



REVENGE'S ISLAND

DJEBARNI Imad-Eddine
BELTRAN Romaric

Table des matières

Chapitre 1 : Présentation du projet	2
Chapitre 2 : Description de la demande	2
Chapitre 3 : Contraintes	3
Chapitre 4 : Déroulement du projet	3
Tâche 0 : Rédaction du cahier des charges	3
Tâche 1 : Mise en place du diagramme des modules	3
Tâche 2 : Création du jeu en mode texte	3
Tâche 3 : Création du jeu en mode graphique	5
Tâche 4 : Création du jeu en mode graphique avec des bruits sonores et sauvegarde	6
Annexes	7
Diagramme de Grantt	7
Diagramme des modules	8

Chapitre 1 : Présentation du projet

AN 750 après JC. Ragnar Lothbrock roi des vikings décide de partir à la conquête de contrées inconnues occidentales. Terres lointaines, hostiles ou redoutées tel que tous ceux qui s'y sont aventuré n'y sont jamais revenus. Réunifiant les Yarls des quatre coins du pays, il prend la mer à bord de son sinistre drakkar et de toute sa flotte de guerriers sanguinaires. S'en suit des mois d'abordages et de pillages, ignorant la colère des dieux qui finit par s'abattre sur le navire par une tempête aux allures cataclysmique. Après plusieurs jours de combats acharnés contre les éléments, Ragnar voit enfin une terre à l'horizon. Odin n'en ayant pas fini avec lui, il se réveille sur une plage et observe les alentours. Il n'y a que l'épave de son navire mais hurlements, doléances et peurs résonnent encore dans son esprit. Il n'est pas seul sur cette île. Un vent glaçant souffle, l'effroyable Ragnar, hache en main compte bien affronter tous les dangers pour sauver les siens.

Notre client fasciné par cette histoire nous donne deux mois pour coder un jeu d'aventure de Ragnar sur cette île. Ce document est le cahier des charges de ce projet qui définit les tâches à suivre. Il sera remis à notre client et comprends une description détaillée de l'application, une liste exhaustive et détaillée des tâches à réaliser, un diagramme de Gantt ainsi qu'un diagramme des modules de bases du projet.

Chapitre 2 : Description de la demande

Après le lancement de Revenge Island, on a d'abord une fenêtre avec un menu proposant la création ou le chargement d'une partie. Ensuite, le contexte est introduit par un cours récit et le premier niveau s'enclenche. Le joueur contrôle un personnage sur un terrain en vue plongeante, et doit éliminer les ennemis qui se dressent devant lui. Une fois la mission accomplie, le joueur passe au niveau suivant et la difficulté croît. Il y a plus d'ennemis et ils sont plus coriaces. Les niveaux s'enchaînent et le joueur récupère des alliés qui vont l'aider à éliminer ses ennemis. Au niveau final, le personnage fait face à un boss qui doit être terrassé de façon particulière.

Le personnage se déplace de bas en haut et de gauche à droite. Il lance des haches pour abattre ses cibles et doit éviter les tirs ennemis ou se cacher derrière un obstacle. Les alliés jettent également des haches mais n'ont pas la possibilité de se déplacer. Le héros, les ennemis et les obstacles disposent tous de points de vie variables. Lorsque leurs points de vie sont à 0, ils disparaissent. Si le héros disparaît il perd un cœur. Il dispose de 5 cœurs pour finir le jeu et dans le cas où il épuise ces derniers, il doit recommencer la partie à zéro. Le dernier ennemi est plus imposant et la stratégie de combat pour en venir à bout n'est pas dévoilée.

Lorsque le boss est tombé, la partie se termine par une conclusion "récit" et un message de félicitation. On évalue un score à partir des statistiques de tirs, de la durée et de l'accomplissement des niveaux qui est alors dévoilé. On a un fond sonore durant toute la partie.

Chapitre 3 : Contraintes

Notre budget est celui d'étudiants universitaires, et nous développons sur nos propres machines. Nous avons trois mois pour aboutir à une version finale fonctionnelle. Nous commencerons le projet par la réalisation d'un diagramme des modules. Celui-ci nous permettra de mieux visualiser le partage des tâches, et de suivre un chemin ordonné pour gagner du temps. Nous coderons en C++ sous Linux/Windows. Les bibliothèques utilisées seront "conio.h/windows.h" pour la version txt, et SDL2 pour la version graphique. L'indentation et la propreté seront primordiales à la réalisation du code. Doxygen nous permettra de le documenter et l'espace d'échange et de stockage utilisé sera la Forge LYON 1 via Mercurial. Nous vérifierons l'absence de bug et de fuites mémoires via gdb et valgrind. L'IDE utilisé sera CodeBlocks la plupart du temps.

Chapitre 4 : Déroulement du projet

Ψ Tâche 0 : Rédaction du cahier des charges

Ressources humaines : Les deux membres du groupe.

Permanence : 1 semaine.

Ψ Tâche 1 : Mise en place du diagramme des modules

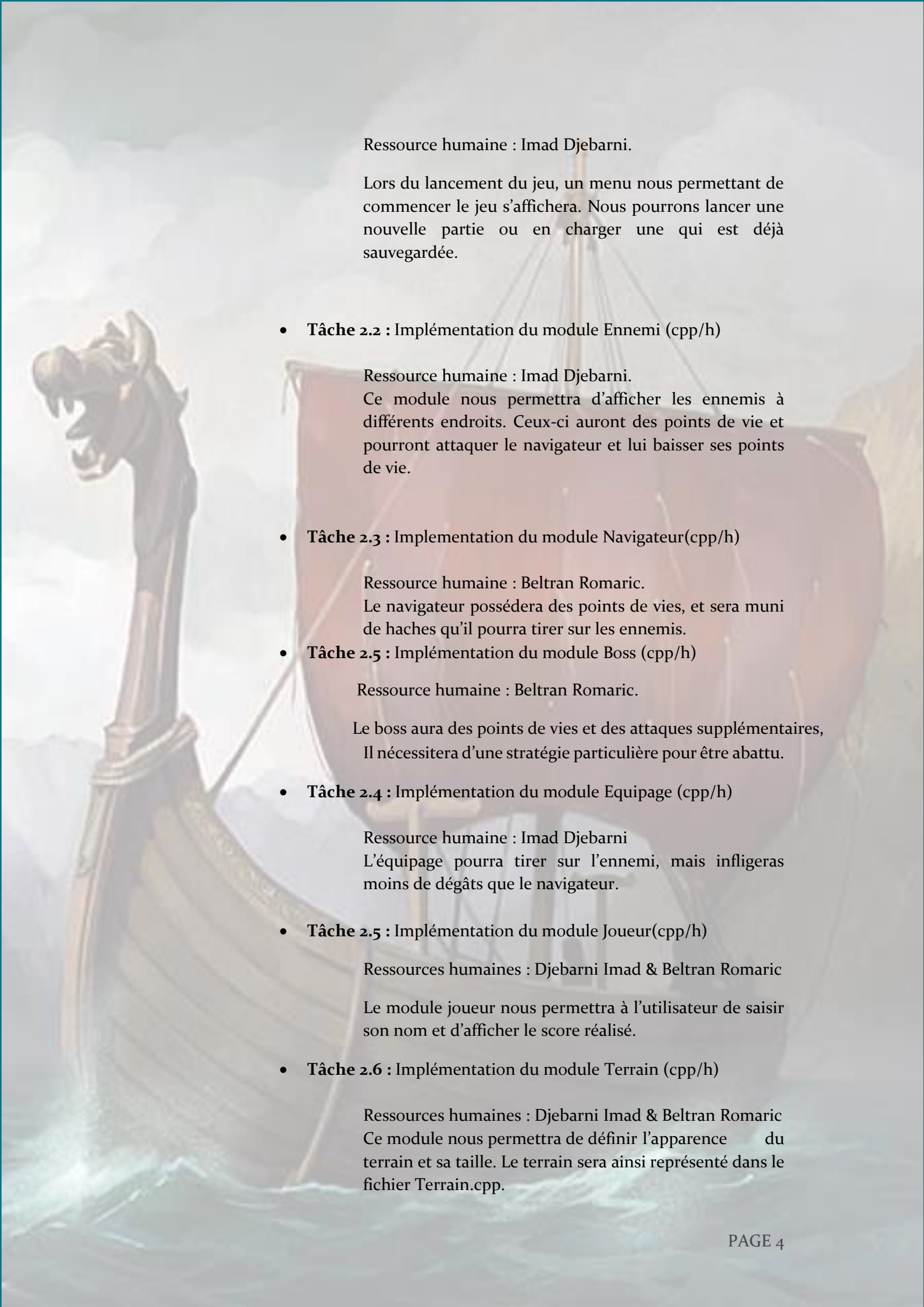
Ressources humaines : Les deux membres du groupe.

Permanence : 1 semaine.

Ψ Tâche 2 : Création du jeu en mode texte

Permanence : 2 semaines.

- Tâche 2.1 : Codage d'un menu (cpp/h)



Ressource humaine : Imad Djebarni.

Lors du lancement du jeu, un menu nous permettant de commencer le jeu s'affichera. Nous pourrons lancer une nouvelle partie ou en charger une qui est déjà sauvegardée.

- **Tâche 2.2 : Implémentation du module Ennemi (cpp/h)**

Ressource humaine : Imad Djebarni.

Ce module nous permettra d'afficher les ennemis à différents endroits. Ceux-ci auront des points de vie et pourront attaquer le navigateur et lui baisser ses points de vie.

- **Tâche 2.3 : Implémentation du module Navigateur(cpp/h)**

Ressource humaine : Beltran Romaric.

Le navigateur possédera des points de vies, et sera muni de haches qu'il pourra tirer sur les ennemis.

- **Tâche 2.5 : Implémentation du module Boss (cpp/h)**

Ressource humaine : Beltran Romaric.

Le boss aura des points de vies et des attaques supplémentaires, Il nécessitera d'une stratégie particulière pour être abattu.

- **Tâche 2.4 : Implémentation du module Equipage (cpp/h)**

Ressource humaine : Imad Djebarni

L'équipage pourra tirer sur l'ennemi, mais infligera moins de dégâts que le navigateur.

- **Tâche 2.5 : Implémentation du module Joueur(cpp/h)**

Ressources humaines : Djebarni Imad & Beltran Romaric

Le module joueur nous permettra à l'utilisateur de saisir son nom et d'afficher le score réalisé.

- **Tâche 2.6 : Implémentation du module Terrain (cpp/h)**

Ressources humaines : Djebarni Imad & Beltran Romaric
Ce module nous permettra de définir l'apparence du terrain et sa taille. Le terrain sera ainsi représenté dans le fichier Terrain.cpp.

- **Tâche 2.7 : Implémentation du module Niveau (cpp/h)**

Ressources humaines : Djebarni Imad & Beltran Romaric

Chaque terrain est associé à un niveau. De plus à chaque fois qu'il y a un changement de terrain les ennemis sont de plus en plus redoutables.

- **Tâche 2.8 : Implémentation du module Jeu (cpp/h)**

Ressources humaines : Djebarni Imad & Beltran Romaric
Celui-ci nous permettra d'appeler toutes les fonctions utiles au fonctionnement du jeu : Déplacement, tir, score...

- **Tâche 2.9 : Implémentation de JeuTxt (cpp/h)**

Ressources humaines : Djebarni Imad & Beltran Romaric
Nécessite un appel à toutes les fonctions et procédures permettant d'afficher le jeu en mode texte.

Ψ **Tâche 3 : Développement en mode graphique.**

- **Tâche 3.1 : Implémentation des fonctions graphiques**

Ressources humaines : Beltran Romaric & Djebarni Imad
Cela représente la conception de l'apparence du jeu : Décor, Personnage, Ennemis

- **Tâche 3.2 : Implémentation de jeuSDL (cpp/h)**

Ressources humaines : Beltrain Romaric & Djebarni Imad
Cela nous permettra d'afficher le jeu en mode graphique.

Ψ **Tâche 4 : Création du jeu en mode graphique avec bruits sonores et sauvegarde**

Permanence : 1 semaine.

Ressources humaines : Beltran Romaric & Djebarni Imad.

Cette étape nous permettra de finaliser la conception et le développement de ce jeu.



Annexes

Diagramme de Grantt :

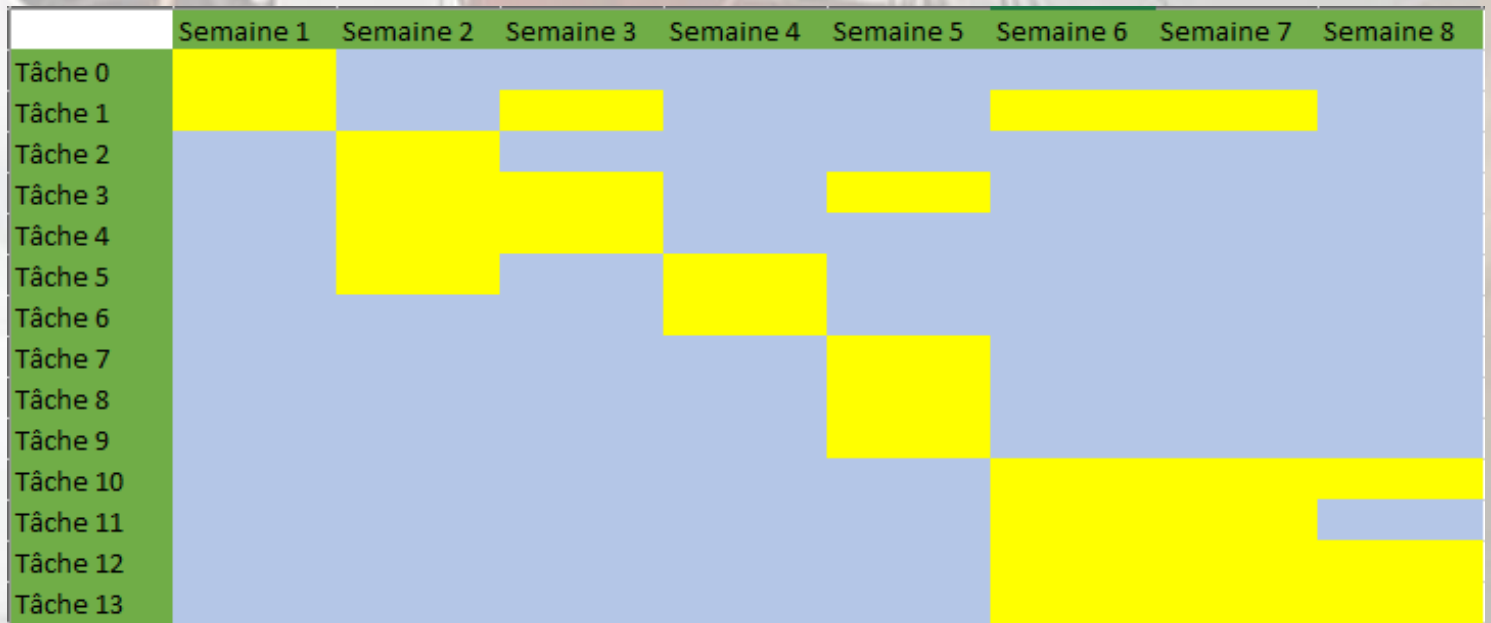


Diagramme des modules :

