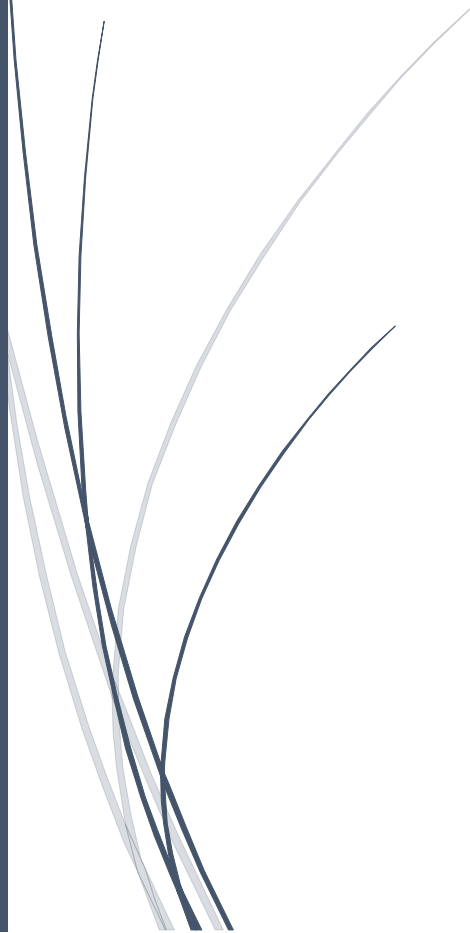


A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from this bar, containing the date.

12/03/2020

MilleBourne (Mille Bornes like)

Cahier des Charges

Several thin, curved lines in dark blue and light grey originate from the bottom left and sweep upwards and to the right.

Benjamin FORISSIER, Emie LAFOURCADE, Judith
MILLET, Antoine RYBACKI
NOM DU GROUPE : MILLEBOURNE

Sommaire :

SOMMAIRE :	1
CHAPITRE 1 : PRESENTATION DU PROJET	2
CHAPITRE 2 : DESCRIPTION DE LA DEMANDE	3
<i>Chapitre 2.1 : Description de la demande</i>	3
<i>Chapitre 2.2 : Règles du jeu</i>	3
CHAPITRE 3 : CONTRAINTES	4
CHAPITRE 4 : DEROULEMENT DU PROJET	5
<i>Liste des tâches :</i>	5
➤ Tâche 1 :	5
➤ Tâche 2 :	5
❖ Tâche 2.1 :	5
❖ Tâche 2.2 :	5
❖ Tâche 2.3 :	5
❖ Tâche 2.4 :	5
➤ Tâche 3 :	6
❖ Tâche 3.1 :	6
❖ Tâche 3.2 :	6
➤ Tâche 4 :	6
❖ Tâche 4.1 :	6
➤ Tâche 5 :	7
➤ Tâche 6 :	7
❖ Tâche 6.1 :	7
❖ Tâche 6.2 :	7
➤ Tâche 7 :	8
❖ Tâche 7.1 :	8
CHAPITRE 5 : DIAGRAMME DE GANTT	8
CHAPITRE 6 : DIAGRAMME DES CLASSES	9

Chapitre 1 : présentation du projet

Qui : Benjamin FORISSIER, Emie LAFOURCADE, Judith MILLET, Antoine RYBACKI.

Client : Le personnel enseignant en charge.

Contexte : Dans le cadre de leurs études, les étudiants susnommés doivent réaliser un projet ensemble.

Pourquoi sommes-nous quatre ?

Le projet étant assez conséquent puisque nous avons prévu de faire une partie réseau sur le jeu et de permettre à un utilisateur de jouer en ligne, nous ne serons pas trop pour travailler sur les différents aspects du projet.

Chapitre 2 : Description de la demande

Chapitre 2.1 : Description de la demande

Résultats visés : Un jeu de type Mille-Bornes en graphique et texte nommé MilleBourne.

Fonctionnalités :

- Jouer à un Mille-bornes classique.
- Jouer en réseau avec plusieurs joueurs, de 2 à 4 joueurs.
- Jouer seul avec des ordinateurs.
- Pouvoir changer les règles et personnaliser son jeu.

Chapitre 2.2 : Règles du jeu

Règles par défaut :

Le but du MilleBourne est d'être le premier joueur à effectuer 1000 kilomètres précisément. Pour cela, il vous faudra éviter les pièges de la route : Pannes d'essence, feux rouges, crevaisons, accidents et limitations de vitesses.

Pour commencer une partie, chaque joueur se voit distribuer 6 cartes qu'il ne doit pas montrer aux autres. Le reste du paquet servira de pioche. Pour commencer la partie, le premier joueur doit piocher une carte. S'il a en sa possession un feu vert, il pose cette carte devant lui et peut donc commencer sa partie. Le joueur peut aussi poser une carte « Botte » qui le permet d'être immunisé contre certaines attaques ou encore une carte « Attaque » devant l'un de ses adversaires. Si le joueur ne peut ou ne désire pas poser une carte, il doit obligatoirement en jeter une afin d'avoir toujours 6 cartes dans sa main.

C'est ensuite au tour du joueur suivant de jouer. Lui aussi a alors plusieurs possibilités. En plus de celles énumérées ci-dessus, il peut contrer une attaque en posant au-dessus le contraire de celle-ci pour l'annuler. Exemple : Il pose une carte « fin de limitation de vitesse » sur une carte « Limitation de vitesse ». Il doit ensuite poser un feu vert avant de pouvoir redémarrer.

Autre choix : le joueur peut également attaquer le joueur qui a posé un feu vert en posant une carte feu rouge.

Le gagnant de la partie est celui qui a réussi à poser exactement 1000 kilomètres devant lui à l'aide des cartes « distance » ou celui ayant atteint le plus de kilomètres s'il n'y a plus de cartes.

Règles adaptables :

Le coup fourré :
Lorsqu'un adversaire vous attaque et que vous avez dans votre jeu la botte correspondante pour parer l'attaque, vous pouvez la poser même si ce n'est pas votre tour de jouer. Vous empochez ainsi 300 kilomètres supplémentaires. De plus vous pouvez rejouer immédiatement, après avoir pioché une carte.

Les feux verts :
Il est possible de jouer sans devoir poser un feu vert après avoir contré une attaque.

Les cartes :
Possibilité d'adapter le nombre de cartes.

Chapitre 3 : Contraintes

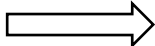
Coût : Pas de frais.

Durée de développement : Rendu le 4 Mai au plus tard.

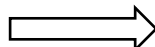
Présentation orale du projet le 5 Mai.

Liste des tâches :

➤ Tâche 1 :

- Brainstorming sur la conception du projet en groupe.
- Livrable  Cahier des charges.
- Réalisé quand le cahier des charges sera entièrement fini.

➤ Tâche 2 :

- Conception de la première version du jeu en version console.
- Livrable  Des premières versions du code source.
- Réalisé quand on aura une première version du logiciel opérationnelle en mode console.

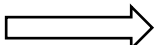
❖ Tâche 2.1 :

- Réalisation de la classe Joueur.
- Livrable  Un fichier Joueur.cpp et .h

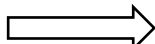
❖ Tâche 2.2 :

- Réalisation de la classe Paquet.
- Livrable  Un fichier Paquet.cpp et .h

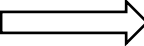
❖ Tâche 2.3 :

- Réalisation de la classe Défausse.
- Livrable  Un fichier Defausse.cpp et .h

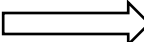
❖ Tâche 2.4 :

- Réalisation de la classe Jeu.
- Livrable  Un fichier Jeu.cpp et .h

➤ Tâche 3 :

- Conception de la deuxième version du jeu en implémentant des bots (ordinateurs).
- Livrable  Des nouvelles versions du code source comprenant les ordinateurs et les intégrant au jeu.
- Réalisé quand on aura une 2^e version du jeu intégrant des bots fonctionnels qui se comportent comme des joueurs normaux.

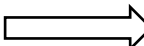
❖ Tâche 3.1 :

- Réalisation de la classe Bots.
- Livrable  Un fichier Bots.cpp et .h

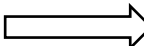
❖ Tâche 3.2 :

- Implémentation de la classe Bots dans le jeu.

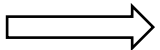
➤ Tâche 4 :

- Conception de la 3^e version du jeu en implémentant un affichage console.
- Livrable  Une mise à jour du code source contenant un affichage complet en mode console.
- Réalisé quand le jeu sera opérationnel avec un affichage console.

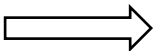
❖ Tâche 4.1 :

- Un débogage complet et vérification du code pour tout afficher correctement.
- Livrable  Un jeu fonctionnel et jouable en mode console.

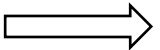
➤ Tâche 5 :

- Conception de la 4^e version du Jeu en implémentant un système de personnalisation du jeu via des options de changement de règles comme le nombre de cartes d'un certain type, la taille du paquet, la possibilité d'activer ou non la règle des feux verts ou celle du coup fourré.
- Livrable  Un jeu personnalisable par le créateur de la partie via différentes options citées ci-dessus dans le chapitre 2 - sous-titre 2.2 – section « règles adaptables ».
- Réalisé quand le créateur de la partie pourra personnaliser sa partie comme cité ci-dessus.

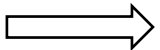
➤ Tâche 6 :

- Conception de la 5^e version du jeu en implémentant un mode de jeu à plusieurs joueurs en réseau.
- Livrable  Un jeu opérationnel à plusieurs en réseau.
- Réalisé quand plusieurs joueurs humains et non machines pourront jouer ensemble, selon les règles définies par le créateur de la partie et non selon leurs règles.

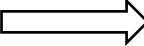
❖ Tâche 6.1 :

- Conception des classes Client/Serveur
- Livrable  Un fichier Client.h/.cpp et serveur.h/.cpp.

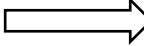
❖ Tâche 6.2 :

- Implémentation de la classe Client/Serveur dans le logiciel.
- Livrable  Un logiciel comprenant les classes Client/Serveur opérationnelles.
- Réalisé quand le logiciel pourra créer des parties en réseau sans bug majeur qui empêche le déroulement correct d'une partie.

➤ Tâche 7 :

- Conception de la 6^e version du jeu en implémentant un affichage graphique.
- Livrable  Un jeu dans sa version « finale » qui permet un affichage graphique de la partie en cours ainsi que du Menu et des différentes fonctionnalités prévues pour une partie.
- Réalisé quand le jeu permettra un affichage graphique de la partie, de la main du joueur, des cartes « en main », du menu, des différentes options de règles possibles, des cartes adverses une fois jouées sur le plateau et des autres fonctionnalités qui découlent de ces fonctionnalités.

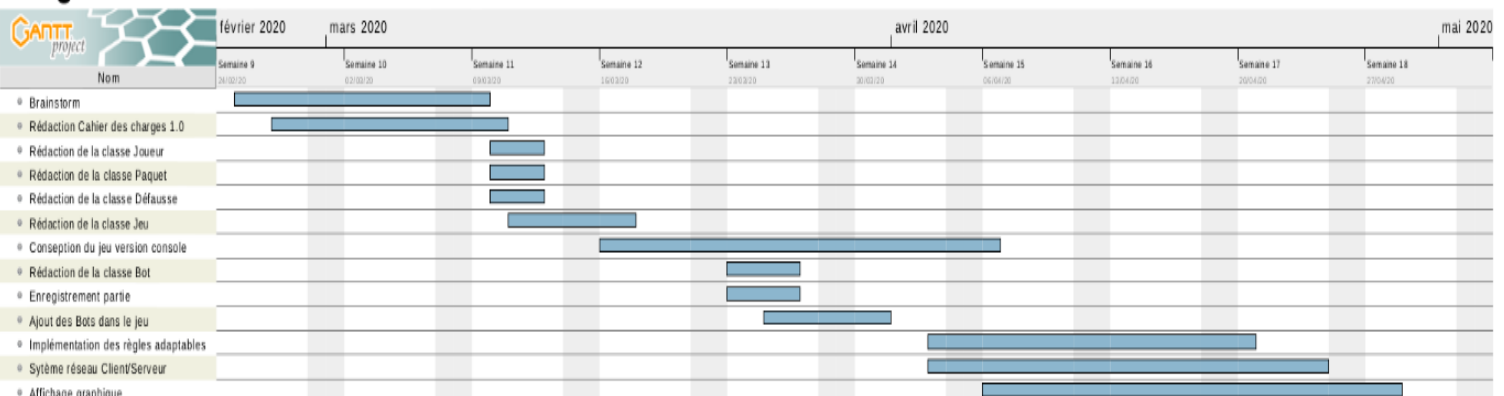
❖ Tâche 7.1 :

- Implémentation de l’affichage graphique du jeu et débogage de celui-ci.
- Livrable  Un jeu opérationnel qui affiche une partie en temps réelle et qui permet une lecture simple et rapide du jeu, ainsi qu’une compréhension facilitée sur ce qui se passe dans la partie.

Chapitre 5 : Diagramme de Gantt

Diagramme de Gantt

3



Chapitre 6 : Diagramme des classes

