Bataille Navale

Projet Informatique IN104

Katleen Blanchet - katleen.blanchet@gmail.com

FIGURE 1 – Jeu de bataille navale

1 Présentation du sujet

Chaque binôme devra implémenter un jeu de bataille navale se jouant en réseau, au travers d'une architecture client/serveur. Deux modes de fonctionnement devront être disponibles : « Jouer contre un autre joueur » et « Jouer contre l'ordinateur ». Ce dernier mode s'appuiera sur une IA de votre choix. Enfin, pour rendre le jeu plus agréable, une interface graphique sera créée. Le projet sera développé en Python.

2 Objectifs

Au delà de l'automatisation d'un jeu de société classique, ce projet vise à vous initier :

- au mode client/serveur à l'aide d'une librairie facilitant les échanges réseaux (ZeroMQ [1])
- à l'IA
- aux interfaces graphiques (Tkinter [2])

Vous mettrez également en application vos connaissances en gestion de projet (planification des objectifs à atteindre pour chaque séance, validation par des tests unitaires).

3 Règles du jeu [3]

La bataille navale oppose deux joueurs. Au début du jeu, chaque joueur possède une flote de cinq bateaux, dont il choisit les emplacements sur une grille numérotée de 1 à 10 horizontalement et de A à J verticalement :

— porte-avion (5 cases)
— croiseur (4 cases)
— contre-torpilleur (3 cases)
— sous-marin (3 cases)
— torpilleur (2 cases)

Le but du jeu est de détruire la flotte ennemie. A tour de rôle, les joueurs envoient un missile sur la flotte du joueur adversaire. Si le missile manque sa cible et tombe à l'eau, son emplacement est marqué d'un symbole blanc. Sinon, si un bateau est touché, il est marqué d'un symbole rouge. Si toutes les cases occupées par un bateau sont touchées, on dit que le bateau est « coulé ». La partie se termine quand tous les bateaux d'un des deux joueurs sont détruits.

4 Contenu prévisionnel des TPs

Durant les TPs, nous consacrerons du temps à comprendre les nouvelles notions qui vous seront utiles pour réaliser votre projet, à savoir l'architecture client/serveur et l'interface graphique. Ils seront organisés comme suit (planning indicatif, sujet à modification suivant l'avancée des groupes à chaque TP) :

1. Création du dossier git et premières étapes de conception du jeu.
2. Fin de conception du jeu de base et prise en main de la librairie ZeroMQ pour réaliser le client/serveur.
3. Recherche et implémentation de l'IA.
4. Prise en main de Tkinter pour la création de l'interface graphique.
5. Amélioration du code, approfondissement de certaines parties, et affrontement entre IA en fin de séance pour les plus avancés.

5 Evaluation

Atteinte des objectifs Afin de valider le projet informatique, tous les objectifs décrits ci-dessus doivent être remplis. Il est attendu que chaque binôme ait utilisé les librairies ZeroMQ, Tkinter, ainsi qu'un algorithme d'IA. Une utilisation à minima est suffisante, le temps consacré au projet étant limité. Chaque binôme doit simplement être en mesure de montrer qu'il a compris et su appliquer les nouvelles techniques abordées lors des TPs. Le jeu de base doit, quant à lui, être fonctionnel.

Justification des choix Chaque choix d'implémentation doit être expliqué dans le rapport. Il doit également contenir les problèmes rencontrés et solutions apportées.

Robustesse et clarté du code Le code doit être lisible, commenté et s'exécuter facilement (pensez à écrire un fichier README). De nombreux tests unitaires sont également requis pour prouver la robustesse du code, notamment dans les cas limites de placement des bateaux.

Références

[1] Pieter Hintjens. Zeromq : The guide, 2010. `http://zguide.zeromq.org/page:all`.

[2] Tkinter, 2016. `https://wiki.python.org/moin/TkInter`.

[3] Wikipedia. Bataille navale (jeu), 2017. `https://fr.wikipedia.org/wiki/Bataille_navale_(jeu)`.