décrire une vue globale qu'une vue micro managériale ou un état atomique du projet[2].

2. Workflow

Un workflow est un ensemble de processus, généralement non linéaires, souvent non conceptuels et non définits. Mais on peut le définir à partir de l'ensemble des processus qui interviennent sans pour autant avoir un produit final à la clé.

Pour mieux comprendre, nous allons prendre l'exemple de la partie administrative d'un projet. La partie administrative d'un projet est souvent crucial pour le bon déroulement d'un projet et fait partie du workflow car il mobilise du personnel mais n'interfère en aucun cas dans le « comment » un produit est réalisé.

En d'autres termes, le workflow est un système global et déterministe. *[Voir ref 1.1 Annexe 2, Page 63 : Définition du Workflow]*

3. Développeur

Dans le monde du jeu vidéo, la notion de développeur englobe plusieurs professions et ne se réfère pas uniquement aux programmeurs.

A titre d'exemple :

- Un programmeur, un designer, un artiste : Sont des développeurs.
- Un expert marketing, un directeur d'opération, un analyste : Ne sont pas des développeurs.

[2] Un état atomique désigne un état simple qui ne peut pas être décomposé ; exp : une tâche.

4. Indie Game Studio

Indie Game Studio ou *Indie company* est l'expression qui désigne une société de développement de jeux vidéo qui, comme dans le secteur de la musique ou du cinéma, désigne des jeux vidéo crées sans l'aide financière d'un éditeur de jeux vidéo. Le jeu vidéo indépendant connait une montée en puissance depuis le début des années 2000, notamment grâce à la distribution dématérialisée des jeux vidéo dans des plateformes tel que Steam[3] ou bien Epic Games Store[4] pour les jeux PC. *[Voir ref 1.2, Annexe 2, Page 63 : Définition du jeu vidéo indépendant]*

Depuis le milieu des années 2000, le jeu indépendant profite de l'émergence du financement participatif. Ce système permet de financer une production auprès d'une communauté d'internautes. La quantité de fonds levée peut être très importante du fait du nombre de contributeurs, même si individuellement, chaque contribution est faible.

De manière classique, un studio peut avoir recours à un prêt bancaire pour financer le développement d'un jeu. Cependant, la plupart des productions indépendantes sont autofinancées, notamment grâce aux fonds propres et à l'épargne issue de productions précédentes.

La société peut avoir recours dans certains cas à un éditeur, alors dans ce cas, le jeu n'est plus considéré comme « indie » mais la société le reste de par son statut si elle n'est pas rachetée par la société éditrice.

5. Jeu AAA

Dans l'industrie du jeu vidéo, AAA (prononcé « triple A ») est un terme de classification utilisé pour les jeux vidéo dotés de budgets de développement et de promotion les plus élevés. Les joueurs et la critique s'attendent à ce qu'un titre considéré comme étant de type AAA soit un jeu de grande qualité ou figure parmi les jeux les plus vendus de l'année. Un titre AAA est destiné à montrer le meilleur des capacités d'une entreprise de jeu ou d'une franchise. Les jeux qui ne sont pas de type AAA sont considérés comme étant des « titres

[3] Steam est une plateforme de distribution de contenu en ligne, de gestion des droits et de communication développée par Valve et disponible depuis le 11 septembre 2003.
[4] L'Epic Games Store (aussi appelé Epic Games Launcher ou Boutique Epic Games) est une plateforme de distribution de contenu en ligne et de gestion des droits développée par Epic Games et disponible depuis le 6 décembre 2018.

B » ou AA, de la même façon que la série B pour les films. *[Voir ref 1.3, Annexe 2, Page 63 : Définition d'un jeu AAA]*

6. Unreal Engine

Unreal Engine est un moteur de jeu vidéo propriétaire développé par Epic Games et est disponible sur l'Epic Games Store. Les principaux concurrents de ce moteur sont Unity développé par Unity Technologies et le CryEngine développé par Crytek.

Figure 1.1 : Logo Unreal Engine

Un moteur de jeu est un ensemble de fonctions, permettant le développement d'un jeu, avec accès à des fonctions haut-niveau directement (de la sorte, on masque toutes les opérations bas-niveau). De plus, il doit fournir une base, sur laquelle le programme est capable de tourner de manière autonome, jusqu'à ce que son arrêt soit requis. On parlera alors de *squelette*, qui sera complété par tout jeu utilisant le moteur. Le moteur de jeu doit permettre de faire abstraction de toutes les contraintes liées à l'environnement, et ainsi faciliter le développement de jeux par la suite. En résumé, l'utilisateur du moteur (le développeur) doit se contenter de fonctions simples (chargements, rendus, mises à jour...). Cependant, plus le moteur de jeu rend l'accès haut-niveau, moins l'utilisateur a de contrôle dessus (à moins de modifier directement le moteur). *[Voir ref 1.4, Annexe 2, Page 64 : Définition d'un moteur de jeu]*

II. L'entreprise d'accueil, son historique, nature de l'activité de l'entreprise

1. Période d'avant stage, choix, motifs et recrutement

Comme tout joueur de jeu vidéo et passionné par l'art d'en créer pour soit même, il était évidant pour moi d'en faire mon métier, c'est pourquoi ce stage de fin d'étude est une excellente opportunité pour moi de me lancer pleinement dans une carrière dans ce domaine.

Il n'existe pas beaucoup de projets dans ce sens en Tunisie, mais il y en a cependant quelques-uns qui sont prometteurs.

Lanterns Studios était mon premier choix car je connaissais quelques-uns des membres de l'équipe et pour moi, m'intégrer le plus facilement possible était mon motif principal pour ma première expérience en entreprise.

2. Présentation de l'entreprise

Lanterns Studios, ou brièvement appelé : *Lanterns*, est un studio de développement de jeux vidéo indépendant qui offre également des applications de réalité augmentée/virtuelle et des services de MoCap[5]. Lanterns est composé d'une équipe hautement qualifiée dans divers domaines liés à l'informatique, à la CG[6], à l'image de marque et au contenu multimédia, du développement d'applications mobiles aux expériences AR/VR[7] du PC aux jeux vidéo sur console. *[Voir ref 1.5, Annexe 2, Page 64 : A propos de Lanterns]*

Lanterns Studios (SARL) est co-fondée en Janvier 2020 par 3 co-fondateurs. Tout d'abord, Jihed Jaouabi, CEO[8] et game designer ; ensuite Nabil Brahem, COO[9] et responsable du level design et des animateurs ; et finalement, Oussama Ben Myriam, CTO[10] *[Voir la sous-section : études des rôles du chapitre 2, Page 28]* et son siège social se situe à la Charguia 1 de Tunis en Tunisie. Lanterns compte à ce jour une dizaine de salariés.

[5] Motion Capture est une technique permettant d'enregistrer les positions et rotations d'objets ou de membres d'êtres vivants (acteurs) pour en contrôler une contrepartie virtuelle sur ordinateur.
[6] Computer Graphics.
[7] Réalité augmenté / Réalité virtuelle.
[8] Chief Executive Officer ou Directeur générale en français.
[9] Chief Operating Officer ou Directeur de l'exploitation en français.
[10] Chief Technical Officer ou Directeur technique en français.

Leur logo est une typographie manuscrite du mot « Lanterns » en blanc sur fond rouge écarlate avec une lanterne accrochée sur la lettre T du mot Lanterns.

Figure 1.2 : Logo Lanterns Studios

Sans compter l'administration et le top management, l'entreprise est subdivisée en 5 départements distincts :

Le **premier** est le département **Design**, c'est un département créatif qui est en charge des concepts, de la jouabilité et des améliorations du jeu vidéo.

Le **second** est le département **3D**, qui est en charge de la création des personnages, des objets et du monde en 3D. Il est responsable de tout ce que nous voyons à l'écran.

Figure 1.3 : Lanterns MoCap

Le **troisième** est le département **Animation**, qui est en charge de rendre vivant les personnages et le monde qui entoure le joueur. Le rôle de ce département est de capturer les mouvements des acteurs grâce au MoCap et de les retravailler sur ordinateur mais aussi d'en créer à partir de zéro.

Le **quatrième** département est le département **technique** qui s'occupe de la programmation des jeux vidéo et de l'intégration des différents objets en 3D et animations grâce à Unreal Engine. Leur rôle est par la suite de corriger les bugs[11] présents dans le jeu. Nous pouvons aussi inclure la création des effets sonores et musiques.

Le **cinquième** et dernier département est le département **service et sous-traitance** qui traite des projets mineurs et rapides. Officiellement, personne n'est rattaché au département mais on alloue des ressources des autres départements dès qu'un projet mineur en vaut la peine.

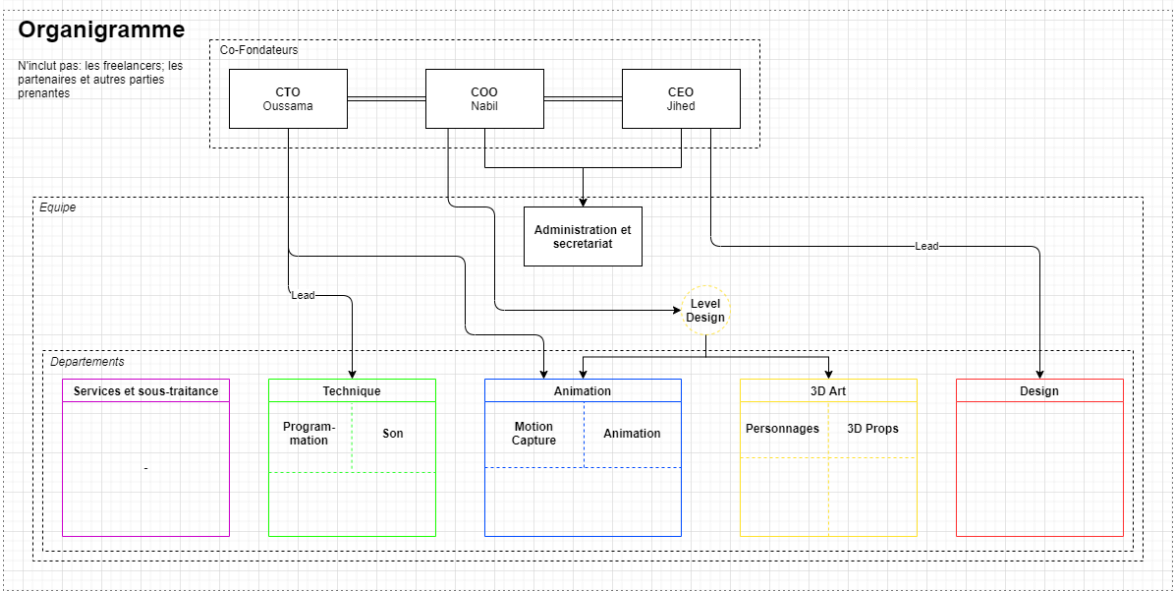


Figure 1.4 : Organigramme de Lanterns [voir Annexe 4]

3. Description des services offerts et des projets en cours

Comme cité précédemment, Lanterns est une entreprise de développement de jeux vidéo, et plus précisément, une indie game company et ils offrent plusieurs services :

[11] Est un défaut de conception d'un programme informatique à l'origine d'un dysfonctionnement.

Premièrement, ils sous-traitent quelques projets (toujours des jeux) pour des clients mineurs. Ensuite, Lanterns loue aussi leur studio de MoCap, mais leur projet principal est un jeu AAA qui comme le terme l'indique bien, c'est un projet de grande taille.

Rare sont les indies[12] qui se lancent dans un très grand projet, et ce, pour plusieurs raisons : et le premier étant qu'un jeu de taille B nécessite beaucoup de temps et de compétences dans divers domaines différents, alors imaginez, un jeu AAA est infiniment plus complexe. Mais leur inspiration principale en termes de business model reste HellBlade, un jeu sorti en Aout 2017 sur PC et console et est développé par Ninja Theory. *[Voir ref 1.6, Annexe 2, Page 64 : HellBlade, un indie AAA]*

Leur jeu AAA se nomme pour l'instant *The Edge* et a déjà réussi à dégager un financement. C'est un jeu de type narratif ou l'histoire a le rôle le plus important dans l'expérience du joueur contrairement à d'autres jeux qui privilégient le gameplay[13]. Le projet est toujours en phase de prototypage mais nous verrons plus tard dans les prochains chapitres, les phases de développement d'un jeu vidéo.

III. Service d'affectation et Lieu de travail

N'ayant pas de service en charge du management du projet, je n'ai été affecter à aucun service ou département en particulier, cependant mon rôle était déjà défini dès la mise en valeur de la problématique que nous évoquerons dans la section suivante de ce chapitre.

Pour être plus précis, mon rôle au début est de collaborer avec le top management afin de faire une analyse de la situation actuelle de Lanterns afin de dégager une problématique. Notre point de départ était d'avoir un regard nouveau sur l'entreprise. Ainsi, de par la première impression ET les premières discussions qui sont souvent révélateurs des besoins imminents de l'entreprise en termes de gestion, nous avons pu dégager une problématique.

Donc en d'autres termes, je n'ai pas été affecter à aucun département ou service mais mon rôle été défini depuis le début, à savoir : analyser le fonctionnement de Lanterns et faire un état des lieux, comme le ferai un consultant externe par exemple.

[12] Indie game companies (au pluriel).
[13] La jouabilité en français.

Lanterns a pour particularité d'être un open space[14], qui a fortement aidé à avoir un regard clair sur les différents membres de l'équipe. L'espace est jugé agréable par la totalité des salariés.

Le studio dispose (les plus importants) :

- Un espace ouvert pour tous les développeurs avec une capacité de 22 développeurs
- Le studio de MoCap
- Une salle de réunion
- La direction.

Figure 1.5 : Vue sur l'espace de travail

[14] Un espace ouvert (≠ espace fermé ou clôt) permettant de faciliter les échanges entre les équipes.

IV. Diagnostique et problématique

Lors de nos premiers échanges avec le CEO de Lanterns, nous avons longuement discuté de Lanterns et de son fonctionnement et de notre future contribution. Etant donné que tous les 12 membres de l'actuelle équipe de Lanterns ont un background « technique », notre rôle été dès lors très important.

Parmi les premières informations dont nous avons été disposés de connaitre s'était : L'ancienne pipeline de Lanterns, celle-ci décrivait le fonctionnement d'une entreprise de développement de jeux vidéo et a été élaboré dès la genèse de Lanterns. A l'époque, ils n'étaient que 3, voir une demi-douzaine à être chez Lanterns. *[Voir Annexe 5 : Ancienne pipeline]*

Après avoir entendu le point de vue du CEO, on nous a été confier plusieurs tâches, dont :

1. A l'écoute du top management

Ma première, et évidente, tâche été d'être à l'écoute des deux autres co-fondateurs afin d'essayer de comprendre le fonctionnement de Lanterns. D'après le CEO, le pipeline initial n'est pas conforme à ce qui se passe réellement chez Lanterns de nos jours, qu'on passe d'une étape a une autre et qu'elle été très linéaire dans sa conception, mais que chaque nœud avez tout de même un sens.

Premièrement, une entrevue avec le CTO ; et il m'a expliqué chaque nœud du pipeline, le processus actuel et le processus au lancement et m'a conseillé de regarder les making-offs[15] de HellBlade. Les making-offs regroupent quelques aspects du développement du jeu, qui, est très proche de ce que Lanterns veut faire. Il m'a aussi expliqué que les programmeurs testent sur des placeholder[16] car l'avancement du game art (3D et animation) est trop lente par rapport à la partie technique (programmation). Il m'a aussi parlé du versionning[17] du game art qui semble être un gros problème, il souhaite une solution comme *Git* pour le game art. Il m'a parler de *shotgun* qui est une solution qui semble être adéquate pour une entreprise de développement de jeux vidéo.

[15] Plus exactement le dev diary de hellblade, qui est, entres autres, un journal de développement
[16] Quelque chose utilisé ou inclus temporairement ou en remplacement de quelque chose qui n'est pas connu ou doit rester générique ; ce qui tient, désigne ou réserve une place à quelque chose à venir plus tard. Exp : un mannequin tout blanc en attendant que le personnage a fini d'être créé.
[17] Enregistrer (stocker) chaque version du processus de création d'un modèle 3D ou animation.

Pour mieux comprendre ce qu'est un logiciel de versionning fait, une petite définition s'impose :

Un logiciel de gestion de versions (ou VCS en anglais, pour version control system) est un logiciel qui permet de stocker un ensemble de fichiers en conservant la chronologie de toutes les modifications qui ont été effectuées dessus. Il permet notamment de retrouver les différentes versions d'un lot de fichiers connexes.

Il existe aussi des logiciels et services de gestion de versions décentralisé (distribué) (ou DVCS en anglais, pour distributed version control system). *Git* et *Mercurial* sont deux exemples de logiciels de gestion de versions décentralisés et sont disponibles sur la plupart des systèmes Unix et Windows, le plus populaire et de loin reste *Git*.

Figure 1.6 : Logo de git

Les logiciels de gestion de versions sont utilisés notamment en développement logiciel pour conserver le code source relatif aux différentes versions d'un logiciel. *[Voir ref 1.7, Page 64 : Définition du versionning]* Mais Lanterns a pour particularité d'utiliser *perforce*, qui est une autre solution pour le versionning. Le problème, c'est que le game art est difficile à versionner du fait de la taille démesuré des fichiers que, aucun de ces logiciels ne supporte.

Quant au COO, il m'a expliqué le processus du level design *[Voir la sous-section : études des rôles du chapitre 2, Page 28]*, qui est une phase importante du process d'implémentation. C'est en quelque sorte la pierre angulaire de la phase de développement du jeu.

Il m'a aussi expliqué que le level design est en pratique une étape parallèle, voir communicante aux autres qui, comme nous l'avons vue dans l'ancienne pipeline *[Voir Annexe 5]* et qu'on le verra dans le chapitre 3 et dans l'annexe 6, n'est pas du tout le cas.

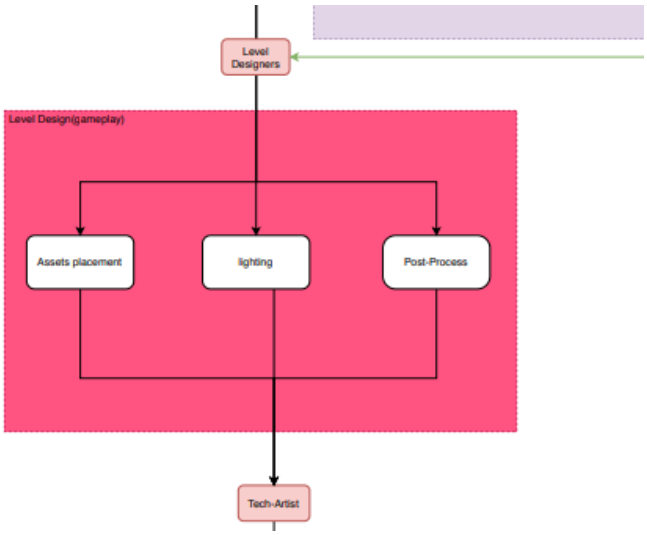


Figure 1.7 : Level design comme étape finale dans l'ancien pipeline

Pour mieux comprendre l'ancien pipeline, expliquons tout d'abord chaque type de nœuds et colorimétrie présents :

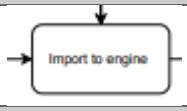
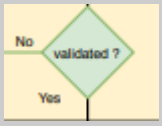
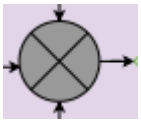
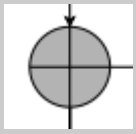
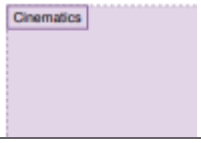
Type de nœud		Définition
	Rectangle rouge	Représente le rôle qui va exécuter la tâche suivante selon la/les tache(s) entrantes.
	Rectangle blanc	Tâche atomique.
	Losange vert avec le mot « validated ? » et deux sorties : yes/no	Processus de validation de tâche, elle a une sortie dépendamment de si elle est validée ou pas.
	Cercle avec le symbole « × »	Regroupement de tâches en AND[18], la tâche de sortie ne s'exécute que si toutes les tâches en entré sont finies.
	Cercle avec le symbole « + »	Regroupement de tâche en OR[19], la tâche de sortie ne s'exécute que si l'une des les tâches en entré est fini.
	Regroupement de nœuds	Regroupe des processus en phases ou en départements exécutant les tâches.

Tableau 1.1 : Explication de l'ancien pipeline

[18] Porte logique ET (AND en anglais) : Elle comporte généralement deux entrées et une sortie, pour que la sortie soit au niveau logique 1, il faut que les deux entrées soient à 1. Dans le cas contraire, la sortie est à 0

[19] Porte logique OU (OR en anglais) : Elle comporte généralement deux entrées et une sortie, pour que la sortie soit au niveau logique 1, il faut que au moins une des entrées soit à 1.

2. Suivi des tâches et du projet

Lanterns a pour particularité d'utiliser ClickUp comme plate-forme (logiciel) de gestion de projet.

Le logiciel de gestion de projet est un logiciel utilisé par un large éventail d'industries pour la planification de projets, l'allocation des ressources et la planification. Il permet aux chefs de projet comme à l'ensemble des équipes de maîtriser leur budget, la gestion de la qualité et toute la documentation échangée tout au long d'un projet. Ce logiciel sert également de plate-forme pour faciliter la collaboration entre les parties prenantes du projet.

Du suivi des livrables à la gestion des ressources et de la gestion du budget à la collaboration avec les membres de l'équipe, il y a beaucoup à prendre en compte lors de l'exécution et de la gestion de projets. Cela est également vrai lors du choix du bon outil logiciel de gestion de projet. Donc, voici les 5 principaux aspects fonctionnels des logiciels de gestion de projet.

- **Listes de tâches :** C'est être capable d'attribuer et de mettre à jour le statut des tâches afin que tout le monde de l'équipe soit sur la même longueur d'onde, cela inclut aussi le backlog[20].
- **Scheduling** : Comme des calendriers, des diagrammes de Gantt ou des outils de jalon qui aident à comprendre où une tâche s'intègre dans le projet dans son ensemble et combien de temps il reste pour la terminer.
- **Partage de fichiers :** C'est la possibilité de partager et d'organiser des documents de projet clés élimine le temps perdu à rechercher des fichiers. Les fichiers sont souvent liés (rattachés) aux tâches.
- **Communication :** Ceci est essentiel dans la gestion de projet car un flux de communication fluide signifie une résolution rapide et facile des problèmes. Nous pouvons ainsi citer les commentaires sous chaque tâche, les descriptions de tâches ou bien un chat.
- **Reporting :** Ceci est important pour tous les membres de l'équipe lorsqu'il s'agit de se mettre à jour sur le projet dans son ensemble. Cependant, c'est aussi un énorme avantage pour les chefs de projet qui veulent s'assurer que le projet avance et que les tâches sont exécutées dans les délais. Des graphiques et des métriques situés dans le dashboard[21] sont des outils puissants.

[Voir ref 1.8, Annexe 2, Page 64 : Logiciels de gestion de projets]

[20] Liste de tâches priorisées définissant les caractéristiques d'un produit. Il est un des éléments fondamentaux de la méthodologie Scrum.
[21] Panneau de contrôle

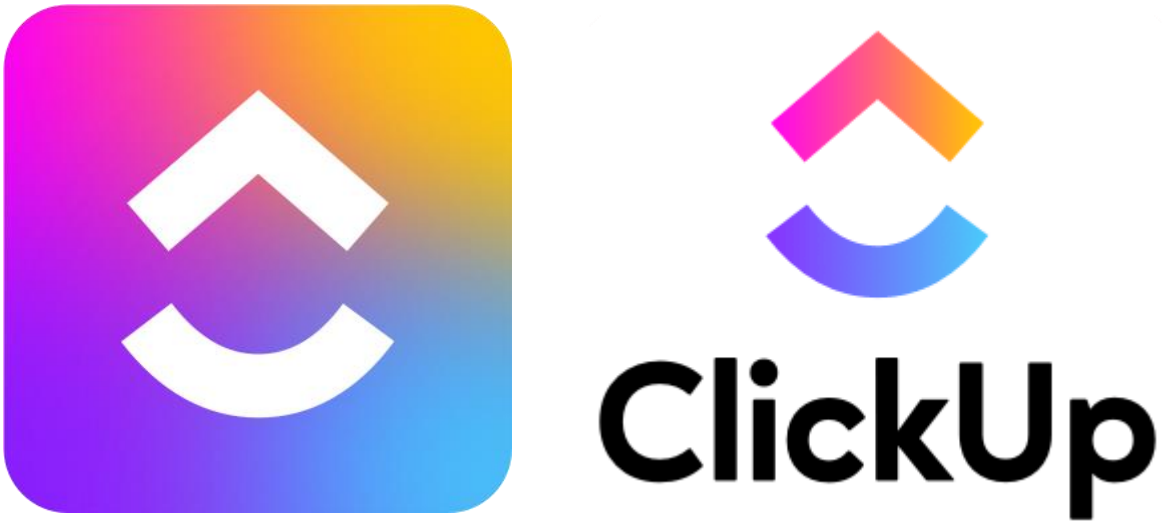


Figure 1.8 : Les logos de ClickUp

La plate-forme ClickUp permet aux utilisateurs d'utiliser plus de 100 fonctionnalités qui incluent : des listes de tâches, attribuer des commentaires, divers diagrammes dont celle de Gantt, automatiser des tâches récurrentes, synchroniser avec Google Agenda, différentes possibilités de tri, personnaliser les affectations, collaboration avec d'autres projets/utilisateurs, maquettes d'images, affecter plusieurs personnes aux tâches, commentaires filetés, slash command[22], barre d'outils multitâche, édition enrichie, extension Chrome, attribution de priorités de tâches, différentes vues, pourcentage d'avancement du projet, hiérarchie des tâches, notifications personnalisées, flux d'activité des utilisateurs, mentions et bien plus de fonctionnalités. *[Voir ref 1.9, Annexe 2, Page 64 : Fonctionnalités de ClickUp]*

Notre rôle est, dans un deuxième temps, d'analyser la progression de leur(s) projet(s) grâce à ClickUp. Leur projet (le jeu AAA) est structuré comme suit :

- La première couche est celle du projet global, il y inclue une timeline[23] globale.
- En dessous, chaque département à son propre code couleur avec sa propre timeline. Par exemple : Jaune pour le département technique. Nous notons aussi que le son n'est pas inclus dans le département technique pour soucis de visibilité.
- Chaque département est décomposé en processus ou tâches majeures. Une tâche majeure est souvent nommée sous cette forme : CHAPITRE 1 : Animation personnage X, avec le rouge illustrant que c'est une tâche de type : Animation.

[22] Les commandes slash sont des messages qui commencent par / et déclenchent une requête HTTP vers un service Web qui peut à son tour publier un ou plusieurs messages en réponse
[23] Ligne de temps pour représenter l'état d'avancement du projet/tâche

- Les tâches majeures sont souvent précédées du numéro du chapitre auquel elle y sera implémentée.
- Les tâches majeures peuvent ensuite être décomposés en sous-tâches.

Tous cela sera visible dans plusieurs vues, comme par exemple, le diagramme de Gantt.

Etant donné que nous sommes arrivés dans un moment assez critique de Lanterns vu que l'équipe est mobilisée pour sortir une démonstration du jeu lors des jours qui viennent, les tâches sont parfois ajoutés le jour-même et qu'ils travaillent au jour le jour.

Conclusion

Nous avons notamment vite compris qu'il faut comparer le workflow présent actuellement chez Lanterns avec le pipeline fait dès les premiers jours de la fondation de la jeune société. En effet, établir un pipeline clair permet de résoudre le flou qui entoure le workflow de la société ainsi que d'avoir une vision fixe de comment sortir, tout d'abord, un jeu vidéo, mais aussi, LEUR jeu vidéo, et cette nuance est radicalement fondamentale pour la conception d'un pipeline qui réponds à ces questions :

- Est-ce qu'elle est assez claire pour n'importe quelle recrue ?
- Est-ce qu'elle est conforme aux expériences que, des sociétés ayant déjà développé des jeux, ont expérimentés ?
- Est-ce qu'elle est conforme au business model et à l'essence même du produit final ?
- Est-ce qu'elle est adaptable a tous types de projets (jeux vidéo) ?

Mais aussi, nous devons aussi travailler sur tous les points liés au workflow et notamment, le versionning des fichiers et ressources.

Chapitre **Deuxième**

Déroulement du stage

Introduction au chapitre

Ce chapitre est dédié au déroulement du stage à proprement parler, le travail effectué en profondeur et les méthodes de recherche utilisées afin de répondre aux questions soulevées par la problématique mentionnée ci-précédemment.

I. Hypothèse de recherche

Comme nous l'avons mentionné dans le précèdent chapitre, la meilleure approche était de communiquer, mais pas seulement avec le top management, pour comprendre chaque rôle au sein de Lanterns.

Comprendre les rôles de chaque membre va nous permettre de comprendre le processus créatif aboutissant à un jeu vidéo. L'ancien pipeline est un bon outil de démarrage mais beaucoup trop complexe a première vue. La meilleure façon de le savoir, c'est de le simuler dans un cas réel et de savoir, quels sont les nœuds qui poseront un problème. Même si d'après leurs dire, ils n'utilisent plus le pipeline, ça sera aussi un moyen de savoir s'ils devront plutôt l'utiliser.

Si un des nœuds du pipeline se retrouve à bloquer les suivantes, alors il faut reconstruire cette dernière, mais pour cela il nous faut un certain nombre d'information à assimiler.

Premièrement, il faut étudier chaque rôle au sein de Lanterns, mais pour que notre compréhension des rôles soit conforme à l'industrie du jeu vidéo, il faut aussi faire une fiche des postes majeurs et comprendre ce qui se fait réellement dans d'autre studios plus expérimentés et essayer de lier les deux visions.

Ensuite il faut décortiquer les phases de développement d'un jeu vidéo. En effet, cette dernière est essentielle dans le processus de création du nouveau pipeline car le processus de création d'un jeu est tout sauf un processus mécanique et statique. Comme nous le verrons plus tard, c'est un processus dynamique et très changeant. Les premières phases de développement n'ont strictement rien à voir avec les dernières.

Plus qu'un travail théorique, il s'agit d'un travail de recherche et de structuration qui durera tout le long du stage.

En pratique, ce rapport est le guide de Lanterns.

II. Activités menées

1. Simulation du pipeline

Pour simuler un process il faut utiliser un outil qui décrira au mieux le comportement de cette dernière.

Nous partons du principe que les tâches dans la partie « technique » coûtent moins cher en temps de développement que la partie « artistique », donc nous distinguons une claire asynchronie des deux parties qui semblent parallèles.

Nous pouvons très clairement « couper » le pipeline en deux parties :

- Programming pipeline (partie programmation)
- Level Art pipeline (partie 3D, animation et tout le reste)

[Voir Annexe 5]

Même si Lanterns ne l'utilise plus dans cette forme-ci, nous devons quand même critiquer cette dernière pour pouvoir apporter des modifications si besoin est.

Nous pouvons dès alors distinguer 3 anomalies :

La première est que les deux parties se rejoignent dans un seul et unique point alors qu'on nous a bien expliqué au paravent que la partie *Level Art* prenait beaucoup plus de temps que la partie *Programming*.

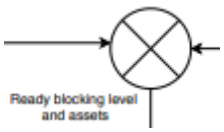


Figure 2.1 : Regroupement des deux parties

La seconde est que la partie Programming semble beaucoup moins grande et beaucoup plus linéaire que la partie Level Art ce qui, peut-être, explique la différence du temps de production. *[Voir Page suivante]*

Et enfin, la troisième est que la partie Level Design semble être en fin de production ce qui n'est pas en adéquation avec ce qui se fait chez Lanterns en ce moment-là. De plus, ce pipeline-là semble être incomplet vu que nous n'avons aucune information sur ce que

signifie le dernier nœud, et la question est : *Que va faire le game designer avec tout ce travail effectué ?*

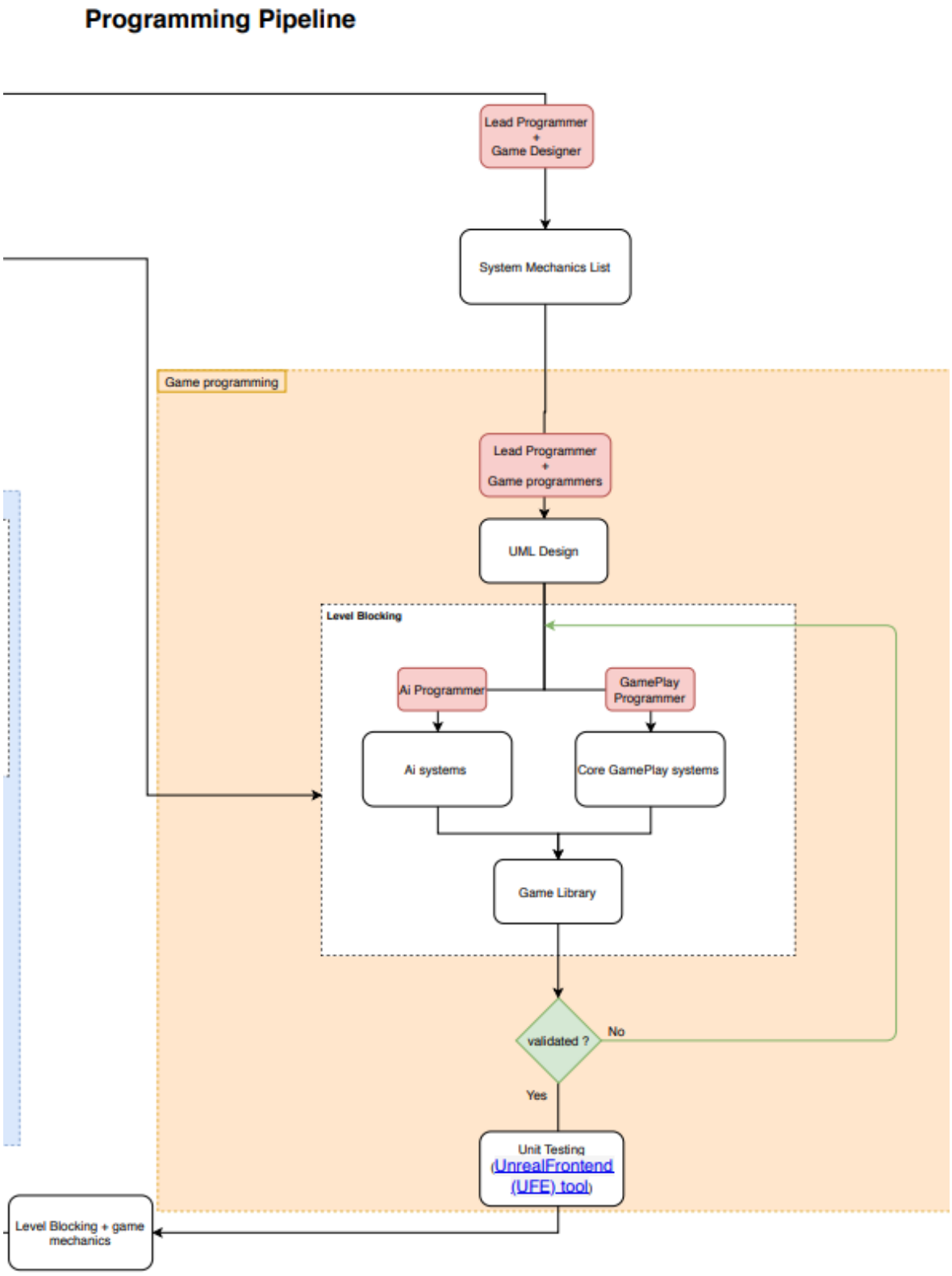


Figure 2.2 : Partie « Programming »

Pour revenir à cet outil, nous avons utilité un outil informatique qui s'appelle *Machinations*. Machinations est un outil en ligne qui permet de simuler des idées sous forme de diagramme, cet outil est spécialisé dans la création de systèmes pour les jeux vidéo avec divers outils mathématiques et portes logiques. Le but, c'est qu'avec des paramètres en entrés choisis, nous nous « amusons » à faire traverser ces paramètres dans divers convertisseurs, portes et échangeurs afin d'avoir le résultat souhaité.

Figure 2.3 : Logo Machinations

L'outil transferts des *tokens*[24] de nœud en nœud et ces tokens représentent chacune, une charge de travail. Même si cette donnée est difficile à quantifier, malheureusement l'outil ne nous permet pas de choisir la fréquence de travail de chaque nœud et donc, tous les nœuds ont la même charge de travail. Mais même si c'est un cas particulier, notre but ici c'est de comparer les deux parties du pipeline et savoir si cette dernière bloque à un certain stade.

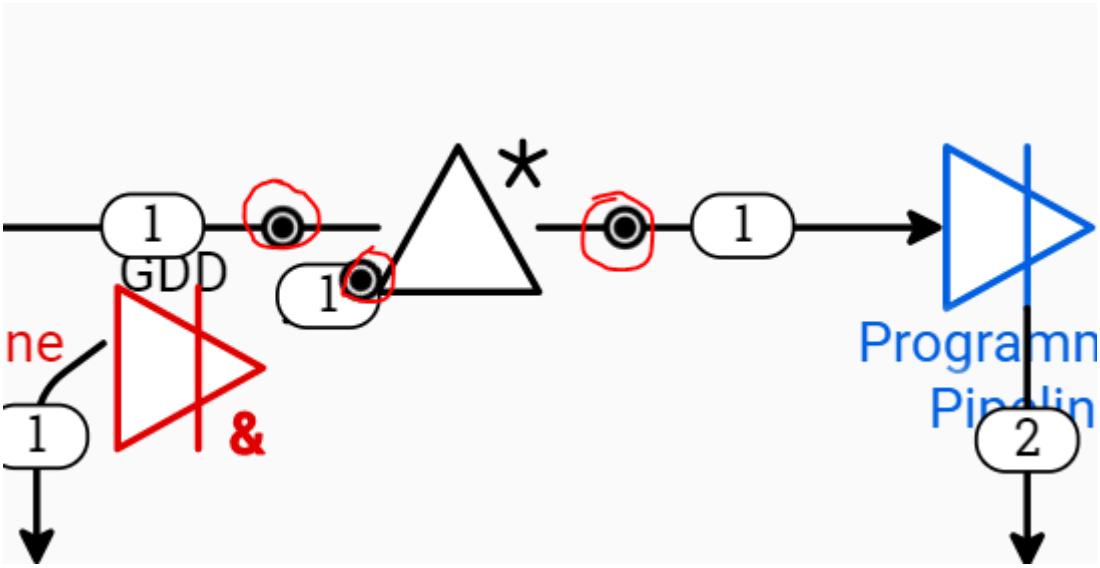


Figure 2.4 : Représentation des tokens encerclé en rouge

[24] Jetons en français

2. Expériences communes

a. Questionnaire et Compréhension des rôles.

A la suite de la simulation du pipeline, dans l'optique de comprendre les rôles au sein de Lanterns et comme nous l'avons fait pour le top management, nous avons interviewer tous les autres membres de Lanterns un par un et nous les avons soumis à un questionnaire. Cette initiative permet de situer chacun des membres de Lanterns par rapport aux autres (top management y compris) et de détecter d'éventuels conflits de vision au sein de l'entreprise. Nul autre que ceux qui comprennent au mieux le métier peut nous guider dans notre projet de compréhension des rôles.

[Voir Questionnaire à la page suivante]

Le questionnaire est divisé en 4 types de question :

- 6 Questions sur l'environnement
- 11 Questions sur les tâches qu'ils font aux quotidiens (forme et fond)
- 7 Questions de nature psychotechnique
- 5 Questions sur leur vision qui se font de leur rôle au sein de Lanterns

Vous noterez que des questions sur leur rôles au sein de Lanterns sont minoritaire par rapport aux autres mais nous avons dissocier leur vision subjectif (en noir) de leur tâche quotidienne (en vert) qui est plus concret et objectif.

Les types de questions sont aussi variés, passant de la question directe et sans options à la question avec choix multiples ou double (exp Oui/Non) à la question de tri à la question d'opinion.

Ces dernières sont divisées en 7 niveaux d'opinions

- **HD** → Highly Disagree, Fortement pas d'accord en français.
- **D** → Disagree, Pas d'accord en français.
- **SD** → Slightly Disagree, Plutôt pas d'accord en français.
- **N** → Neutral, Neutre en français.
- **SA** → Slightly Agree, Plutôt en accord en français.
- **A** → Agree, En accord en français.
- **HA** → Highly Agree, Fortement en accord en français.

Lanterns STUDIOS

Q1. Quel est votre Rôle au sein de Lanterns ?

Q2. Pour votre rôle, Quels sont les inputs/outputs ?

Q3. Quand vous recevez une tache, généralement, est ce que vous collaborez avec un autre membre de l'équipe ? Si oui, comment ?

Q4. Mis à part vos taches régulières, est ce que vous vous voyez exercer un autre rôle au sein de Lanterns

Q5. Parlez-moi de l'environnement de travail ? Les plus et les moins ?

Q6. Supposons que Nabil et Oussama ne sont pas là pour une durée indéterminée ? Comment vous validez une tache ?

Environnement (6) – Tache (11) – Psychotechnique (7) – Rôle (5)

Figure 2.5 : Questionnaire page 1

Q7. Même question, Si vous avez besoins d'une ressource lié à votre rôle (ex : fichier), comment vous l'obteniez ? et aimerez-vous l'obtenir différemment ?

Q8. S'il existe un moyen différent pour obtenir/envoyer un fichier, quels sont les priorités ?

□ Organisation (Arbres, Versions, Qui ?)

□ Communication (Commentaires, Notifications, Visibilité du fichier source avant l'obtention)

□ Autre :

Q9. Quels sont les informations que vous connaissez à l'avance quand vous obtenez une tache ?

□ Nom	□ Durée	□ Complexité
□ Description	□ Affectation	□ Sous-Taches
□ Dépendance	□ Tags	□ Piece-jointe

Q10. Je suis plus organisé en arrivant chez Lanterns

HD	D	SD	N	SA	A	HA

Q11. Quand j'ai une idée, je ne vous censure jamais

HD	D	SD	N	SA	A	HA

Q12. Les taches reçues sont claires

HD	D	SD	N	SA	A	HA

Q13. J'ai souvent plusieurs taches en parallèle

HD	D	SD	N	SA	A	HA

Q14. J'ai souvent des taches en attente

HD	D	SD	N	SA	A	HA

Q15. Depuis mon embauche, j'ai évolué techniquement et professionnellement.

HD	D	SD	N	SA	A	HA

Comment ?

Q16. J'ai besoin d'avoir une vision globale du projet.

HD	D	SD	N	SA	A	HA

Q17. Dans l'état actuel, J'ai besoin de déléguez quelques taches

HD	D	SD	N	SA	A	HA

Q18. Je vais volontiers vers les autres

HD	D	SD	N	SA	A	HA

Q19. Je suis plutôt minutieux

HD	D	SD	N	SA	A	HA

Environnement (6) – Tache (11) – Psychotechnique (7) – Rôle (5)

Figure 2.6 : Questionnaire page 2

Q20. Avez-vous effectué un test psychologique comme 16personalitites.com ? Acceptez-vous d'en faire un sachant que le résultat a de forte chance d'être erroné ?

□ Oui □ Non

Type :

Q21. Un game Producer qui gère les taches, mais constamment sur mon dos me gêne.

HD	D	SD	N	SA	A	HA

Q22. Parmi ces points, les quelles sont les plus importants pour vous ? classez-les de 1 à 5.

□ Briefing Quotidien

□ Vue clair sur le planning

□ Formations sur des sujets techniques

□ Ice breaking hebdomadaires

□ Brainstorming / Focus group régulier

Q23. J'apprécie de travailler à l'aveugle (task focused only)

HD	D	SD	N	SA	A	HA

Q24. La pression aujourd'hui est plus forte qu'il y a quelques semaines.

HD	D	SD	N	SA	A	HA

Q25. Je gère aussi bien la pression chez Lanterns qu'avec n'importe quel aspect extérieur a Lanterns

HD	D	SD	N	SA	A	HA

Q26. Qu'est-ce qu'une tache finie pour vous ?

Q27. Sacrifierez-vous l'organisation si les deadlines sont très proches ?

Q28. Le niveau d'exigence de mes supérieurs change selon le contexte

HD	D	SD	N	SA	A	HA

Quand ?

Q29. Parmi toute vos taches, quels sont celle avec un niveau d'exigence élevé ? Et pourquoi ?

Environnement (6) – Tache (11) – Psychotechnique (7) – Rôle (5)

Figure 2.7 : Questionnaire page 3

b. Activité de recherche : études des rôles en sein des différentes entreprises de jeux vidéo et leurs prérogatives

Mis à part le questionnaire, nous avons pris le temps d'étudier les rôles et emplois majeurs qui constituent un studio de développement de jeux vidéo.

Il faut savoir qu'un studio de développement de jeux vidéo peut nécessiter divers rôles selon la situation et l'idée de base, mais certains rôles sont tout de même constamment présents dans n'importe quel studio. Le titre peut être mal compris ici, il ne s'agit pas d'étudier des entreprises bien spécifiques, mais d'extraire les rôles qui sont et seront toujours présent peu importe le studio.

Au total, il y a 9 rôles majeurs commun à tous studios de développement de jeux vidéo et ces rôles sont :

Game designer

Conçoivent le gameplay. Elaborent et créent les règles et la structure du jeu.

Lead designer

Coordonne le travail des autres designers et c'est celui qui dicte la vision du jeu. Il assure la communication entre les équipes et prends aussi de lourdes décisions en matière de game design.

Le lead designer peut être le fondateur de la société.

Artiste

Un artiste de jeux est un artiste visuel qui crée du « game art » et dessine pour le studio. La production artistique est généralement supervisée par la direction artistique ou le responsable artistique (lead game art).

Programmeur

Un programmeur de jeux est un ingénieur logiciel qui, principalement, développe des jeux vidéo ou des logiciels en relation aux jeux vidéo, comme par exemple : des outils pour les développeurs.

Un programmeur peut être spécialisé un rôle en particulier :

- Intelligence artificielle
- Graphisme
- Jouabilité
- Scripting[25]
- Interface du jeu
- Input processing[26]
- Réseaux et communications
- Création d'outils pour les développeurs

Ingénieur Son

Responsable des effets sonores et du positionnement du son dans l'espace. Parfois supervise le jeu d'acteur et tout ce qui est en relation avec la création d'assets[27]

Les compositeurs quant à eux, créent de la musique pour le jeu, mais cette tâche en particulier est souvent externalisée par des freelances.

QA/Testeur[28]

C'est l'assurance qualité du produit et elle est souvent gérée par des testeurs de jeu. Leurs rôles principaux sont :

- Le contrôle qualité
- Le listing des bugs

Level designer

C'est des game designers qui créent l'environnement du jeu, les niveaux et les scénarios en utilisant un éditeur de level design[29]. Ils travaillent souvent le niveau du tout début de la préproduction jusqu'à son achèvement. Ils travaillent avec des versions complètes et incomplètes du jeu (souvent très incomplètes).

[25] Création de scénarios.
[26] Gestion des entrées (comme la frappe du clavier ou de la souris par exemple).
[27] Asset est un terme technique qui décrit une ressource simple crée comme une table en 3D ou un effet sonore.
[28] QA : Quality Assurance ou Assurance Qualité en français.
[29] Très souvent, un niveau est créé dans le moteur de jeu lui-même, mais on peut aussi créer un outil au sein même du moteur qui renforcera se dernier. Un éditeur de niveau est souvent au cas par cas (type de jeu).

Les programmeurs produisent les éditeurs de niveaux et créent des outils pour que les level designer puissent développer tous leurs potentiels créatifs. Et cela permet aussi d'éliminer le besoin de modifier le code source par les level designers.

Ils travaillent souvent avec des placeholders et des prototypes et visent notamment la cohérence des niveaux et une disposition des objets propres dans le niveau avant que les illustrations requises ne soient produites par les artistes.

Artiste Technique

Agissent comme lien entre les artistes et les programmeurs. Ils sont en quelque sorte un hybride entre les artistes et les programmeurs car ils peuvent faire les deux en même temps puisqu'ils ont les deux compétences.

Chaque entreprise peut avoir une idée différente de ce que sont les rôles et les responsabilités d'un artiste technique, cependant, leur principal objectif est d'intégrer les assets artistiques dans un jeu sans enlever la vision artistique ni dépasser les limites techniques du moteur de jeu.

Producer (Chargé de production / Producteur)

Est la personne la plus haute en charge qui supervise le développement d'un jeu vidéo.

Il est responsable de la livraison dans les délais et de la qualité finale du jeu.

Pour les petits jeux, le producteur peut interagir directement avec le staff en charge de la programmation et le staff créatif. Pour les jeux plus importants, le producteur demandera l'aide du lead programmer, du lead art, du lead game designer et de l'assurance qualité.

Ses rôles principaux sont :

- Négocier le contrat, y compris les accords de licence.
- Assurer la liaison entre l'équipe et les parties prenantes (éditeurs ou conseil d'administration).
- Élaboration et tenue des échéanciers et des budgets.
- Assurer la livraison en temps voulu des livrables et des milestones[30].
- Planification du QA en temps opportun.

[30] Jalons.

- Gestion des bêta-tests et des groupes de discussion (focus group).
- Présentation d'idées de jeux aux éditeurs (pitching).

3. Analyse des pratiques et outils

a. Méthodologie de travail

La méthodologie de travail est le processus de développement / séquence d'étapes ordonnées pouvant aboutir à un logiciel complet, une application web ou mobile, une extension d'un logiciel existant ou la modification d'un système existant et même la création d'un jeu. Le but est de produire en ayant un certain standard qualité et qui doit répondre aux différents besoins des utilisateurs finaux tout en respectant les coûts en temps de développement.

Voici une liste des différentes méthodologies de travail pour des projets informatique :

Méthodologie	Description	Avantages	Inconvénients
Waterfall	Comme son nom l'indique, l'approche en cascade suit la logique d'une chute d'eau. Une fois que l'eau a dévalé le flanc de la montagne, elle ne peut plus remonter, mais seulement continuer son chemin. Ainsi, dès qu'une étape du projet est terminée, l'équipe passe à l'étape suivante ; il n'y a pas (ou peu) de retour en arrière. L'idée est d'avancer naturellement, étape par étape, jusqu'à atteindre l'objectif final en suivant une direction claire et précise.	La méthode waterfall est simple, facile à mettre en place, logique et structurée. Elle s'adapte parfaitement à des projets qui répondent à des objectifs clairement identifiés ainsi qu'aux projets où la qualité prime sur le coût et les délais.	L'inconvénient majeur de cette approche est son manque de flexibilité à cause de son déroulement séquentiel. En effet, la méthode waterfall ne laisse aucune place aux changements et aux imprévus.
V Model	Le V-Model se concentre sur une méthode typiquement en cascade qui suit des phases strictes étape par étape. Alors que les étapes initiales sont des phases de conception générales, les étapes progressent de façon de plus en plus granulaire, menant à la mise en œuvre et au codage, et finalement, à travers toutes les étapes de test avant la fin du projet	Convient aux projets restreints : en raison de la nature rigoureuse du modèle V et de sa conception linéaire. Idéal pour la gestion du temps : dans le même ordre d'idées, V-Model est également adapté aux projets qui doivent respecter une échéance stricte et respecter les principales dates importantes pendant tout le processus	Manque d'adaptabilité : semblable aux problèmes auxquels est confronté le modèle de cascade traditionnel sur lequel repose le modèle V, l'aspect le plus problématique du modèle V est son incapacité à s'adapter aux changements nécessaires pendant le cycle de vie du développement

2TUP[31]	Le 2TUP propose un cycle de développement en Y, qui dissocie les aspects techniques des aspects fonctionnels. Il commence par une étude préliminaire qui consiste essentiellement à identifier les acteurs qui vont interagir avec le système à construire, les messages qu'échangent les acteurs et le système, à produire le cahier des charges et à modéliser le contexte (le système est une boîte noire, les acteurs l'entourent et sont reliés à lui, sur l'axe qui lie un acteur au système on met les messages que les deux s'échangent avec le sens).	2TUP est un modèle réaliste et naturel qui conserve le caractère « étapiste » de waterfall mais l'intègre dans une approche itérative. Dans 2TUP, le risque est un facteur qui est tenu en compte explicitement.	Il est difficile de faire comprendre au client le mode l'opération de ce modèle. L'évaluation des risques exige une expertise pointue et repose sur une capitalisation de l'expérience vécue.
Agile (SCRUM)	La méthodologie Agile s'oppose généralement à la méthodologie traditionnelle waterfall. Elle se veut plus souple et adaptée, et place les besoins du client au centre des priorités du projet. A l'origine, cette approche a été créée pour les projets de développement web et informatique. Aujourd'hui, la méthode Agile est de plus en plus répandue car elle est adaptable à de nombreux types de projets, tous secteurs confondus.	L'avantage majeur de l'approche Agile est sa flexibilité. Les changements du client et les imprévus sont pris en compte et l'équipe projet peut réagir rapidement. Autre atout : la collaboration et la communication fréquente avec le client, ainsi que sa forte implication dans le projet. Une relation de confiance se tisse entre le client et l'équipe projet.	Comme le dialogue est privilégié, la méthode Agile laisse peu de place à la documentation, ce qui peut poser problème en cas de changement d'équipe projet, par exemple. Le client doit être disponible et s'intéresser à son projet afin de s'assurer qu'il répondra parfaitement à ses besoins. Tous les clients n'ont pas le temps, ni l'envie de s'impliquer pleinement dans la réalisation d'un projet
Extreme Programming	Extreme Programming ou extreme agile est l'une des méthodologies Agile. Il partage tous les principes Agiles, y compris une forte implication des clients dans le processus de développement logiciel, une bonne communication au sein des équipes et des cycles de développement itératifs. XP[32] pousse à l'extrême des principes simples.	Le principal avantage de Extreme Programming est que cette méthodologie permet aux sociétés de développement de logiciels de réduire les coûts et le temps nécessaires à la réalisation du projet. Les économies de temps sont possibles car XP met l'accent sur la livraison en temps voulu des produits finis. La simplicité est un avantage supplémentaire des projets de programmation extrême.	Dans les projets XP, la documentation des défauts n'est pas toujours satisfaisante. L'absence de documentation sur les défauts peut conduire à l'apparition de bugs similaires à l'avenir. Un autre inconvénient de XP est que cette méthodologie ne mesure pas l'assurance qualité du code. Cela pourrait causer des défauts dans le code initial.

Tableau 2.1 : Comparaison des différentes méthodologies de travail

[31] Two Tracks Unified Process
[32] eXtreme Programming

Dans le cas de Lanterns, et d'après leur ClickUp, la méthodologie de travail qu'ils utilisent est Agile, cependant, elle n'est pas respectée à la lettre et au moindre détail. Mais embraser l'agilité est une excellente idée pour un studio de petite taille. Cela permet d'être flexible à tout moment avec l'équipe, les clients et les parties prenantes.

b. Phases de développement d'un jeu

Quand on parle de phases de développement, il ne s'agit pas de prendre une méthodologie de travail comme cité précédemment et de l'appliquer. Il faut aussi déterminer le cycle de vie du développement d'un jeu. Et dans le cas d'un jeu vidéo, c'est un cycle de vie bien définit qui obéit à quelques règles.

Nous avons déterminé qu'il se compose en 3 grand segments majeurs :

i. Préproduction

Quand nous parlons de préproduction, étape avant la production et le début réelle du travail, nous devons nous poser quelques questions essentielles :

De combien de personnes avons-nous besoin ?

Quelles sont les ressources que nous avons besoin ?

Combien le projet va-t-il coûter ?

…etc

Il faut aussi savoir que la préproduction doit être entre 10% et 20% de l'estimation en temps total de toute la phase de développement. Et généralement entre 1 semaine et 1 an, mais peut prendre beaucoup plus selon la taille du jeu et de l'équipe.

La phase de préproduction est elle-même divisée en 3 sous parties :

- **Le concept (1.1)**

Nous pouvons aussi appeler cette phase : l'émergence de l'idée, qui peut être :

+ Une histoire

+ Une forme unique de gameplay

+ Une nouvelle forme de technologie

- **Le plan (1.2)**

Nous pouvons aussi appeler cette phase : le raffinement de l'idée. C'est là que toutes les informations sont rassemblées et étoffées. Elle sont ensuite enregistrées dans un document de game design GDD[33] qui exprime pleinement la vision du jeu. Nous réalisons également une cartographie du plan de production.

Quelle que soit la qualité de la documentation et de la planification d'un jeu vidéo, de nombreux éléments changeront radicalement au cours de la phase de préproduction et de production en raison de : limitations techniques, mécanismes de jeu qui ne fonctionnent pas bien ensemble, départ des développeurs clés... etc.

Solution : Un document de conception Macro beaucoup plus petit et plus flexible (par Mark Sahmi, architecte principal / producteur de PS4[34] et PS Vita).

Le Macro Design Document (MDD) est un document très court qui contient des descriptions de haut niveau du jeu et peut être développé et mis à jour au cours du développement du jeu.

- **Prototypage (1.3)**

Un prototype peut être :

+ Une direction artistique

+ Des mécaniques de jeux fonctionnels

+ Les deux

Les assets artistiques du jeu et ne sont pas nécessaire vu que c'est une phase où il n'y a peu, voir pas, d'artiste au sein du studio. Donc le studio se rabat sur des placeholders. Les mécaniques de jeu peuvent être tester sur ces derniers ou alors sur des objets primitifs.

La clé de réussite est de prototyper de nombreuse fois et très souvent pour :

[33] Le Game Design Document est le cahier des charges d'une projet de type : jeu vidéo.
[34] PlayStation 4 par SONY.

+ Découvrez les limites de votre équipe et si vous souhaitez contourner ces limites ou prolonger votre temps de production pour les surmonter.

+ Trouver les aspects les plus agréables et les plus désagréables dans vos mécanismes de jeu de base.

ii. Production

La phase de production est la phase la plus longue, mais aussi la plus importante quand on veut créer un jeu vidéo car c'est la phase ou le jeu en lui-même y est développé à proprement parler. C'est une phase ou de nouvelles personnes sont engagées et de nouveaux rôles et positions aux seins des équipes crées.

C'est une phase où on se focalise sur le contenue, les assets et la création du code, mais aussi l'implémentation des contenus en plus de diverses tâches qui servent au développement et à l'amélioration continue du jeu.

Cette phase se divise généralement en 5 parties :

- **Vertical Slice (2.1)**

Une section du jeu (5 à 30 minutes) qui est responsable de ce à quoi ressemblera et donnera le jeu final, ainsi que le gameplay.

C'est une bonne pratique pour les projets de jeux à grande échelle et son objectif principal est :

+ A utiliser comme pitch.

+ Pour commencer à tester et commercialiser le projet en amont.

+ Mieux évaluer la durée du temps de développement des jeux (délai réalistes).

- **Pre-Alpha (2.2)**

Fait référence à toutes les activités effectuées pendant le développement du jeu avant la sortie officielle et la majorité du contenu est développé.

+ Les artistes créent les personnages/l'environnement.

+ Les animateurs donnent vie aux personnages et aux créatures.

+ Les concepteurs mettent en place les cartes et les niveaux.

+ Les programmeurs écrivent toutes les fonctions, événements et interactions.

Dans cette phase, nous devons d'abord travailler sur les éléments de jeu les plus importants car, les choses devront être coupées en raison de facteurs X.

Les Facteurs X	
Temps	**Pertinence**
+ Direct et quantifiable. + C'est souvent le temps contre l'impact, par exemple : est-ce que cette fonctionnalité en vaut la peine d'y consacrer notre temps ou plutôt, ne pas l'avoir, aura un impact négatif sur le jeu ou le projet ? Si la réponse est NON, alors : Est-ce que notre temps sera mieux consacré à quelque chose qui aura un impact sur le jeu ou le projet ?	+ Des trucs qui ne correspondaient tout simplement pas à l'état final du jeu. + Des assets ou des fonctions plus anciens ou actuels ne fonctionnent plus bien ensemble / ne sont tout simplement pas amusants. + Ne convient plus au jeu dans son état actuel. + Souvent, le résultat final est bien meilleur qu'initialement conceptualisé.

Tableau 2.2 : Les facteurs X

- **Alpha (2.3)**

Commence lorsque le jeu est terminé au niveau des fonctionnalités.

À ce stade, le jeu est entièrement jouable du début à la fin. Il est courant d'avoir peu de composants manquants et d'assets artistiques, mais les commandes et l'interface du joueur doivent être dans un état finalisé.

Cette phase consiste à terminer et à peaufiner le jeu plutôt que de créer du contenu supplémentaire. Et il s'agit de tester et d'atteindre la date de sortie.

S'il y a une fonctionnalité ou une fonction qui doit être supprimée ou minimisée dans le but d'atteindre la date de sortie, ce sera à ce stade.

Enfin, le QA est chargé de s'assurer que chaque mode de jeu, les fonctions, mécaniques de jeu, les performances fonctionnent correctement et enregistrent les incohérences, dysfonctionnements, problèmes de performances ou erreurs dans une feuille de bug ou une base de données de bug[35].

- **Beta (2.4)**

+ Le jeu est considéré comme *content complete*[36].

+ Concentration sur la correction des bugs.

+ Tous les assets sont intégrés au jeu.

+ Tout le processus de production s'arrête

Les principaux objectifs de cette étape sont de stabiliser et d'optimiser le jeu et d'éliminer autant de bugs que possible. Le QA doivent hiérarchiser les bugs de Elevé à Faible en fonction de la feuille de bug.

Élevé : bugs majeurs qui nuisent aux performances du jeu / atteinte à la qualité.

Faible : bugs mineurs extrêmement difficiles à reproduire.

En d'autres termes, si nous devons résumer cette phase en un mot, et le mot d'ordre sera : stabilité.

- **Gold (2.5)**

Le jeu final est envoyé pour être testé par l'éditeur. S'il est validé, il est rendu public (à la vente). Après cela, nous passons instantanément à la postproduction et commençons à travailler sur des patchs et correctifs.

Cependant, un jeu sur le marché n'est pas parfait, une écoute attentive des critiques de la part de la presse et des joueurs est nécessaire.

[35] Les termes techniques en anglais sont : Bug sheet / Bug database.
[36] Tout le contenu est créé

iii. Postproduction

Cette phase clôture définitivement le projet. Même si le jeu est déjà dans les mains du joueur, ce n'est pas fini pour autant, il reste 3 choses très importantes à faire pour pérenniser le jeu et atteindre une bonne maturité.

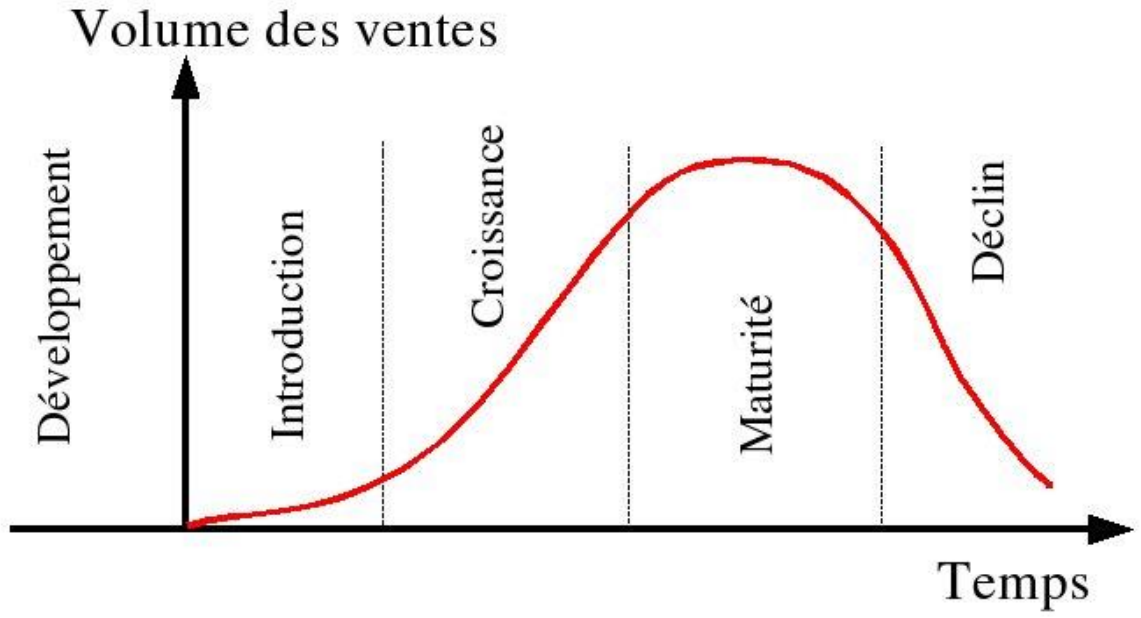


Figure 2.8 : Cycle de vie d'un produit

- **Patches (3.1)**

Les patches / mise à jour ont pour objectifs d'améliorer l'expérience du joueur et de la rendre plus agréable. Elle corrige aussi certains bugs qui n'ont été découvert qu'au moment du lancement.

- **Post-Mortem (3.2)**

Le post-mortem est essentiellement une réunion avec tous les équipes et pas seulement les responsables en chef. Elle donne à tout le monde l'occasion de s'exprimer sur les hauts et les bas du développement et ainsi, avoir une idée globale du déroulement des deux premières phases pour ensuite les améliorer par la suite. Le post-mortem permet l'échange des idées qui touchent de près ou de loin au management des équipes de jeu vidéo. Le but c'est de savoir ce qui a marché et ce qui n'a pas marché.

Il ne faut surtout pas pointer du doigt et blâmer les individues. Il faut se concentrer sur :

+ Les process de production

+ Les divers process

+ Le scheduling

+ Le management du temps

+ L'implémentation des fonctionnalités

+ …etc.

L'objectif est aussi de répondre ces questions :

+ Avons-nous atteint l'objectif initial du jeu que nous nous sommes fixés pour produire ?

+ Qu'est-ce qui s'est bien passé ? Qu'est ce qui ne s'est pas bien passé ?

+ La portée du projet, les délais, les fonctionnalités sont-elles réalistes ?

+ Leçons que nous avons apprises ?

- **Closing Kit (3.3)**

Il est important d'organiser tous les assets, les documents et le code source du jeu dans un closing kit[37].

L'objectif est d'avoir des assets facilement disponibles pour une utilisation future ou pour les utiliser comme références.

Une utilisation future peut être un nouveau jeu, une suite ou un DLC[38]

Les choses à sauvegarder sont généralement :

+ Les GDD et MDD

+ Le code

[37] Kit de clôture en français

[38] C'est un contenu téléchargeable (DownLoadable Content) est un contenu supplémentaire créé pour un jeu vidéo déjà sorti, distribué via Internet par l'éditeur du jeu. Il peut être ajouté sans frais supplémentaires ou il peut s'agir d'une autre forme de monétisation de jeux vidéo, permettant à l'éditeur de tirer des revenus supplémentaires d'un titre après son achat, souvent en utilisant un certain type de système de microtransaction.

+ Les assets artistiques

+ Les assets finaux

+ Les musiques et sons

+ …etc.

c. Communication, partages des ressources et gestion des versions

Au cours de mon stage, nous avons aussi travailler sur le versionning des fichiers. En effet le versionning chez Lanterns n'est pas synchronisé entre le département technique et le département artistique.

Pour le département technique, on sauvegarde chaque fichier grâce à *perforce* et le processus est le suivant :

+ Un programmeur code une fonctionnalité

+ Le programmeur sauvegarde le code sur perforce

+ Si un bug est détecté sur ce morceau de code, n'importe quel autre programmeur peu modifier le code.

En d'autres termes, toute la partie technique est sauvegardée en ligne sur perforce et que les programmeurs, eux, n'ont chacun, qu'une copie de cette version sur leurs machines.

Les copies peuvent être tous différentes en un instant T, donc les changements sur perforce se font séquentiellement.

Un changement sur perforce = Une version de la partie technique du jeu ; avec chaque changement accompagné d'un numéro de série, le nom du développeur qui a effectué le changement et la date du changement.

De cette manière, n'importe quel développeur peut avoir accès a une version antérieure du code à n'importe quel moment.

Le problème est que la partie artistique est beaucoup trop complexe est volumineuse à gérer grâce à perforce, est qu'une autre solution doit être adoptée.

Il faut savoir que la gestion des versions est une partie clé de la communication entre les différents départements. Une gestion asynchrone / différente peut poser des problèmes de communication et de temps. Donc avoir un seul projet au lieu de deux projets séparés (partie technique et artistique) peut grandement faire avancer le développement et faciliter la tâche aux artistes techniques et différents intégrateurs par la suite.

Figure 2.9 : Logo Perforce

Perforce, légalement Perforce Software, Inc., est un développeur américain de logiciels utilisés pour développer et exécuter des applications, y compris un logiciel de contrôle de version, une gestion de référentiel Web, une collaboration avec les développeurs, une gestion du cycle de vie des applications, des serveurs d'applications Web, des outils de débogage et un logiciel de planification Agile.

Perforce développe des logiciels utilisés par les développeurs de logiciels pour gérer le code pendant le processus de développement. La gamme de produits comprend les éléments suivants :

Helix Core, anciennement Perforce Helix, est le logiciel de contrôle de version de la société pour les environnements de développement à grande échelle. Le système de contrôle de version Helix gère une base de données centrale et un référentiel maître des versions de fichiers.

Les clients Helix Core se répartissent en cinq catégories environ : Git, commande, interface graphique, Web et plug-in. Le système perforce peut rendre tout ou partie de son contenu disponible en tant que référentiels Git. Les utilisateurs de Git et d'autres clients peuvent travailler avec le même contenu de fichier et le même historique. Les commits[39] Git sont visibles par les utilisateurs d'autres clients en tant que listes de modifications

[39] Enregistrement.

perforce, et vice versa. Les utilisateurs soumettent les fichiers modifiés ensemble dans des listes de modifications, qui sont appliquées en tant que commits atomiques.

Le logiciel serveur et client est publié sous forme d'exécutables prédéfinis pour Microsoft Windows, macOS, Linux, Solaris, FreeBSD et d'autres systèmes d'exploitation. *[Voir ref 2.1, Annexe 2, Page 64 : Perforce]*

Bonne nouvelle pour nous, nul besoin d'aller chercher plus loin que perforce. Il y a eu cependant de la recherche et le candidat premier été GitLab.

GitLab est un logiciel libre basé sur git proposant les fonctionnalités de wiki, un système de suivi des bugs, l'intégration continue et la livraison continue. Développé par GitLab Inc et créé par Dmitriy Zaporozhets et par Valery Sizov, le logiciel est utilisé par plusieurs grandes entreprises informatiques incluant IBM, Sony, le centre de recherche de Jülich, la NASA, Alibaba, Oracle, Invincea, O'Reilly Media, Leibniz Rechenzentrum, le CERN, European XFEL, la GNOME Foundation, Boeing, Autodata et SpaceX. *[Voir ref 2.2, Annexe 2, Page 64 : Gitlab]*

Figure 2.10 : Logo Gitlab

L'une des avantages de Gitlab c'est qu'il offre une prévisualisation des assets en 3D, cependant, le format voulu qui est utilisé par Lanterns (Le format monficher3d**.fbx**) n'est pas compatible avec ce dernier ce qui a fortement joué en la défaveur de Gitlab.

Perforce propose pour Helix Core un plugin[40] qui est spécialement conçue pour les artistes et que nous pouvons facilement intégrer. Donc nul besoin de migrer vers un autre logiciel. Le plugin comporte toutes les fonctions que nous désirons, à savoir le versionning des fichiers 3D au format .fbx et une prévisualisation avant téléchargement du dit fichier.

III. Division du travail et méthodologie de suivi

A. Techniques de suivi et l'organisation personnelle

Tout comme Lanterns le fait avec son projet grâce à ClickUp, nous aussi ayant pris part d'avoir une organisation des tâches à faire lors du stage.

Nous avons décidé de nous orienter sur un autre logiciel que ClickUp, et ce logiciel s'appelle Hack'n'Plan. Cette décision de nous mettre en retrait est due au fait que notre travail ne va jamais interférer avec l'organisation propre que Lanterns fait au moyen de ClickUp. En effet, nos tâches sont parfaitement indépendantes des tâches de Lanterns. De plus, ayant déjà de l'expérience sur Hack'n'Plan, nous évite de prendre du temps a se former au tout début et nous pouvons avoir une bonne organisation dès le début du stage.

Figure 2.11 : Logo Hack'n'Plan

Hack'n'Plan est un outil de gestion de projet pour le développement de jeux, qui rassemble la conception de jeux et la gestion de projet pour fournir un meilleur flux de travail et une organisation plus intuitive pour les équipes de développement de jeux.

Bien qu'il ne suive pas de modèle spécifique, l'outil s'inspire fortement des méthodologies agiles. C'est la raison pour laquelle l'application s'articule autour du tableau kanban[41]

[40] Extension du logiciel

[41] Les tableaux Kanban décrivent visuellement le travail à différentes étapes d'un processus en utilisant des cartes pour représenter les éléments de travail et des colonnes pour représenter chaque étape du processus. Les cartes sont déplacées de gauche à droite pour montrer les progrès et pour aider à coordonner les équipes

comme outil principal pour suivre l'avancement de vos tâches. Définir des tâches/user stories significatives, petites et bien définies à partir de nos exigences fonctionnelles, puis les mettre sur un tableau kanban, nous a grandement aidé à vérifier facilement l'état de nos tâches et à garder tout sous contrôle. *[Voir ref 2.3, Annexe 2, Page 64 : Hack'n'Plan]*

B. Résumé des sprints

Et bien comme cité précédemment, nous avons aussi opté pour une organisation des tâches grâce a la méthodologie Agile et donc par conséquent, nous devons respecter des « sprints ».

Un sprint est une courte période de temps pendant laquelle une équipe Scrum travaille pour accomplir une quantité de travail définie. Les sprints sont au cœur même des méthodologies Agile, et aideront les équipes agiles à livrer de meilleure performance avec moins de maux de tête. Dans cette période de temps, nous organisons les tâches à faire. *[Voir ref 2.4, Annexe 2, Page 64 : Définition d'un sprint]*

Nous avons divisé notre stage sur 2 parties,

Une partie consacrée à la compréhension et à la recherche d'une problématique qui a duré 1 mois, qui elle-même est subdivisé en 4 sprints.

Et une partie de deux mois, consacré à la R&D du nouveau pipeline. Même si un sprint de deux mois n'est plus conforme à la méthodologie, nous avons quand même pris le choix de ne pas subdiviser cette dernière puisqu'au final, ce n'est qu'une seule et grande tâche.

Conclusion

Dans ce chapitre, nous avons résumé notre travail effectué au sein de Lanterns en expliquant les techniques utilisées par Lanterns et nous-même pour avoir un bon résultat et travailler dans de bonnes conditions.

effectuant le travail. Un tableau Kanban peut être divisé en « couloirs de nage » horizontaux représentant différents types de travail ou différentes équipes effectuant le travail.

Chapitre **Troisième**

Approche empirique et Résultats

Introduction au chapitre

Dans ce troisième et dernier chapitre nous allons exposer nos solutions trouvées - souhaitées ou non- et faire une synthèse de ces dernières.

Nous allons aussi approfondir plus en détails, grâce aux résultats, le travail effectué au sein de Lanterns.

Enfin, nous allons interpréter nos résultats et donner des recommandations ainsi qu'un avis général plus étoffé.

I. Contribution et travail effectué

1. Gestion des versions commun

Comme nous l'avons vu dans le précédent chapitre, avoir un seul et unique logiciel centralisé pour le versionning des fichiers est important pour Lanterns, cela impactera à long terme le studio dans leur organisation de travail. En effet, nous somme convaincu que de cette manière, nous éviterons plusieurs problèmes qui peuvent survenir dans les phases de développement avancés, car il faut savoir une chose, c'est que Lanterns viens tout juste de se lancer dans la phase de production.

Pour donner une image qui illustre nos propos, imaginez que dans les phases les plus avancées, des départs ont eu lieu au sein de Lanterns ; et bien dans ce cas de figure qui est fréquent et fort probable à n'importe quelle entreprise, les nouveaux arrivants qui prendront les places des anciens n'auront plus de mal à remonter le fil des versions pour savoir ce qui a été fait et, surtout, avec quelles techniques.

Mais aussi au management d'avoir accès à des versions ultérieures comme bon leur semble. Imaginez qu'un artiste 3D doit modéliser une pomme rouge en 3D qui servira pour le jeu, et à cause d'une modification dans le scénario par les game designers quelque mois après, la pomme devra changer de couleur pour être au final : une pomme verte ; et bien grâce au versionning nous pouvons ainsi retrouver la trace de cette pomme rouge et la transformer directement en pomme verte sans la récréer depuis le début.

C'est évidemment un exemple tout bête, car une pomme est facile et rapide à créer, mais imaginer des objets bien plus complexe qu'une simple pomme et qui nécessite de

retravailler la partie 3D et la partie technique en même temps. Là, le versionning commun devient indispensable et évitera une perte de temps considérable à coup sûr.

2. Simulation du pipeline

Revenons un peu sur le sujet principal.

La simulation de l'ancien pipeline a déterminé que le processus est fluide au début mais qu'elle devient bloquante dès qu'on approche de la phase ou il faut intégrer la partie 3D avec la partie technique

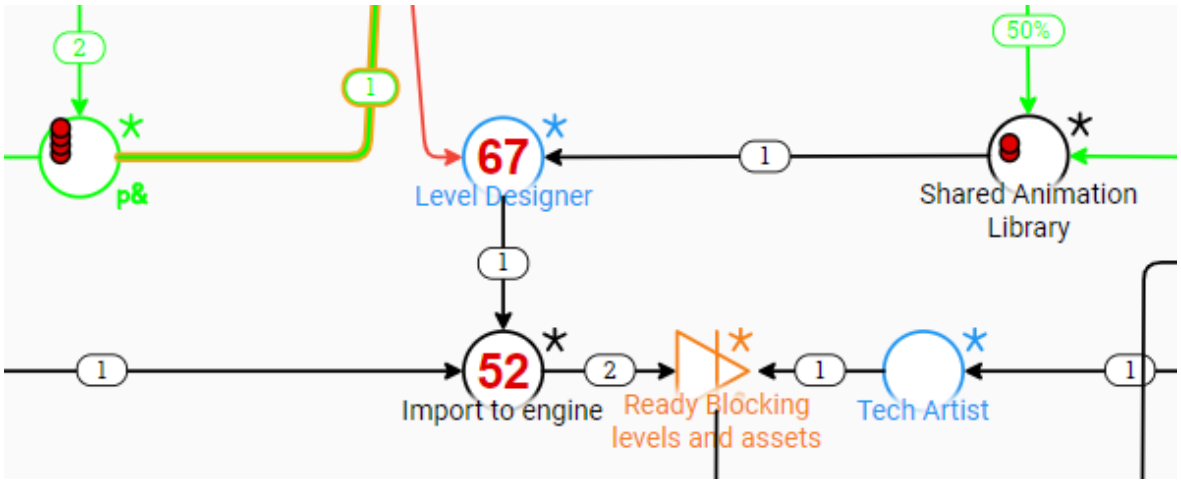


Figure 3.1 : Intégration bloquante (pipeline)

En effet, sur 100 itérations du pipeline, on arrive à ce résultat. Les chiffes que nous voyons ici dans cette figure ci-dessus, dans le nombre de tâche en attente prêt à être livrées pour la suite, les tâches en attentes sont beaucoup trop nombreuses. On remarque que sous cette forme, le level design (partie très importante du processus de création d'un jeu vidéo) et l'importation dans Unreal Engine devient surchargé par les tâches qui arrivent en amont.

Pour mieux expliquer cette figure, Le nœud « Shared Animation Library » a 2 tokens en attente prêts à être livrés, alors que « Import to engine » en contient 52 après 100 itérations. Et ces nombres phénoménaux continuerons à accroitre de façon linéaire au fil du temps jusqu'à ce que la production cesse.

Grace a l'outils « Execution Chart » de Machinations, nous pouvons comparer en détail plusieurs nœuds et voir leurs efficacités.

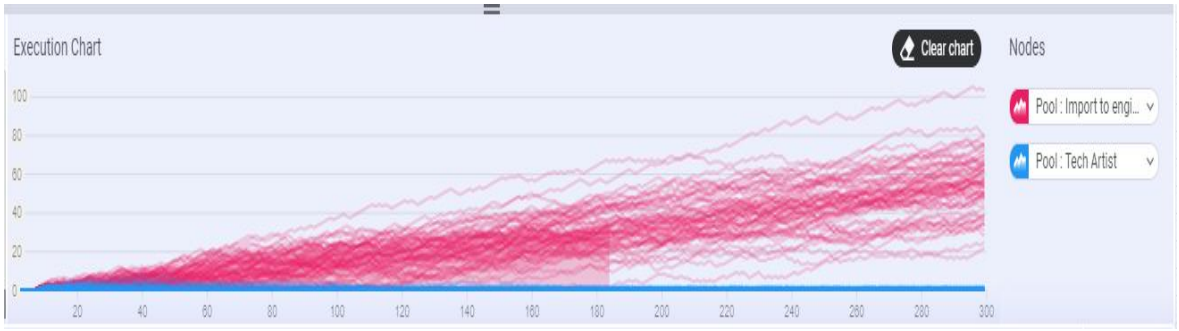


Figure 3.2 : Intégration bloquante (charte)

Voyez bien, en bleu nous avons pris comme exemple le nœud « Tech Artist » et en rose, le nœud « Import to engine ».

Cet outil charte permet d'exécuter plusieurs simulations à la fois. Ici, nous avons exécuté 100 simulations sur 300 itérations du travail et nous pouvons constater que Tech Artist est constante et reste toujours à 0, 1 ou 2 ; alors qu'Import to engine fluctue beaucoup plus. Mais la courbe est toujours en hausse quel que soit la simulation, allons de 20 tokens à la sortie à plus de 100 tokens à la sortie.

Dans l'axe X nous avons le nombre d'itération de la simulation et dans l'axe Y, le nombre de tokens présent dans chaque nœud.

[Voir Annexe 6]

3. Généralisation du process

Pour nous, il faut complètement remodeler le pipeline pour qu'il soit mieux compréhensible pour n'importe quel nouvel arrivant et surtout, il doit obéir aux phases de développement d'un jeu vidéo.

Premièrement, en tant que nouveau membre de Lanterns, je dois bien savoir ou je me situe dans la chaine de production. Savoir si je suis en tout début ou en toute fin ou partout à la foi me permettra de juger mes attentes par rapport aux autres et savoir quelles tâches sont à prioriser dans un moment T.

Deuxièmement, en tant que top management de Lanterns, je peux visualiser plus facilement où se situe mon jeu par rapport aux phases de développement d'un jeu et ainsi adapter mes décisions et la communication vis-à-vis aux joueurs et aux équipes.

Et enfin en tant qu'investisseur, je peux aussi savoir en toute transparence où se situe le jeu par rapport aux délais attendues ; car en effet, un pipeline bien structuré avec des étiquettes chaires permet d'attribuer à chaque phase et sous phase (et même à chaque tâche), une estimation du temps que prendra le jeu à être développé.

Tout d'abord, grâces aux recherches effectuées et aux différentes interviews, nous avons pris la peine de généraliser le process, pour ensuite, l'adapter aux besoins de Lanterns.

Généraliser le process nous permet de mieux assimiler chaque partie de la production très vite et très simplement ; non seulement pour Lanterns mais aussi pour nous qui ayons pris la décision de partir de zéro.

Il est facile de tomber dans le piège qu'un pipeline se doit d'être sous la forme d'une charte regroupant tout le processus de développement, mais c'est faux. Une forme classique et linaire ne correspond nullement aux principes de l'agilité. De plus, il faut savoir que le jeu de Lanterns est bien unique dans son approche et est décomposé en « Chapitre ».

Tout comme un livre ou même ce rapport, Il est évidant qu'il faut d'abord travailler le plan de rédaction (comme par exemple une table des matières en version primitive) et ensuite de travailler chapitre par chapitre pour avoir une ligne directrice claire. Et c'est cette logique qu'à pris Lanterns ; en ce moment même, ils développent chapitre par chapitre. Dès qu'un chapitre est fini, on passe au suivant.

Bien sûr, rien n'interdit de passer à un autre chapitre quand les ressources nécessaires pour le chapitre en cours de développement sont épuisées, mais l'idée reste la même.

Travailler selon ce principe est une bonne idée, il n'est pas fréquent d'imposer la trame scénaristique d'un jeu comme processus de développement et de suivi, car très peu de jeu en font leur force.

Donc pour nous, le pipeline doit aussi suivre ce modèle, mais doit être optionnel. En effet, le pipeline se doit d'être générique à tous les prochains jeux de Lanterns et ne doit pas imposer cette vision très particulière du développement.

En résumé, notre pipeline se doit d'être à la fois : Générale et qui suit les standards du développement des jeux vidéo dans le monde ET spécifique dans le cas du projet courant de Lanterns.

C’est deux visions qui semblent contradictoires éliminent toute tentative de création d’un pipeline sous la forme de l’ancienne version.

La solution est donc de la découpé en plusieurs parties.

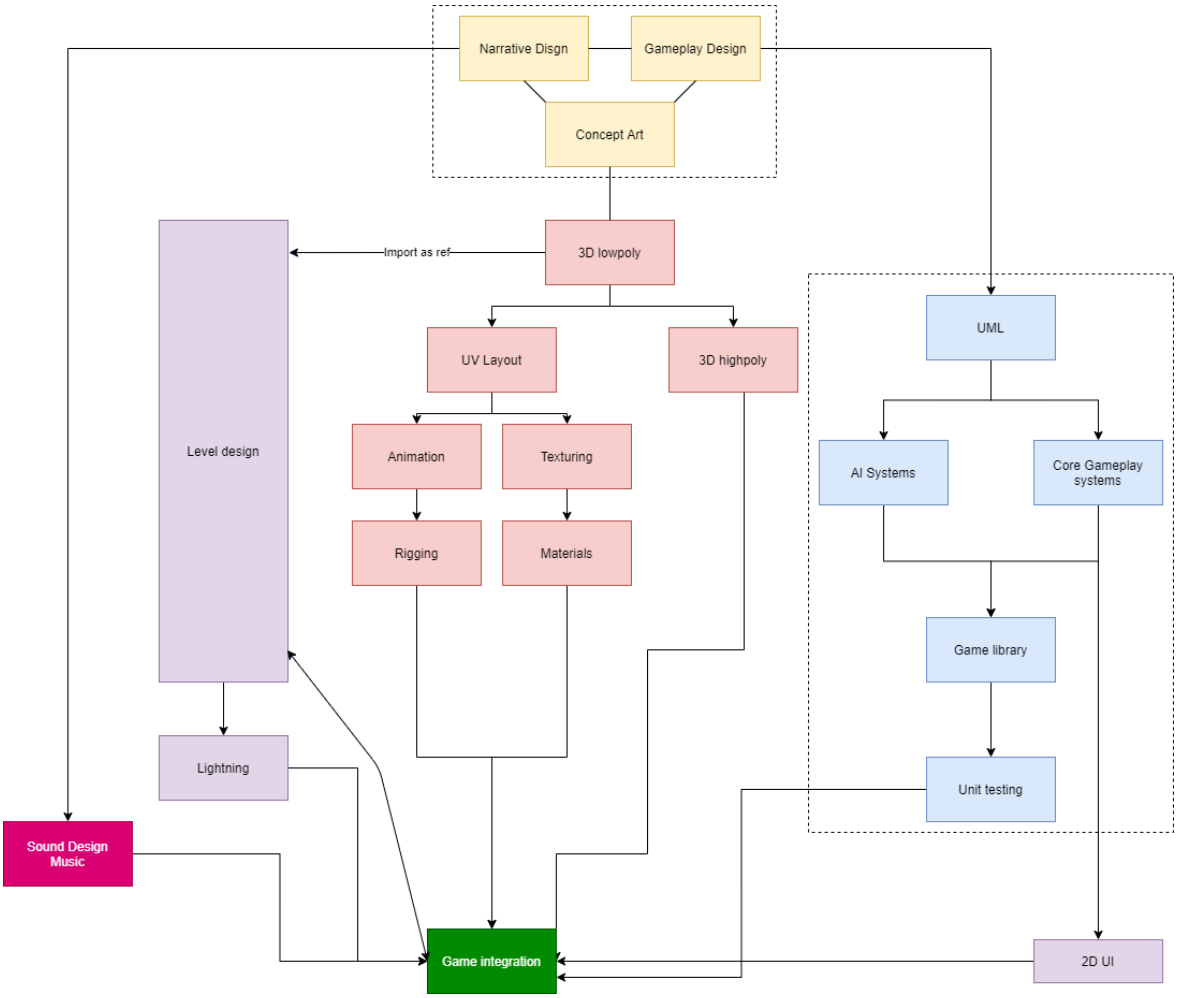


Figure 3.3 : Top view du pipeline

Cette figure qui illustre le process général est appelé le « top view » ou « vue générale » et est la première étape qui a était réaliser. *[Voir Annexe 7]*

Cette vue est décomposée comme suit :

- Les rectangles rouges représentent la partie Animation et 3D.
- Les rectangles bleus représentent la partie programmation.
- A gauche et en violet, c’est la partie Level Design.
- En jaune, c’est la partie Dame Design.

Et toutes les parties rejoignent l'intégration.

Nous avons gardé la même structure pour la partie programmation, alors que les autres parties sont totalement remodelées et encapsulées dans des entités majeures. Et donc pour expliquer son fonctionnement, le concept art sert de base pour la modélisation qui se fait en 2 stages :

Le premier étant, la modélisation en low poly[42] qui est un modèle 3D avec le moins de détails que possible.

Et le second c'est le high poly, qui, inversement au low poly, contient le plus de détails que possible et, qui, à titre d'exemple, pour un personnage en 3D, ce personnage doit s'approché du réalisme. En ce qui concerne le 2D UI, cela représente l'interface du joueur.

Nous avons aussi pris en considération les dires du COO et nous avons opté pour une mettre le Level Design comme une étape parallèle aux autres. Il ne suffit bien entendue pas d'entendre seulement qu'une seule partie, il faut aussi confirmer ses dires.

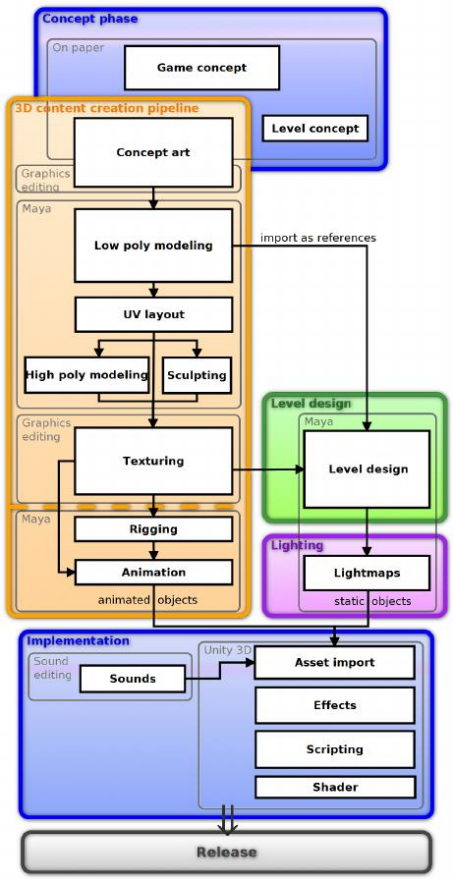


Figure 3.4 : Level Design en parallèle du process

[42] Poly pour polygone

Nous constatons ici que c'est assez commun que la partie du Level Design se place en parallèle du processus de développement car c'est une partie qui est à la fois au début, au milieu et à la fin du processus. Ça se voit en théorie grâce à la figure précédente de researchgate.net mais aussi dans la pratique chez Lanterns.

Ce processus peut être itératif comme définitif et peut suivre une méthodologie agile comme en cascade (waterfall).

Cependant même si les nœuds présents dans ce nouveau pipeline représentent les différents rôles majeurs dans le processus de développement d'un jeu, ce n'est qu'une vue basique et n'apporte pas les détails que présente l'ancien pipeline mais cette dernière doit respecter les 3 phases du développement d'un jeu vidéo.

En effet, serte nous comprenons bien ce que « Animation » veut dire, mais que fait l'animateur réellement dans chaque phase de développement ?

4. Décomposition du process générale

a. Phase de Préproduction

Level Design	3D	Rigging	AI System	Gameplay systems	Animations	2D UI	Design
Exterior Level art	Character Head	Some characters	Basic Allies NPCs	Superpowers	Cutscenes Animations	Main game UI	Game design document
Interior Level art	Character Clothing		Basic Enemies NPCs	Commentary systems	Gameplay Animation		Macro design documents
Level blocking	Props and assets			Dialogues and Subtitles	Interactions		Narrative design
Mood board	Miscs			Quick events			Gameplay design
Ref board				Inspect systems			Concept Arts
Landscape							

Tableau 3.1 : Tâches dans la phase de préproduction

Ici dans ce premier tableau nous avons décomposé les nœuds qui ne sont pas atomiques en plusieurs tâches qui peuvent être faite dans cette phase.

Bien sûr, toutes les tâches citées dans ce tableau sont adaptées pour le jeu en production actuel de Lanterns mais puisque c'est un tableau, nous pouvons à tout moment ajouter des lignes si le besoin s'en fait sentir.

b. Phase de Production

Level Design	3D	Rigging	AI System	Gameplay systems	Animations	2D UI	Sound Design
Exterior Level art	Character Head	Main Characters	Allies NPCs	Superpowers	Cutscenes Animations	Main game UI	Cinematic sound design per chapter
Interior Level art	Character Clothing	Other characters	Enemies NPCs	Commentary systems	Gameplay Animation	Main game UI	Ambient/Env
Level blocking	Props and assets			Dialogues and Subtitles and Sound Integration	Interactions	Mini games UI	NPC sound
Mood board				Quick events	MoCap		Interactions
Ref board				Inspect systems			Dialogues
Landscape				Saving systems			In game voices
							Cutscenes voices
Dès les phases Alpha/Beta							
Miscs	Miscs		Bug fixing	Mini games	Bug fixing	Icons	
				Bug fixing			

Tableau 3.2 : Tâches dans la phase de production

Comme pour la précédente phase, ce tableau illustre aussi les tâches à accomplir pendant la phase de production.

Contrairement à la phase de préproduction ou les éléments dans le tableau sont des éléments vitaux pour créer un prototype, ici on trouve les mêmes mais accompagnés d'autres tâches nécessaires à la production d'un jeu.

Nous pouvons aussi remarquer que le design à totalement disparue du tableau car c'est une phase propre à la préproduction et nous l'avons remplacé par le Sound Design. Cependant,

nul n'empêche la modification du MDD par les designers à n'importe quel moment, mais c'est plus de l'ordre du management que du design à ce point.

Nous pouvons aussi constater que nous y avons inclue la fixation de bugs et quelques tâches mineurs dans le tableau. En effet, dès la phase alpha et beta, la production ralentie de cadence pour ne se consacrer que sur l'amélioration des performances du jeu.

c. Phase de postproduction

La production cesse complétement dans cette phase car il n'y a plus de jeu à développer à proprement parler, la seule chose qui reste à faire, c'est de créer des patchs et faire notre analyse grâce au post-mortem et sauvegarder des données clés du projet dans un kit de clôture.

Figure 3.5 : Pipeline, partie postproduction

Souvent dans cette phase, on ne mobilise pas beaucoup de manpower, mais seulement une petite équipe ou un petit comité pouvant déployer les patchs à temps et en heure.

Il faut savoir aussi que les DLC, sont des projets à eux seuls. Même si un DLC n'est que du contenu additionnel du jeu précédemment vendu, ils ne demeurent pas moins qu'il nécessite beaucoup d'effort à conceptualiser et à développer les nouvelles mécaniques de jeux et personnages, et il faut donc repartir de zéro, et au donc, au design.

5. Adaptation en cycle chapitrale

Pour être au plus proche de ce qu'il se fait chez Lanterns, il faut encore repenser notre pipeline car elle ne répond pas complétement aux besoins du projet. Et ceux pour une raison majeure :

Le nouveau pipeline n'est toujours pas adapté à l'approche atypique que Lanterns fait en termes de management et du développement. En effet, nous avons vu qu'ils développent chapitre par chapitre et pas forcément dans un ordre croissant. A titre d'exemple, le premier chapitre qu'il sont actuellement en train de développer, n'est que le numéro 21 sur

une liste d'une trentaine de chapitres, et donc, il faut apporter à notre pipeline, une nouvelle vue qui soit itérative a tout chapitre.

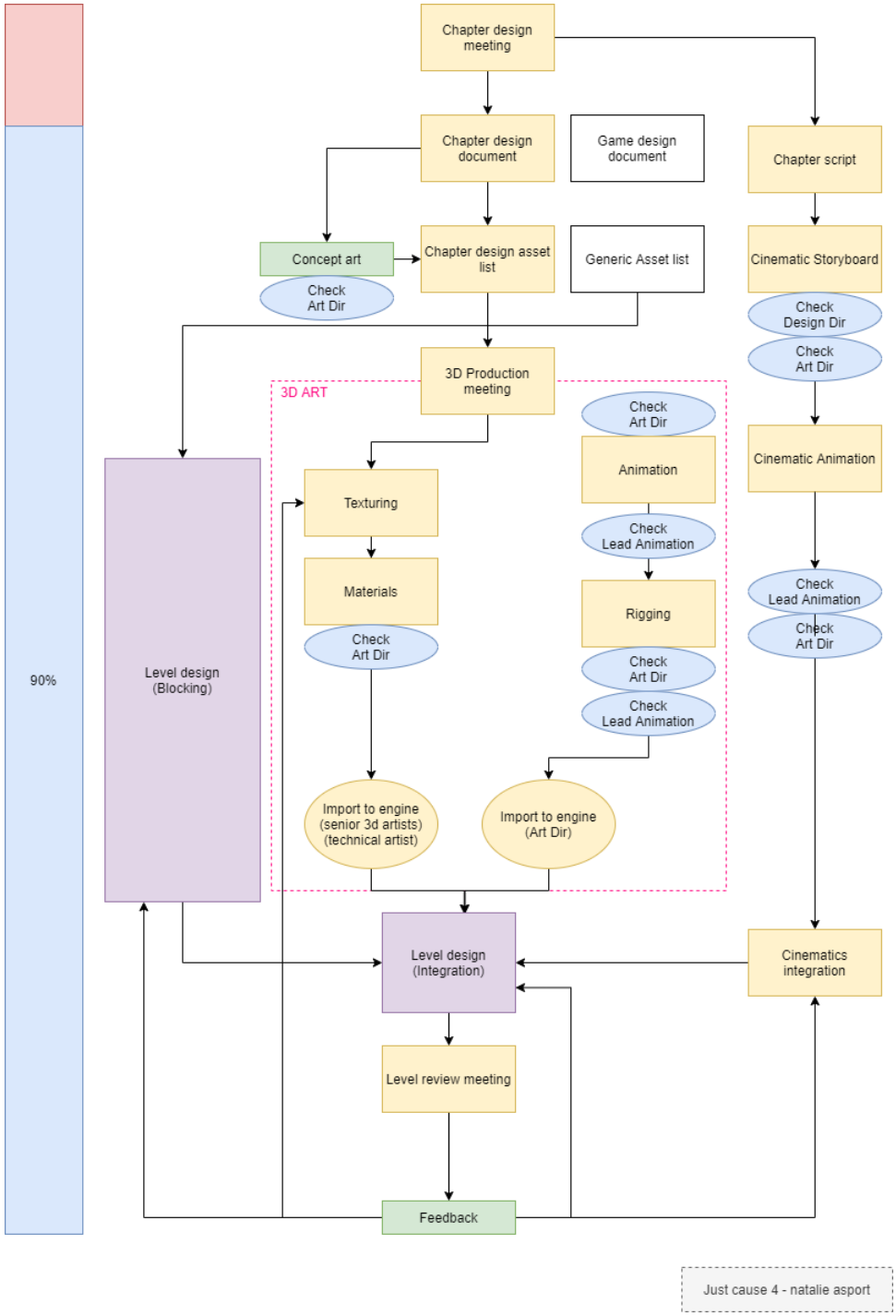


Figure 3.6 : Adaptation en cycle chapitrale (suivant un rythme de livrable par chapitre)

Cette seconde vue du pipeline permet entre autres de visualiser la progression du développement du chapitre grâce à la barre tout à gauche de la figure 3.6. *[Voir Annexe 8]*

Ce système inspiré des travaux de Natalie Asport sur le jeu « Just Cause 4 » permet d'avoir une vue très spécifique centrée sur le chapitre en cours de développement. Un chapitre commence toujours par une réunion des designers pour la conception du chapitre et finit par l'intégration en Level Design.

Nous avons aussi ajouté en ellipse bleu, les responsables qui devrons superviser les tâches en cours.

Cette vue représente cependant que la partie artistique du chapitre car la partie technique englobe toujours le jeu entier. En effet, si nous créons un système, il sera présent dans tous les chapitres.

Ce n'est qu'une version plus détaillée de la figure 3.3 mais seulement centré sur la partie rouge de cette dernière.

Ce nouveau pipeline permettra, sous réserve des résultats de :

- Flexibilité pour le top management en tant que techniciens.
- Partage de fichiers plus structurel => pas de temps perdu.
- Vue d'ensemble sur l'état du chapitre, nous pouvons faire une estimation du temps selon les chapitres précédents.
- Moins de « si nous réessayons ».

II. Analyse, interprétation et difficultés

Il est bien entendu difficile d'estimer un travail théorique sans pouvoir l'appliquer au sein même de l'entreprise mais il est clair qu'un travail effectué en 3 mois ET à la vue de l'entreprise en pleine activité, a de meilleure chance d'aboutir qu'un travail fait à l'aveugle en pleine création de l'entreprise.

Mais avoir ce type de charte sur soi quand on est manager est toujours utile pour pleins de raisons que nous pouvons oublier de citer ici dans ce rapport.

Il est bien clair qu'un producteur / chargé de production pourrait grandement améliorer le quotidien de l'entreprise. Ce chargé de production pour avoir la casquette d'un Scrum Master par exemple qui encadrera les équipes de Lanterns et faire un suivi propre. Donc nous avons aussi pris la peine de rédiger la fiche de poste d'un Scrum Master.

Scrum master fiche de poste

Description du poste

Nous sommes à la recherche d'un Scrum Master pour coordonner et accompagner notre équipe de développement de jeu.

Qu'est-ce qu'un Scrum Master ?

Pour faire simple, vous serez notre personne-ressource pour appliquer la méthode Scrum et produire un travail de haute qualité. Le Scrum Master est notamment chargé de gérer les délais, de résoudre les problèmes et de former l'équipe aux méthodes Agile.

En fin de compte, vous aiderez les équipes à s'organiser pour être flexibles et pleinement productives pendant les sprints.

Expérience et compétences du Scrum Master

Vous devez avoir une excellente connaissance du cadre Scrum ainsi que de tous ses artefacts et techniques. Vous aurez également besoin de savoir coordonner les projets et les ressources humaines (parfois de savoir faciliter les changements) en vous focalisant sur les résultats. Si vous êtes un excellent communicant, un leader compétent et investi dans les cadres Agile, nous aimerions vous rencontrer.

Responsabilités

- Gérer la portée et le calendrier de chaque projet
- Coordonner les sprints ainsi que les réunions quotidiennes et rétrospectives
- Former les membres de l'équipe aux cadres Agile
- Faciliter la communication en interne et l'efficacité de la collaboration
- Servir d'interlocuteur pour les communications externes (par exemple auprès des clients ou des parties prenantes)
- Travailler avec les Product Owners pour traiter les backlogs (listes ordonnancées) et les nouvelles demandes

Exigences

- Excellente connaissance des techniques et des artefacts Scrum (notamment la définition de « fini », les témoignages d'utilisateurs, les tests automatisés, le raffinement du backlog)
- Bonne connaissance des autres cadres Agile (Crystal, XP, etc.)
- Excellentes aptitudes en communication et sens du service et du leadership
- Capacité à résoudre les problèmes et les conflits
- Excellentes compétences organisationnelles
- Diplômé(e) en informatique, en affaires ou dans un domaine similaire
- La certification Scrum Master est un plus

_____

Un profil de ce type permet la continuité de ce stage en appliquant à la lettre le nouveau pipeline.

La difficulté majeure était de compiler toutes les données parfois très vagues sur le sujet du management des entreprise spécialisés des jeux vidéo et trouver les cas qui se rapprochent au plus de la vision de Lanterns.

Parfois il s'agit de travail d'interprétation des sources par rapport à d'autres qui a conduit à la construction du pipeline. Qu'ils s'agissent de sources traitants le management en générale ou de forums de discussion très flou qui parle de indie games, les compilés et extraire le meilleur de ces derniers, tout en gardant un œil dur notre objectif a été la partie la plus difficile et celle qui nous a couté le plus de temps pour un résultat purement théorique.

Cependant nous pensons que notre travail est sans conteste une brique parmi plusieurs qui seront rajoutés au fil du temps.

Comme cité précédemment, pour que ce travail soit concluant, il faut le tester sur le long terme et traverser toutes les phases de développement, et au moins finir un chapitre et le

comparer au suivant pour, avoir la première confirmation que le système fonctionne ou non.

Mettre des chiffres sur les nœuds qui composent le pipeline va permettre de jauger chaque nœud avec précision et pour ensuite estimer la durée totale d'un chapitre et au final prévoir une date de sortie réaliste.

Conclusion

Dans ce chapitre nous avons vu le résultat du travail effectué au seins de Lanterns.

Nous avons analysé le pipeline par rapport à l'ancien tout en donnant notre avis sur le travail effectué et les pistes d'amélioration après ce stage.

Conclusion Générale

Au terme de ce mémoire de fin d'études, je tiens à souligner que sa réalisation était d'un très grand bénéfique pour moi car c'était une bonne occasion pour consolider mes connaissances théoriques dans le domaine du management et de la R&D au sein d'une équipe de développeurs de jeux vidéo.

Il est évidement que ce projet n'est ni une œuvre parfaite ni un projet applicable dès sa conception mais j'ai tenu à ce qu'il soit à la fin de ce stage à la hauteur de mes espérances professionnelles ainsi qu'à la société qui m'a fait confiance tout au long de ces trois mois en espérant qu'elle trouve dans mon travail une bonne solution pour les différents problèmes que cette dernière rencontre.

En ce qui concerne les questions posées dans le chapitre 1 :

Il est difficile de mettre, au moment de la rédaction de ce rapport, un OUI ou un NON ferme à toute ces questions car le nouveau pipeline doit faire ces preuves dans le monde du réel avant de dire avec certitude qu'elle répond aux attentes de Lanterns. Mais malgré tout nous allons essayer de répondre suivant le code couleur vert – orange – rouge. Cette notation est, bien sûr, subjective. Plus la couleur est claire, plus elle tend vers un OUI.

- Est-ce qu'elle est assez claire pour n'importe quelle recrue ?
- Est-ce qu'elle est conforme aux expériences que, des sociétés ayant déjà développé des jeux, ont expérimentés ?
- Est-ce qu'elle est conforme au business model et à l'essence même du produit final ?
- Est-ce qu'elle est adaptable a tous types de projets (jeux vidéo) ?

Merci d'avoir pris le temps de lire jusqu'ici.

Bibliographie / Webographie

Lives :

- Theory of Fun for Game Design, *Raph Koster*. (relecture)
- Inside the Video Game Industry, *Ryan Kaufman, Ken S. McAllister, Judd Ethan Ruggill, Randy Nichols.*

Web:

- https://medium.com/productschool/hey-listen-heres-how-to-be-a-gaming-pm-85759eecf28a
- https://clickup.com/
- https://www.perforce.com/blog
- https://www.youtube.com/watch?v=XvO-vSiIs_A

Formation Web:

- https://www.udemy.com/course/gettingstartedingames/learn/lecture/13466634?start=0#overview (relecture)

Annexe 1 L'école en chiffres et géolocalisation

EBS est l'unique université privée qui propose des masters en alternance en Tunisie. Le système de formation en alternance vient répondre aux exigences des entreprises à la recherche de jeunes diplômés avec expériences professionnelles. Avec un rythme de 3 jours à l'université et 2 jours à l'entreprise, nos étudiants en master professionnel bénéficient d'un encadrement tout au long de leurs parcours de formation. Les alternants se voient généralement proposer des contrats par leurs entreprises d'accueil et évitent ainsi les longues recherches d'emploi une fois leurs diplômes en poche.

EBS compte 98% d'étudiants diplômés

Plus de 20 entreprises partenaires

Plus de 40 enseignants qualifiés

15 Diplômes approuvés

https://www.ebs.tn/

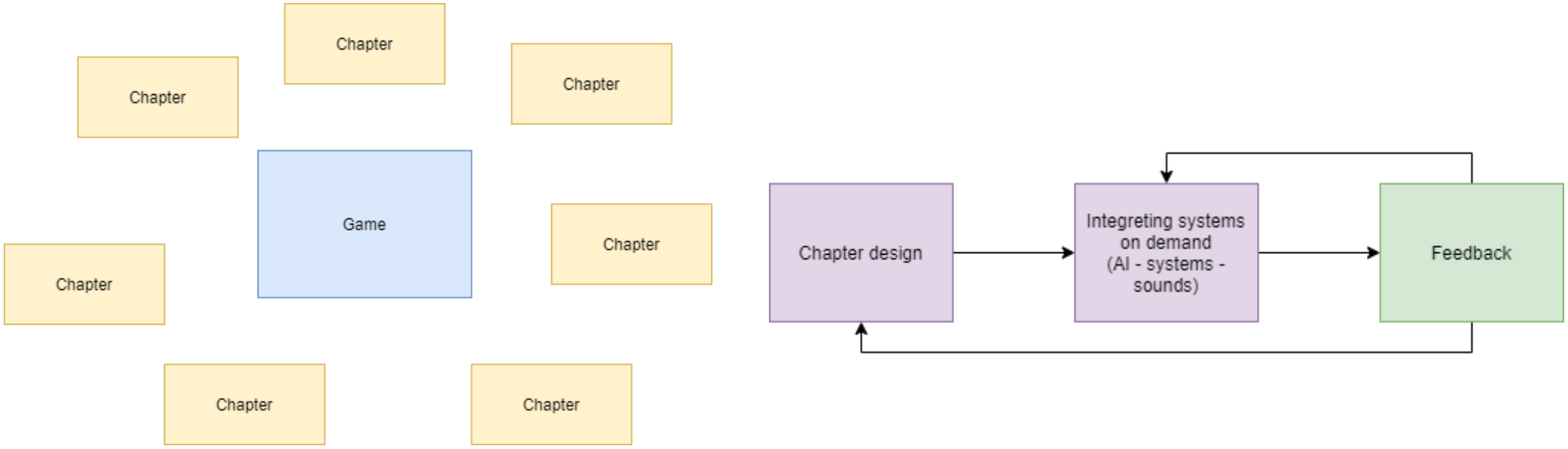
Berges du Lac III, Tunis - TUNISIE.

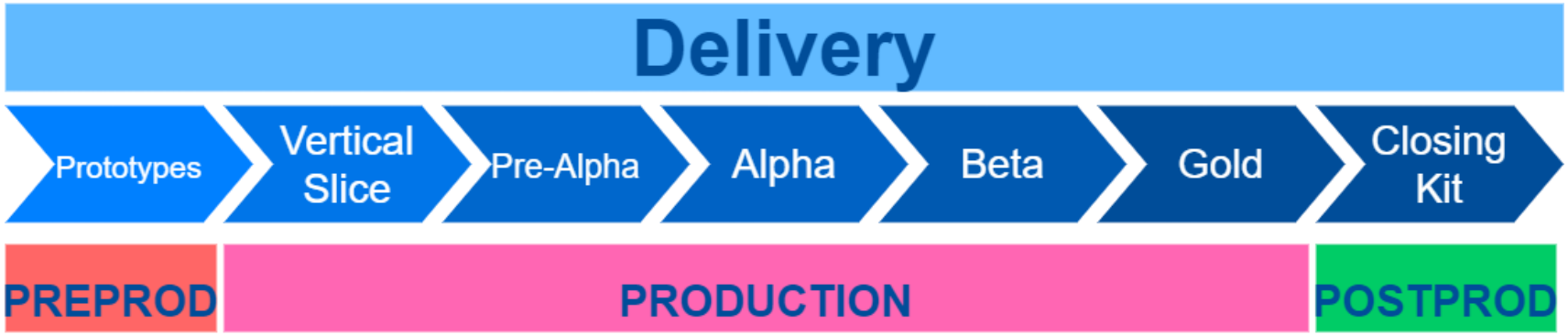
Annexe 2 Sources utilisées

Toutes les sources et références (chiffres, figures, vidéos et textes) utilisées lors de la réalisation du rapport. Ces dernières sont regroupées dans cette annexe.

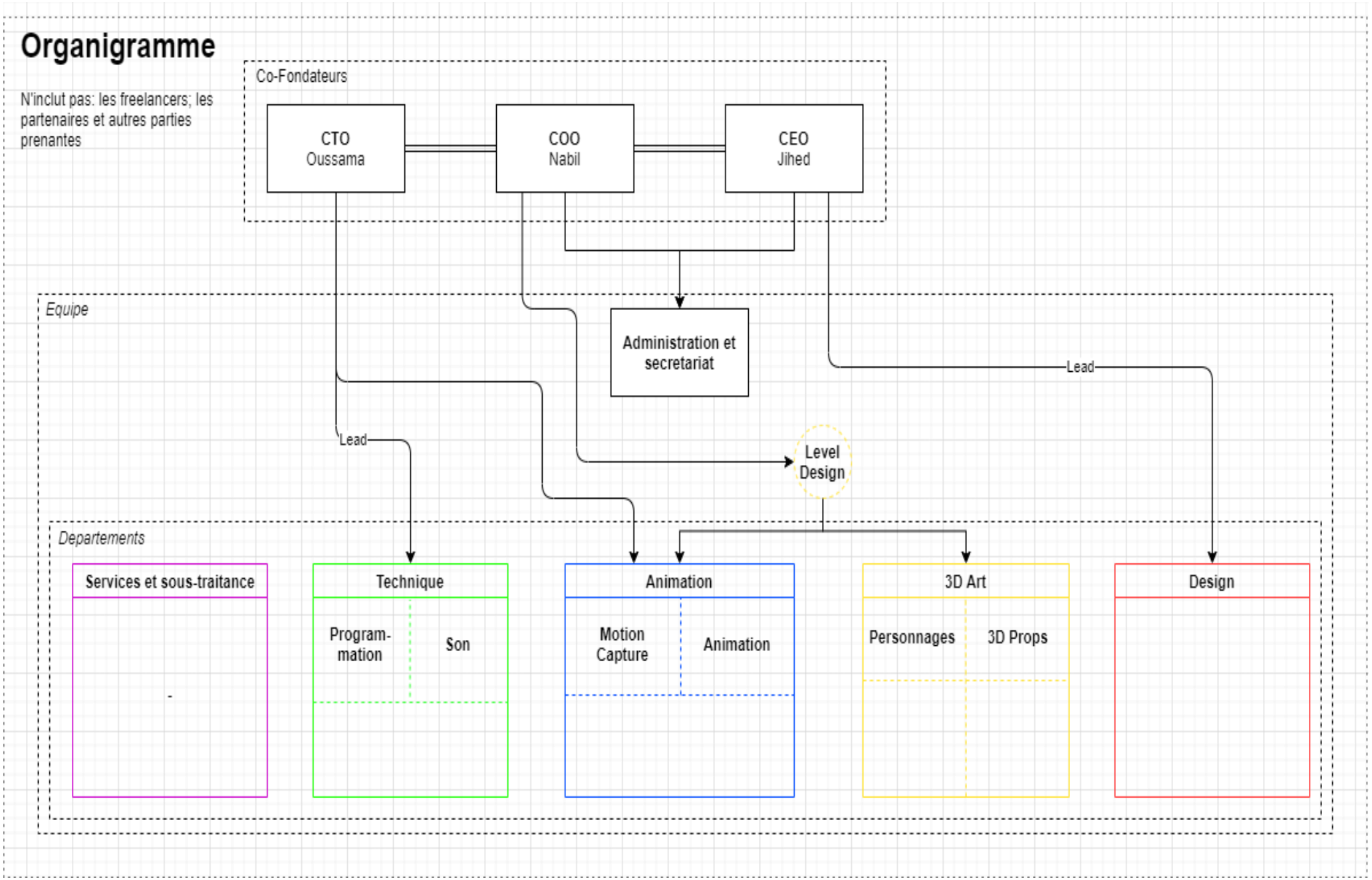
Figures des chapitres et de la page bonne lecture ..... N/A
hthttps://undraw.co/
Figure 1.1 : Logo Unreal Engine 5 07
https://www.logo.wine/logo/Unreal_Engine
Figure 1.2 : Logo Lanterns Studios 09
https://www.facebook.com/LanternsStudios/photos/a.294472957764017/567703087107668
Figure 1.3 : Lanterns MoCap 09
https://www.facebook.com/LanternsStudios/photos/966329477245025
Figure 1.5 : Vue sur l'espace de travail 12
https://www.facebook.com/LanternsStudios/photos/973939973150642
Figure 1.6 : Logo de git 14
https://git-scm.com/downloads/logos
Figure 1.8 : Les logos de ClickUp 17
https://clickup.com/help
Figure 2.3 : Logo Machinations 23
https://machinations.io/about/
Figure 2.8 : Cycle de vie d'un produit 38
https://www.expertinbox.com/2013/07/18/le-cycle-de-vie-du-produit/
Figure 2.9 : Logo Perforce 41
https://www.perforce.com/
Figure 2.10 : Logo Gitlab 42
https://dev.to/gugurel/gitlab-wiki-page-tips-5f10
Figure 2.11 : Logo Hack'n'Plan 43
https://hacknplan.com/
Figure 3.3 : Level Design en parallèle 50
https://www.researchgate.net/figure/Game-production-pipeline-overview-Concept-content-creation-pipeline-level-design_fig2_267417785
Ref ig1 : Définitions sur les jeux vidéo 01
https://fr.wikipedia.org/wiki/Jeu_vid%C3%A9o
Ref ig2 : Plateformes de gaming et cartes graphiques 02
https://www.gamasutra.com/features
Ref 1.1 : Définition du Workflow 05
https://www.wfmc.org : *le site du Workflow Management Coalition, un groupement d'acteurs de l'univers du Workflow et du BPM*
Ref 1.2 : Définition du jeu vidéo indépendant 06
https://fr.wikipedia.org/wiki/Jeu_vid%C3%A9o_ind%C3%A9pendant
Ref 1.3 : Définition d'un jeu AAA 07
https://fr.wikipedia.org/wiki/AAA_(jeu_vid%C3%A9o)

Annexe 3 Utilitaire de la soutenance

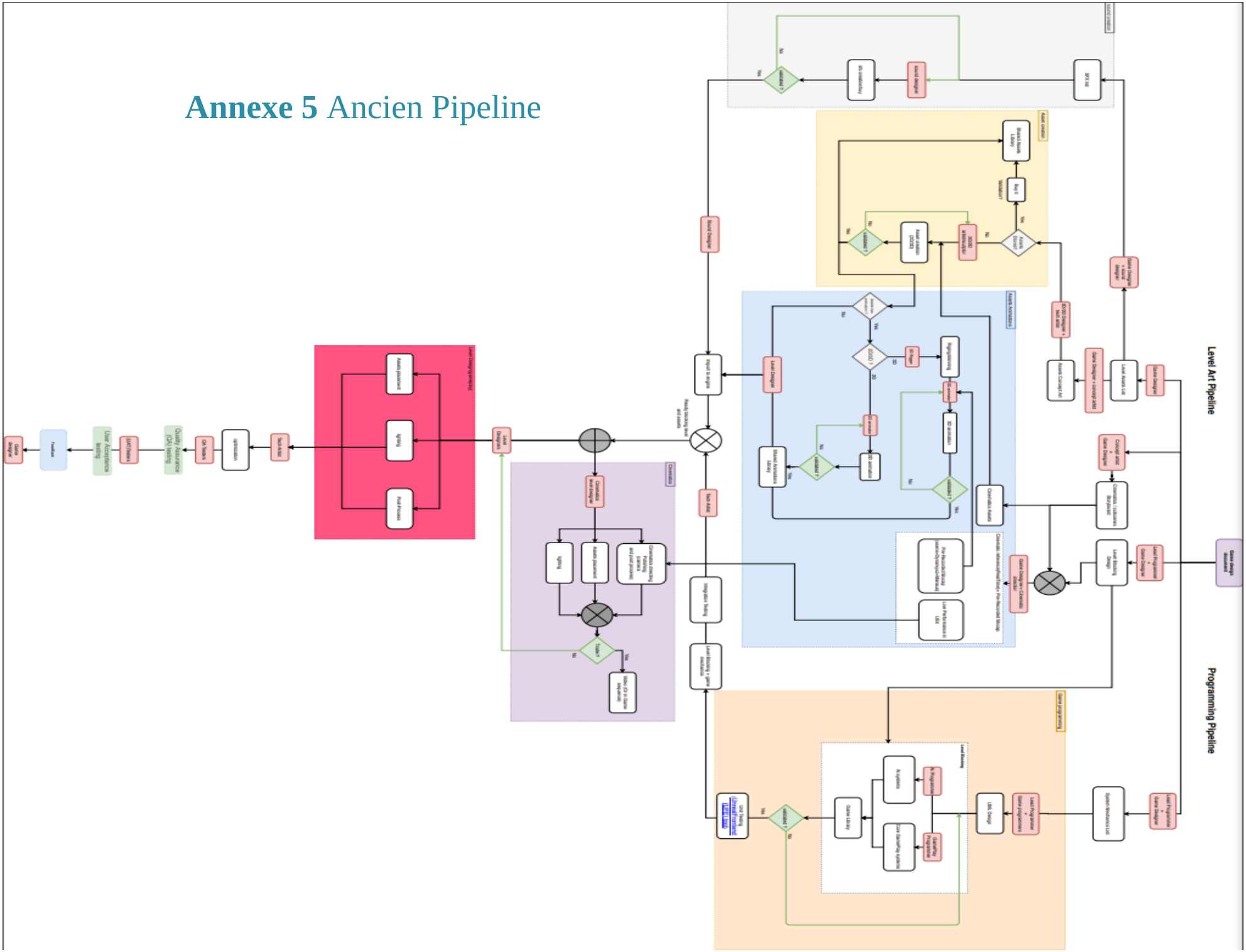




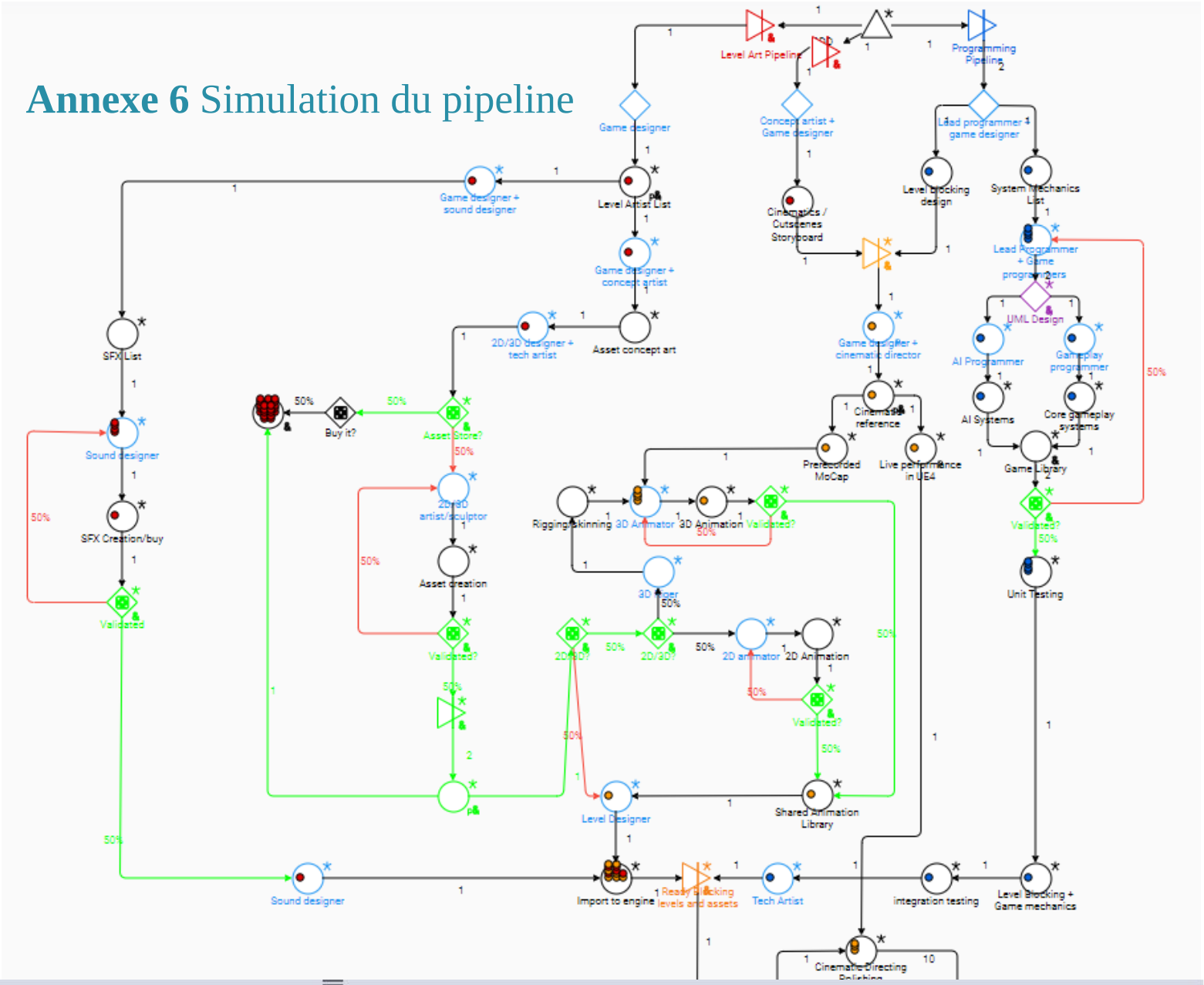
Annexe 4 Organigramme de l'entreprise



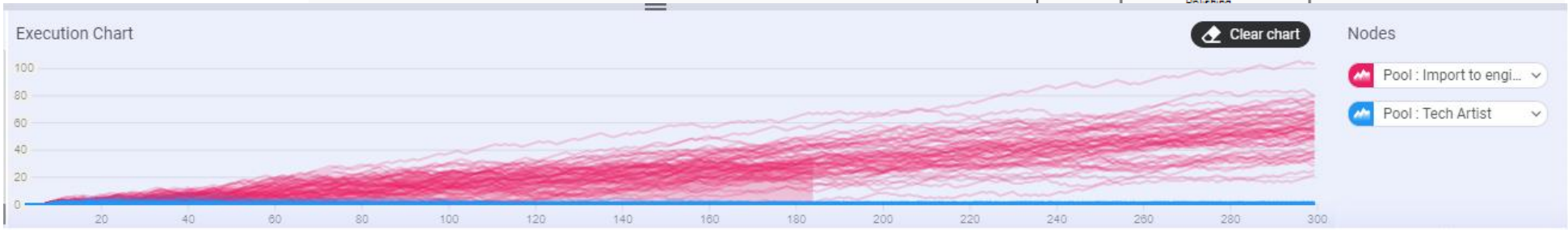
Annexe 5 Ancien Pipeline



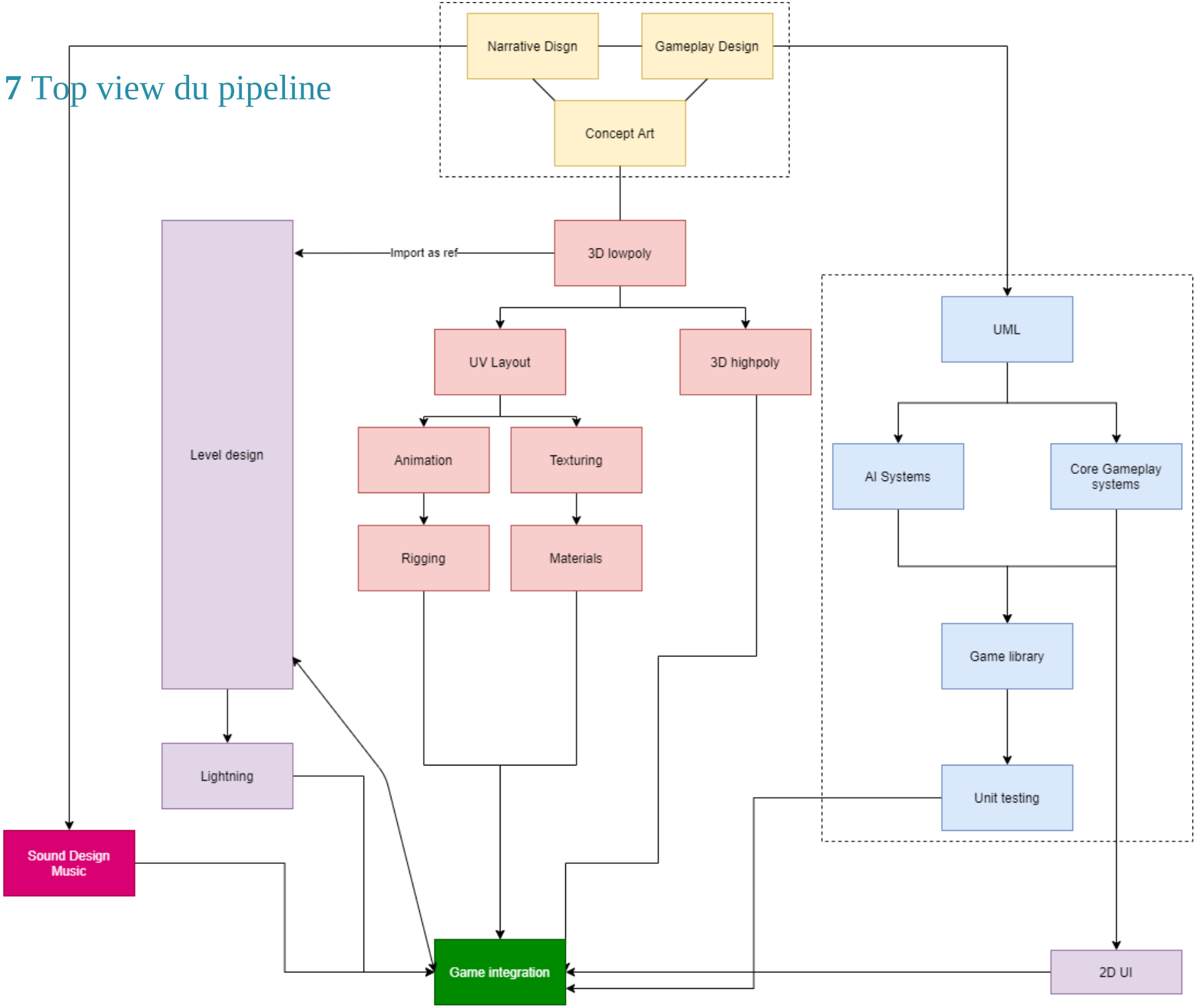
Annexe 6 Simulation du pipeline



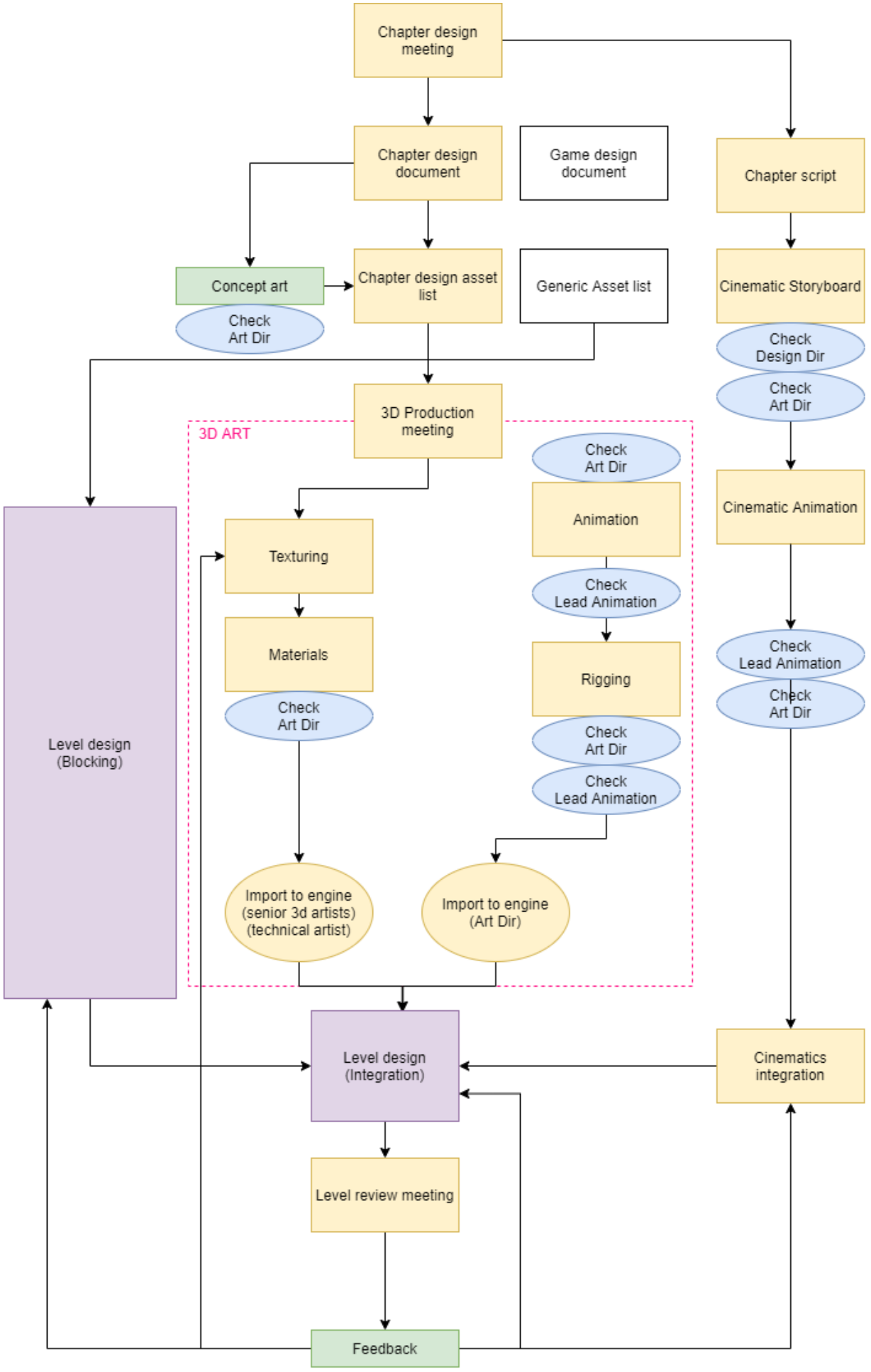
Lien de la simulation :
https://machinations.io/community/uchromia/lanterns-pipeline-06b222fcf10de5a557ed994674094b181/



Annexe 7 Top view du pipeline



Annexe 8 Décomposition en cycle chapitrale



Just cause 4 - natalie asport

Abstract

Français

Ce stage consiste à la mise au point d'une charte illustrant le process de gestion des taches et des livrables pour un projet dont le produit final est un jeu vidéo pour le compte de l'entreprise : Lanterns Studios.

Ce process appelé « pipeline » va notamment permettre aux équipes actuelles de visualiser tout le processus et les tâches à faire lors de la création d'un jeu vidéo en 3D mais aussi au top management d'avoir un moyen d'expliquer aux potentiels nouveaux membres, le déroulement du développement et situer leurs tâches par rapport aux autres tâches des autres membres/rôles. Ce pipeline est une mise à jour d'un pipeline déjà existante au tout début de l'entreprise mais prenant en compte les modifications et changements de leur workflow que l'entreprise a effectué jusqu'à aujourd'hui.

Le stage consiste aussi à trouver un moyen de gérer efficacement les ressources et fichiers partagés par toutes les équipes et leurs versions.

En d'autres termes, le stage consiste à améliorer et rendre plus rapide le workflow des équipes de développeurs de jeux vidéo.

Mots clés : Pipeline – Process – Jeux vidéo – Lanterns Studios – Versioning

Réalisé par Hedi MLIKA

English

This internship consists of the development of a chart that illustrates the process for managing tasks, workflow and deliverables for a project whose final product is a video game for the company: Lanterns Studios.

This process called "pipeline" will, in particular, allow the current teams to visualize the whole process and the tasks to be done during the creation of a 3D video game, but also, to the top management so they have the means of explaining to the potential new members, the development progress and situate their tasks in relation to the other tasks of other members / roles. This pipeline is an update of a pipeline that already existed at the very beginning of the company but takes into account the modifications and changes in their workflow that the company has made to date.

The internship also consists of finding a way to efficiently manage the resources and files shared by all the teams and their versions.

In other words, the internship consists of improving and making faster the workflow of video game developers / teams.

Keywords: Pipeline – Process – Video games – Lanterns Studios – Versioning

Realized by Hedi MLIKA

EBS ESPIMA BUSINESS SCHOOL

Lanterns