ECOLE DES MINES DE SAINT ETIENNE
CAMPUS G. CHARPAK PROVENCE

Rapport

Programmation système - Projet Application Client-Serveur

Gevorg ISHKHANYAN
Gaëtan Houllier

10 juin 2024

Table des matières

1 Introduction

Dans le cadre de ce projet, nous avons décidé d'implémenter un jeu de bataille navale. Le jeu de la bataille navale, appelé aussi communément touché coulé est un jeu de stratégie dans lequel deux joueurs s'affrontent pour couler les bateaux adverses. Il est apparu aux alentours de la première guerre mondiale et est constitué de deux phases. La première consiste pour chaque joueur à placer ses bateaux sur une grille de **taille 10 par 10** qui sera seulement visible par le joueur en question. Dans notre situation, nous nous contenterons de **3 bateaux**. Ensuite, chaque joueur donne à tour de rôle une case. Si sur cette case se trouve un bateau adverse, alors le bateau adverse est touché sinon le tir atterri dans l'eau. La partie se termine lorsqu'un des deux joueurs a perdu tous ses bateaux.

Ainsi dans ce projet, l'objectif sera de permettre à deux joueurs également appelé clients de se connecter sur un même réseau pour pouvoir s'affronter pour une partie de bataille navale. Dans ce projet, nous détaillerons par conséquent deux codes : le code **client.c** et le code **serveur.c** qui permettront d'organiser une partie de ce jeu iconique entre les deux joueurs. La figure 1 montre une illustration graphique d'un déroulement de partie et rappelle les règles qu'on a choisies d'adopter.

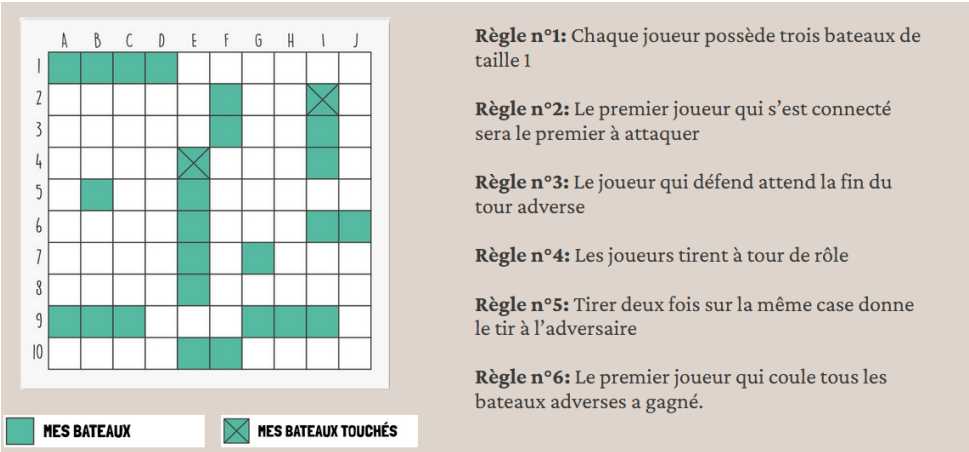

FIGURE 1 – Illustration graphique d'une grille de bataille navale avec les règles adoptées

2 Structure du projet

2.1 Le code serveur.c

Le code **serveur.c** permet d'organiser le déroulement de la partie. C'est lui qui en fonction de ce qu'il reçoit de la part des clients va choisir quel message affiché sur les terminaux de ces derniers. Il est constitué de 4 parties.
Ce diagramme décrit parfaitement les parties du code du serveur :

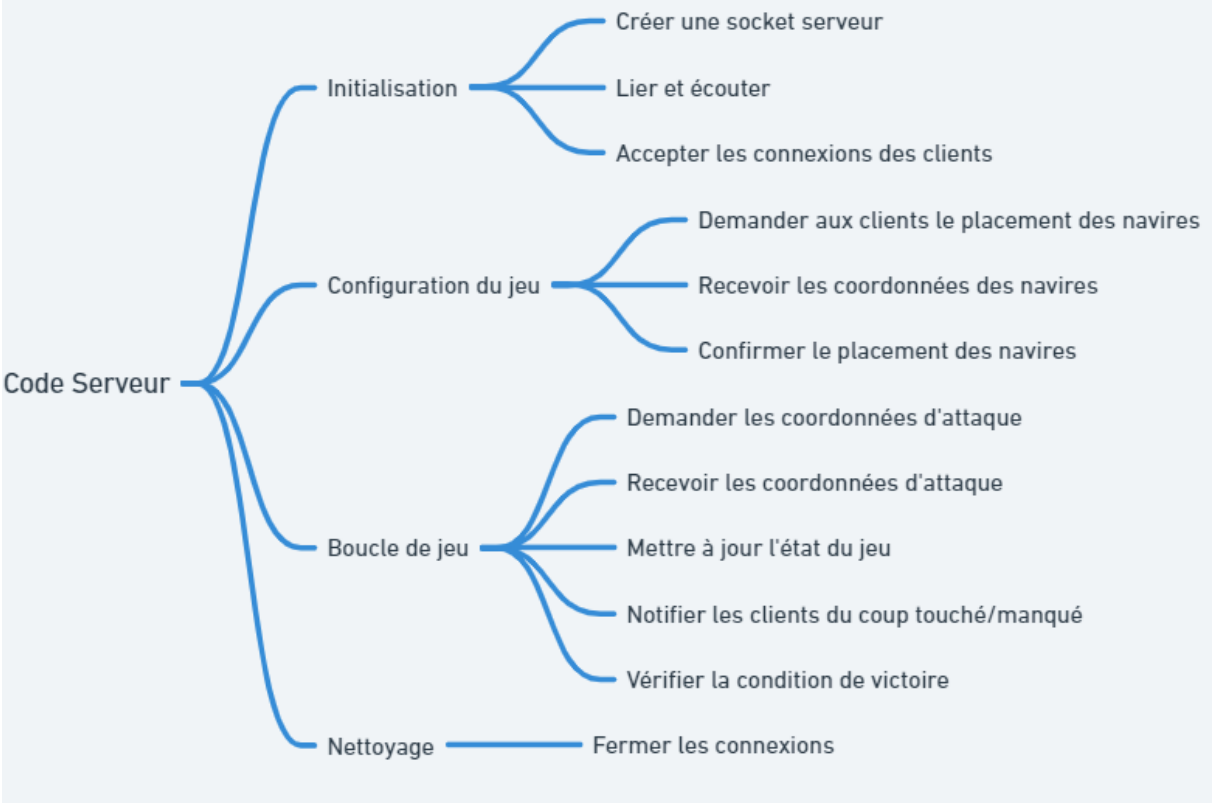

FIGURE 2 – Schéma du fonctionnement du code **serveur.c**

2.2 Le code client.c

Le code **client.c** permet quant-à lui de gérer toutes les communications entre le joueur et le serveur qui donne les directives. Il récupère les données saisies par le client sur sa machine, les envoies au serveur qui lui redonne des instructions. Il est constitué de trois parties qui représentent les 3 étapes :

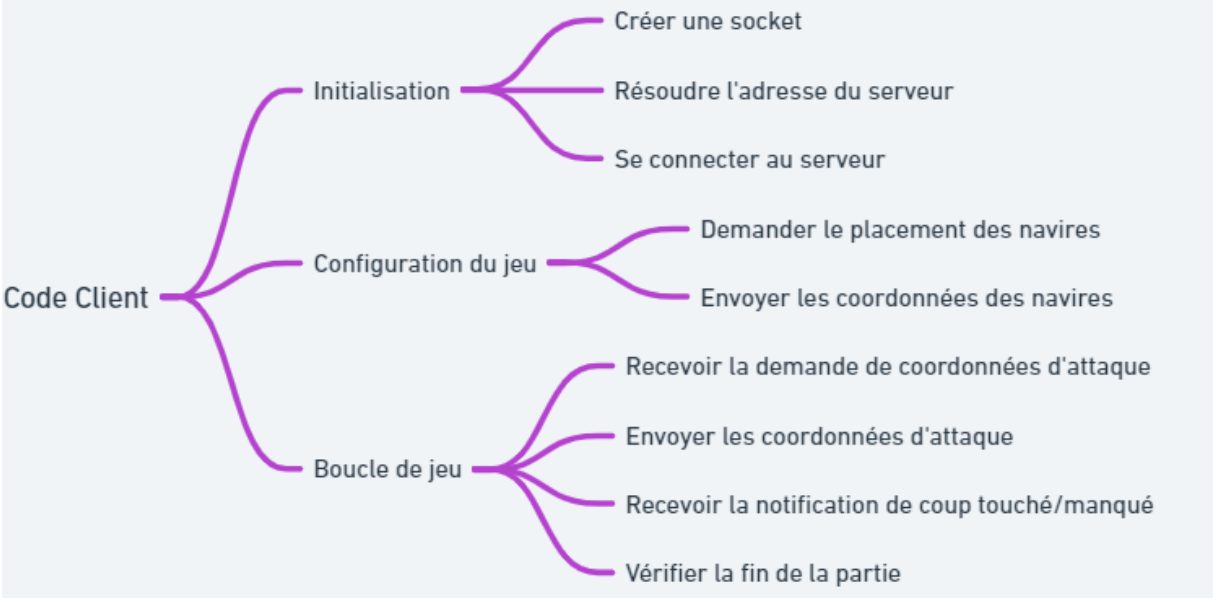

FIGURE 3 – Schéma du fonctionnement du code **client.c**

2.3 Architecture globale du projet

Avec un aperçu des deux codes **client.c** et **serveur.c**, il est maintenant possible de dresser une vision globale du projet. Il faut et il suffit de faire les bonnes connexions entre les deux codes pour que le serveur puisse à la fois donner les bonnes instructions à faire aux joueurs mais également que le client renseigne correctement ce qui lui est demandé. La figure 4 montre quand est-ce que le client et le serveur envoient et reçoivent des informations pour le bon déroulement du jeu.

FIGURE 4 – Diagramme de sequences du fonctionnement du code

Ainsi, après avoir compiler les deux codes et lancer les programmes, le client demande à se connecter au serveur qui l'accepte. Une fois que le joueur est accepté, le serveur lui requiert de placer ses bateaux. À cette demande, le code **client.c** lui renvoie les coordonnées choisies pour les bateaux qui seront par la suite acceptées par le serveur si elles sont valides. Une fois que les deux joueurs ont fini de placer leurs bateaux. Le serveur dirige les phases d'attaque et de défense des différents joueurs tout en les informant continuellement après chaque phase sur l'état de leurs navires ainsi que des navires adverses. Pour finir, lorsque les conditions de victoires sont vérifiées, le serveur annonce aux deux joueurs le gagnant.

3 Guide d'utilisation

Pour le bon fonctionnement du jeu, il faut d'abord compiler le code puis lancer sur le premier terminal le programme **serveur_bataille_navale** puis lancer sur deux autres terminaux le programme **client_bataille_navale**, pour cela suivre les indications suivantes :

- **POUR COMPILER :** Ouvrez un terminal dans le dossier Bataille_Navale, puis tapez '**make**' dans le terminal.
- **POUR LANCER LES PROGRAMMES :**
  - Ouvrez 3 terminaux dans le même dossier Bataille_Navale
  - Tapez dans le premier terminal la commande suivante :
    ./serveur_bataille_navale n (avec n le numéro de port)
    (Si une erreur est rencontrée, essayer un autre port.)
  - Puis dans les deux autres terminaux, tapez les commandes suivantes :
    ./client_bataille_navale localhost n (avec n le même numéro de port)
    (Si une erreur est rencontrée, essayer un autre port.)
  - PAR EXEMPLE : ./serveur_bataille_navale 2024 (dans le terminal 1)
    ./client_bataille_navale 2024 (dans les deux autres terminaux)

Attention : Les deux joueurs peuvent rentrer leurs bateaux en même temps cependant, pour éviter un bug, s'assurer que le joueur qui se connecte en second finisse de rentrer ses bateaux en second.

4 Conclusion

4.1 Difficultés rencontrées

Certaines parties de ce projet ont été synonymes de défi dont :

— La gestion de plusieurs connexions de clients car il faut assurer un jeu fluide dans différentes conditions de réseau. Une solution par exemple a été d'implémenter plusieurs threads pour permettre aux deux joueurs de remplir leur grille de bateaux et non d'attendre que le premier ait fini pour que le deuxième puisse commencer.

— Le débogage du notamment à l'usage multiple de la fonction bloquante read() qui est difficile à débugger.

4.2 Axes d'améliorations

Voici quelques pistes d'améliorations que nous suggérons pour venir peaufiner quelques détails :

— Améliorer l'interface utilisateur avec par exemple l'ajout d'une interface graphique permettant une meilleur immersion pour l'utilisateur.

— Développer une intelligence artificielle pour permettre à un joueur de jouer tout seul.

— Améliorer la sécurité en cryptant la communication entre le client et le serveur.

— Bloquer le terminal d'un joueur lorsque ce n'est pas à lui de jouer. Quand un joueur rentre des tirs alors que ce n'est pas son tour, les coups sont placés dans son "tampon" et sont choisis quand c'est à lui. Donc le joueur ne peut pas annuler son coup s'il change d'avis entre-temps.

— Vérifier que les bateaux saisis en amont ne se superposent pas.

4.3 Remerciements

Nous tenons à remercier notre professeur **Xavier SERPAGGI** pour ses cours intéressants, ainsi qu'à notre camarade **Bastien GARNIER** pour ses conseils au débug du code.