Ecole Publique d'Ingénieurs en 3 ans

Rapport

PROJET GENIE LOGICIEL

Le 17 novembre 2022,
version 1.0

Youssef AGHZERE
Yahya BOUCHIBTI-FAIZ
Mohamed CHERGUI
Ayoub ED-DAHMANY
Aymane EL-OTMANI
Colin HARTVICK
Mohamed-El-Mokhtar SIDI-ABDALLAH
Saad ZTOUTI

www.ensicaen.fr

TABLE DES MATIERES

TABLE DES FIGURES

TABLE DES TABLEAUX

REGATE

1. Introduction

L'objectif du projet était de créer un logiciel de jeu solo destiné à l'apprentissage de la navigation en voilier. Dans le jeu, le joueur doit réaliser un parcours en passant autour de bouées. Pour cela, le joueur agit sur la direction du bateau qu'il contrôle. En fonction de la force du vent et de sa direction, le comportement du bateau change. Ce logiciel été réalisé avec le logiciel IntelliJ IDEA et le langage Java avec sa bibliothèque graphique JavaFx.

2. Analyse des risques

NATURE DU RISQUE	IMPACT SUR LE PROJET	GRAVITE	PROBABILITE	CRITICITE	MESURE DE PREVENTION OU DE REDUCTION	RESPONSABLES
Empêchement urgent d'un ou plusieurs membres de l'équipe	Diminution de l'effectif, et donc diminution de la rentabilité	Grave	Moyenne		Distribution de tâches flexibles entre les membres de l'équipe	Chef de projet
Défaillance du matériel (dysfonctionnement d'un ou de plusieurs ordinateurs ...)	Perte de données et de l'avancement du projet	Très grave	Rare		Utilisation d'un logiciel pour la distribution des données (Git ...)	Chef de projet
Le non-respect des délais des livrables de chaque séance	Non-respect du délai du rendu du logiciel final au client	Grave	Fréquent		Membres d'équipe organisés et préventifs	Responsable de versions

Mauvaise modélisation du problème du projet	Projet non conforme avec les exigences du client	Très grave	Rare		Donner une très grande importance à la compréhension et la modélisation	Chef de projet Architecte
La méconnaissance d'un outil de développement (Java FX ...)	L'interruption de la continuité du travail	Grave	Moyenne		Affecter à un seul membre de l'équipe la mission de se former pour accomplir la tâche	Chef de projet Développeurs

Tableau 1 Analyse des risques

3. Cas d'utilisation

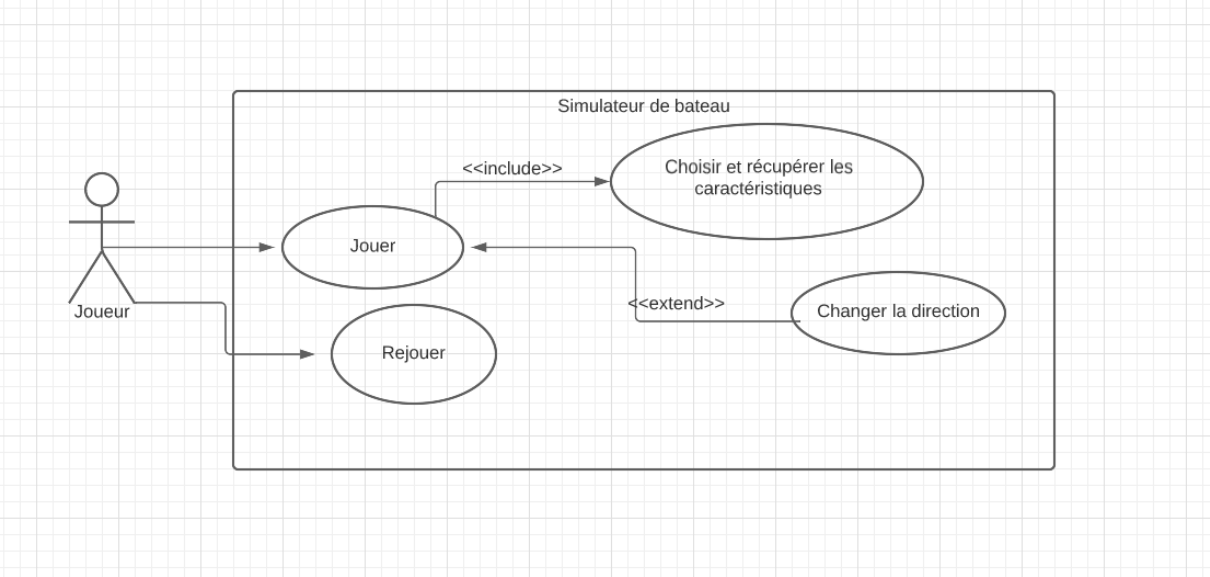

Figure 1 - Diagramme des cas d'utilisation

4. Diagramme de paquet

L'architecture globale du logiciel implémente le modèle MVP (Modèle-Vue-Présentation). Le modèle contient la logique de modélisation. La vue ne gère que l'affichage et ne possède aucune logique. La présentation contient la logique de présentation et fait le lien entre le

modèle et la vue. Nous avons donc décidé de créer trois paquets, un pour le modèle, un pour la présentation et un pour la vue comme ci-dessous.

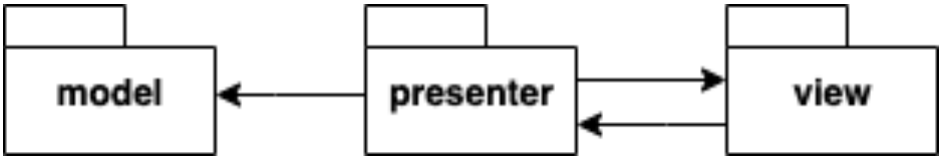

Figure 2 - Diagramme de paquet

5. Conception UML

5.1. Modèle

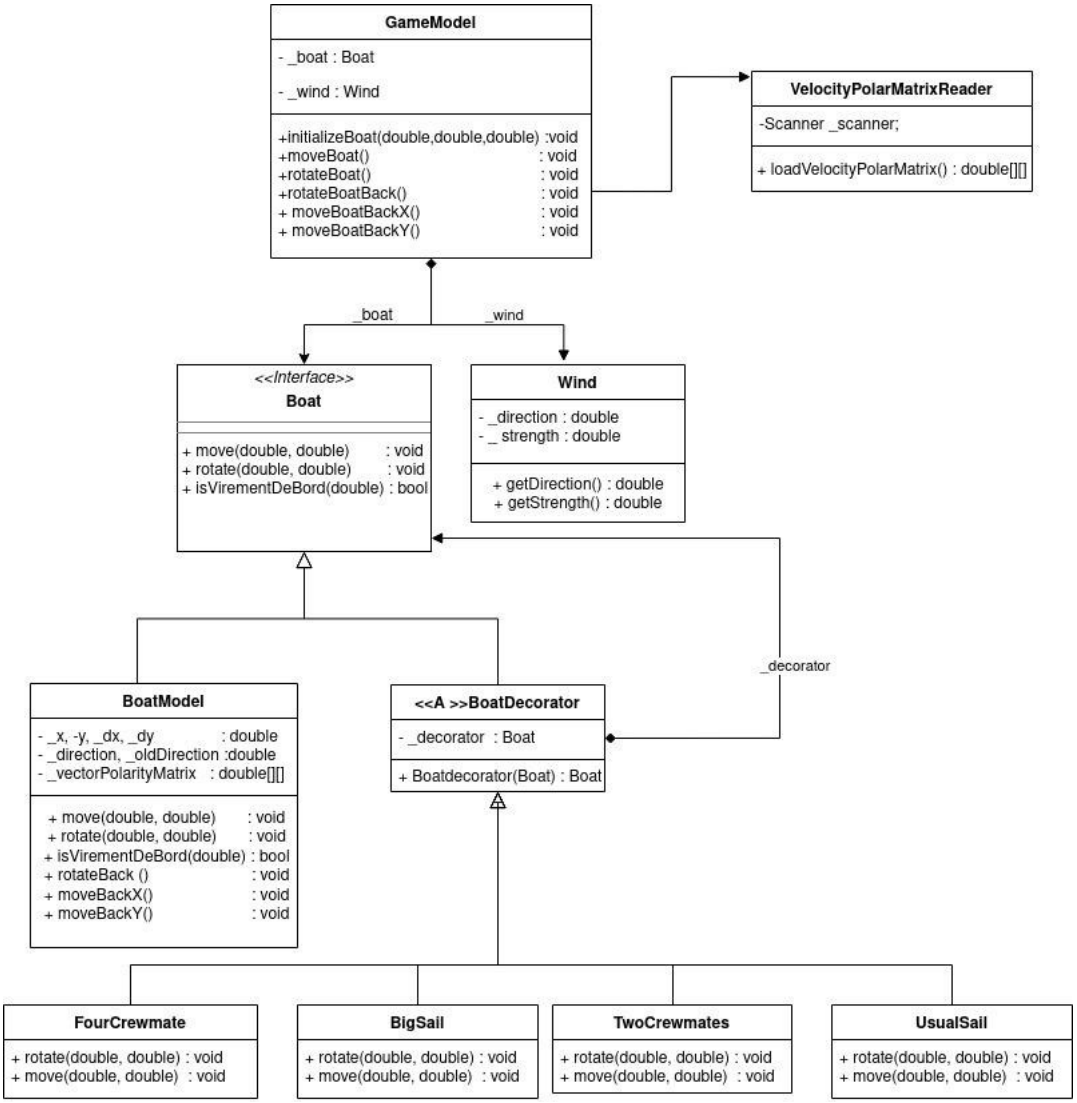

Figure 3 – Diagramme UML du modèle

Le modèle est composé de la classe **GameModel** qui gère l'ensemble du modèle. Elle contient les éléments du modèle comme els classe **Boat** et **Wind**. Elle utilise aussi la classe **VelocityPolarMatrixReader** qui permet de lire un fichier contenant un tableaux de polaire de

vitesses pour en extraire les données et de créer une matrice. On remarque l'implémentation du patron Decorateur avec la classe **BoatDecorator** et les classes qui l'implémentent. Les classes **FourCrewmate**, **TwoCrewmates**, **BigSail**, **UsualSail** décorent les fonctions **rotate** et **move** afin de modifier le compotement du bateau en fonction de la configuration choisie par le joueur.

.

5.2. Présentation

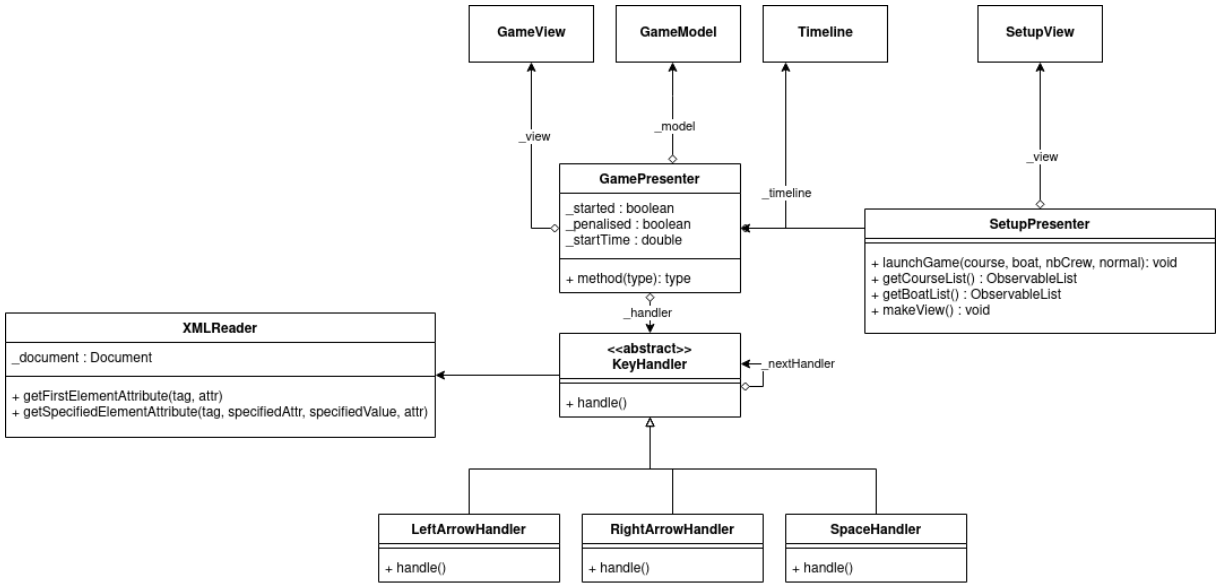

Figure 4 - Diagramme UML de la présentation

La présentation est composé de deux classes principales, **SetupPrensenter** et **GamePresenter**. La première gere la présentation lors de la phase de configuration en lien avec la vue. De plus elle appelle la seconde quand le jeu est lancé. Cette dernière utilise la classe **XMLReader** qui permet d'importer la course et est en lien avec le model et la vue. On remarque l'implémentation du patron Chaine de responsabilité avec la classe **KeyHandler** et ses dérivées. Ces classes gèrent quand le joueur souhaite changer de direction avec les touches du clavier. Chaque classe dérivée, **LeftArrowHandler**, **RightArrowHandler** et **SpaceHandler**, gère son cas dédié.

5.3. Vue

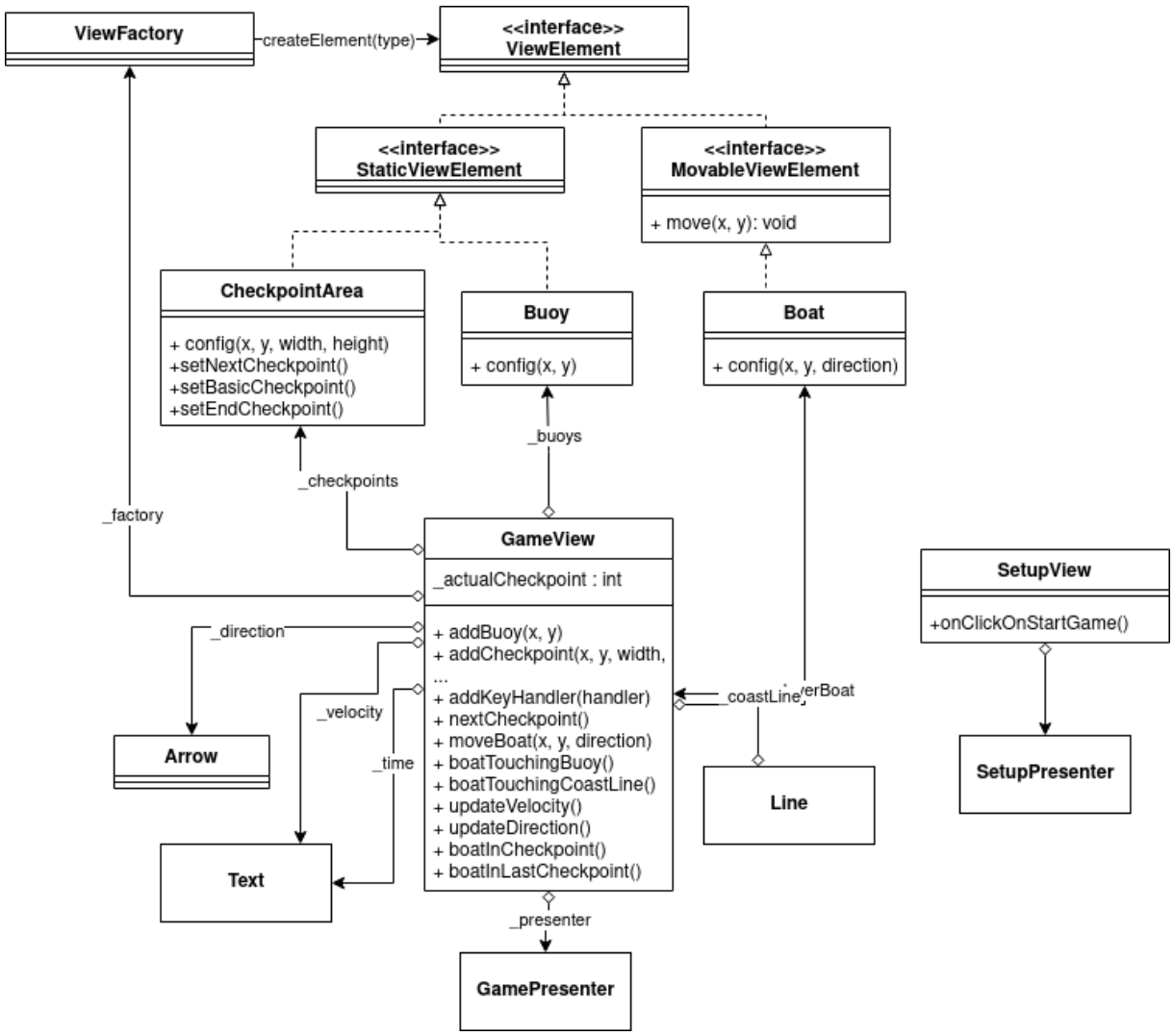

Figure 5 - Diagramme UML de la vue

La vue gère l'affichage. La classe **SetupView** concerne la fenêtre de configuration. **GameView** gère la fenêtre de jeu, ses fonctions permettent d'ajouter ou des mettre à jour les élements de la vue. On remarque l'implémentation du patron Factory avec la classe **ViewFactory** qui permet de créer des éléments **ViewElement** comme les bateaux (**Boat**), les bouées (**Buoy**) ou les zones de checkpoint (**CheckPointArea**). La fonction **config()** permet de mettre à jour un élément de la vue.

6. Configuration

Pour la configuration de la course, elle se fait via un fichier xml qui contient toutes les informations de la courses comme la position de départ, celles des bouées, des checkpoints, ainsi que la direction et la vitesse du vent. Pour le choix du bateau, il n'influt que sur la polaire de vitesse du bateau. Les différentes configurations des bateaux sont stockés dans des fichiers qui contiennent des tableaux de polaire de vitesse.

7. Résultat

7.1. Configuration

Au lancement du logiciel, l'utilisateur peut choisir la configuration du jeu qu'il souhaite. Il chosit la carte de la course, le bateau. Il a aussi le choix entre 2 et 4 equipiers et entre une voila de taille normale ou grande. On peut voir la fenêtre de configuration ci-dessous.

Figure 6 - Fenêtre de configuration

7.2. Jeu

En jeu, on voit le bateau modélisé par une ellipse, il doit passer dans les zones de couleur à côté des bouées jusqu'à la zone finale rouge. Sur le coin haut gauche, il y a les informations de direction et de force du vent ainsi que la direction et la vitesse du bateau. A droite se trouve la ligne modelisant un trait de côte ainsi qu'une échelle.

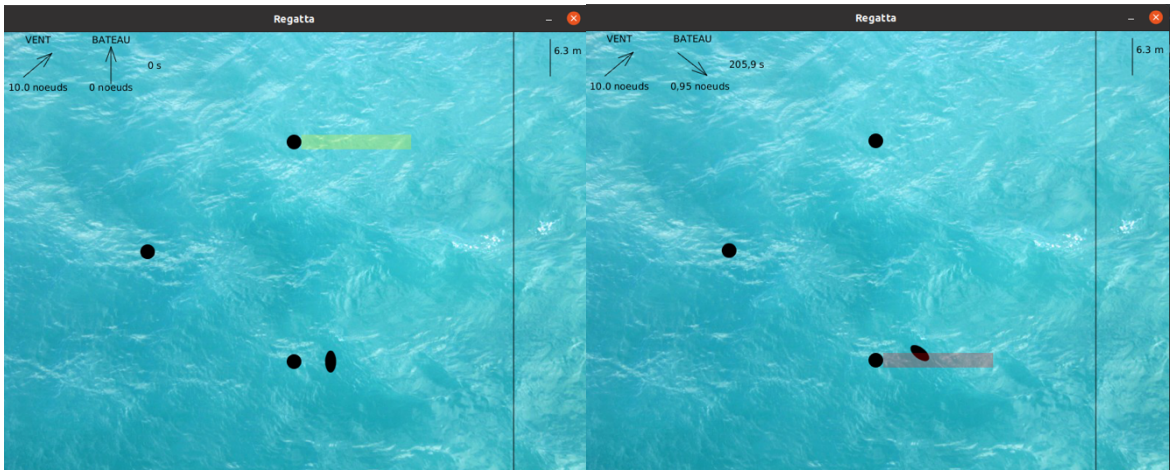

Figure 7 - Fenêtre de jeux (à gauche au départ, à droite à l'arrivée)

8. Conclusion

Ayant étalé en la réalisation du projet en 8 semaines, où le but n'était pas de travailler chez soi pour avancer, mais plutôt de se rassembler avec les membres de l'équipe dans des séances désignées pour le projet au dessein de prendre une idée sur le déroulement de travail pour un groupe d'ingénieurs devant achever un projet, conformément aux exigences de client et aux contraintes de temps. Nous avons réussi à implémenter plusieurs fonctionnalités (configuration, collisions, comportement du bateau en fonction de la configuration). Cependant, nous avons sans doute vu trop gros et nous n'avons pas eu le temps de mettre en place des tests unitaires car nous nous sommes concentrés sur les fonctionnalités à ajouter. De plus, notre projet n'est pas SOLID. Nous n'avons pas réussi à prendre de la hauteur sur notre projet afin qu'il le soit, car nous étions trop préocupés à implementer les fonctionnalités.

Ce projet aura donc réussi à accomplir la tâche qu'il devait réaliser : nous permettre de développer nos compétences en génie logiciel en appliquant les connaissances que l'on a acquises, que ça soit en cours de première ou deuxième année de génie logiciel. Il nous a permis d'apprendre certaines erreurs à ne pas faire pour pouvoir mieux appréhender nos futurs projets.

ENSI CAEN

ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Ecole Publique d'Ingénieurs en 3 ans

6 boulevard Maréchal Juin, CS 45053
14050 CAEN cedex 04

ministère éducation nationale enseignement supérieur recherche
Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

Normandie Université

Cti
Commission des titres d'ingénieur

cdefi
Conférence des Directeurs des Écoles Françaises d'Ingénieurs

European Accreditation of Engineering Programmes
EUR-ACE®

CONFÉRENCE DES GRANDES ÉCOLES