



Une école de l'IMT

ÉCOLE NATIONALE SUPÉRIEURE DES MINES DE
SAINT-ÉTIENNE

LATEXPROJECT
RAPPORT

RAPPORT DE PROJET - PROGRAMMATION SYSTÈME

Élèves :
Samy TALEB
Ylies VEYRAT

Encadrants :
SERPAGGI XAVIER

10 juin 2024

Table des matières

| | |
|---|----------|
| 1 Explication générale du projet | 2 |
| 1.1 Thème abordé | 2 |
| 1.2 Fonctionnement du jeu | 2 |
| 1.3 Interface | 2 |
| 1.4 Codage | 3 |
| 1.5 Gestion du serveur | 3 |
| 1.6 Gestion du client | 3 |
| 2 Explication et organisation des programmes | 5 |
| 2.1 Serveur | 5 |
| 2.2 Client | 5 |
| 3 Difficultés rencontrées et améliorations possibles | 6 |
| 4 Conclusion | 6 |

1 Explication générale du projet

Le projet consiste en la création d'un serveur de jeu d'échecs en réseau, permettant à deux joueurs de jouer une partie d'échecs à distance sur des terminaux différents, l'un contre l'autre, en se connectant à un serveur commun. Le jeu a été développé en utilisant le langage de programmation C sous système d'exploitation Linux et en utilisant les principes du cours de programmation système.

1.1 Thème abordé

Le thème principal abordé dans ce projet est la création d'un jeu d'échecs multi-joueurs en réseau. Les échecs sont un jeu de stratégie populaire connu de tous. Ils sont joués sur un échiquier de 8x8 cases numérotées en abscisses de A à H. L'échiquier est partagé par les deux joueurs, chacun disposant de 16 pièces (chaque pièce ayant des propriétés différentes). L'objectif du jeu est de mettre le roi de l'adversaire en position d'échec et mat, c'est à dire qu'aucun mouvement n'est possible pour sauver le roi des pièces adverses. Le jeu d'échecs est un sujet intéressant pour la création d'un jeu en réseau et nous avons choisi de le modéliser car il nécessite une logique de jeu complexe mais avec une interface utilisateur simple.

1.2 Fonctionnement du jeu

Le jeu d'échecs en réseau développé dans ce projet permet à deux joueurs de jouer une partie d'échecs en se connectant à un serveur commun. Chaque joueur se voit attribuer une couleur (blanc ou noir) et peut effectuer des mouvements sur l'échiquier à tour de rôle. La couleur est attribué au joueur au début de la partie et à chaque coup il est indiqué à qui est le tour de jour. Le serveur gère les connexions des joueurs, les mouvements des pièces et la mise à jour de l'état de l'échiquier pour chacun des joueurs à l'issue de chaque coup. Il est également mis à jour au sein de l'interface du serveur.

Au début de la partie, les joueurs se connectent au serveur en utilisant un terminal client. Une fois connectés, les joueurs sont mis en attente sur une liste d'attente jusqu'à ce qu'un autre joueur se connecte pour former une paire de joueurs. Lorsqu'une paire de joueurs est constituée, le serveur crée une nouvelle partie d'échecs et assigne les joueurs aux couleurs appropriées.

Pendant la partie, les joueurs peuvent effectuer des mouvements en saisissant les coordonnées de la case de départ et d'arrivée de la pièce à déplacer. Le serveur valide les mouvements proposés par les joueurs et met à jour l'état de l'échiquier en conséquence. Le serveur envoie ensuite l'état mis à jour de l'échiquier aux deux joueurs pour qu'ils puissent afficher l'échiquier mis à jour sur leur interface.

1.3 Interface

L'interface du jeu est en mode texte et utilise des caractères ASCII pour représenter les pièces et l'échiquier. L'échiquier est affiché avec les cases numérotées et lettrées, et les pièces sont représentées par des lettres majuscules pour les pièces blanches et minuscules

pour les pièces noires.

Les joueurs peuvent effectuer des mouvements en saisissant les coordonnées de la case de départ et d'arrivée de la pièce à déplacer. Par exemple, pour déplacer un pion de la case e2 à la case e4, le joueur saisit "e2e4". Le joueur peut également saisir la touche "Control"+ C pour abandonner la partie.

L'interface utilisateur est simple et facile à utiliser, ce qui permet à n'importe qui de pouvoir jouer sur le serveur.

1.4 Codage

Le jeu a été développé en utilisant le langage de programmation C, en s'appuyant sur les bibliothèques standard pour la gestion des entrées/sorties, des chaînes de caractères et des threads. La programmation réseau a été implémentée en utilisant les sockets qui permettent de créer des connexions réseau entre les clients et le serveur.

Le code source du projet se compose de deux fichiers principaux : un fichier serveur et un fichier client. Le fichier serveur contient le code pour la création et la gestion du serveur, tandis que le fichier client contient le code pour la connexion et l'interaction avec le serveur.

1.5 Gestion du serveur

Le serveur est conçu pour gérer plusieurs parties d'échecs simultanément, avec un maximum de 10 clients connectés en même temps. Le serveur utilise des threads pour gérer chaque partie d'échecs séparément, ce qui permet de maximiser l'efficacité du serveur.

Lorsqu'un client se connecte au serveur, il est ajouté à une liste d'attente jusqu'à ce qu'un autre client se connecte pour former une paire de joueurs. Une fois qu'une paire de joueurs est formée, le serveur crée une nouvelle partie d'échecs et assigne les joueurs aux couleurs appropriées.

Le serveur gère les mouvements des pièces en validant les mouvements proposés par les joueurs et en mettant à jour l'état de l'échiquier en conséquence. Le serveur utilise une fonction de validation de mouvement pour vérifier si un mouvement proposé est valide ou non. Si le mouvement est valide, le serveur met à jour l'état de l'échiquier et envoie l'état mis à jour aux deux joueurs. Si le mouvement est invalide, le serveur informe le joueur de l'erreur et demande un nouveau mouvement.

Le serveur gère également les déconnexions des joueurs en libérant les ressources allouées à la partie et en informant l'autre joueur de la déconnexion.

1.6 Gestion du client

Le client se connecte au serveur en utilisant une adresse IP. Une fois connecté, le client attend que le serveur lui attribue une couleur et une partie d'échecs. Le client affiche ensuite l'échiquier sur son interface et attend que l'utilisateur saisisse un mouvement.

Lorsqu'un mouvement est saisi, le client envoie les coordonnées du mouvement au serveur pour validation. Si le mouvement est valide, le serveur met à jour l'état de l'échiquier et envoie l'état mis à jour au client. Le client affiche ensuite l'échiquier mis à jour et attend le prochain mouvement de l'adversaire.

Le client gère également les déconnexions du serveur en informant l'utilisateur de la déconnexion et en libérant les ressources allouées à la partie.

2 Explication et organisation des programmes

Le projet de jeu d'échecs en réseau est composé de deux programmes principaux : le serveur et le client. Chaque programme a un rôle spécifique dans la gestion de la partie d'échecs et de la communication entre les joueurs.

2.1 Serveur

Le serveur est le programme principal du jeu d'échecs en réseau. Il est responsable de la gestion des parties d'échecs, de la communication avec les clients et de la validation des mouvements des pièces.

Le serveur utilise des sockets pour écouter les connexions entrantes des clients et pour communiquer avec eux. Les fonctions principales utilisées dans le fichier serveur sont les suivantes :

demarrer_serveur() : cette fonction est le point d'entrée du programme serveur. Elle initialise la socket d'écoute, lie la socket à une adresse et à un port, et écoute les connexions entrantes des clients.

gerer_partie() : cette fonction est le point d'entrée du thread de gestion de partie. Elle gère les mouvements des pièces, la mise à jour de l'état de l'échiquier et la communication avec les clients.

initialiser_echiquier() : cette fonction initialise l'état de l'échiquier avec les pièces et les pions aux positions de départ.

afficher_echiquier() : cette fonction affiche l'état de l'échiquier sur la console du serveur.

valider_mouvement() : cette fonction valide les mouvements proposés par les joueurs en fonction des règles du jeu d'échecs.

2.2 Client

Le client est le programme utilisé par les joueurs pour se connecter au serveur et pour jouer une partie d'échecs. Le client utilise des sockets pour communiquer avec le serveur et pour recevoir les mises à jour de l'état de l'échiquier. Les fonctions principales utilisées dans le fichier client sont les suivantes :

main() : cette fonction est le point d'entrée du programme client. Elle initialise la socket de connexion, se connecte au serveur, et gère la boucle principale de communication avec le serveur.

initialiser_echiquier() : cette fonction initialise l'état de l'échiquier avec les pièces et les pions aux positions de départ.

afficher_echiquier() : cette fonction affiche l'état de l'échiquier sur l'interface du client. Ces fonctions permettent de gérer efficacement la communication entre le serveur et les clients, ainsi que la gestion des parties d'échecs et la validation des mouvements des pièces.

3 Difficultés rencontrées et améliorations possibles

Lors du développement de notre projet de jeu d'échecs en réseau, nous avons rencontré plusieurs difficultés, notamment :

- La gestion de la communication entre le serveur et les clients : nous avons dû mettre en place des protocoles de communication pour envoyer et recevoir les données de manière fiable et efficace.
- La validation des mouvements des pièces : nous avons dû implémenter les règles du jeu d'échecs pour vérifier que les mouvements proposés par les joueurs étaient valides.
- La synchronisation de l'état de l'échiquier : nous avons dû nous assurer que les deux joueurs avaient la même vue de l'échiquier à tout moment, ce qui était particulièrement difficile lorsque les joueurs jouaient simultanément.

Pour améliorer notre projet, nous aurions pu synchroniser la mise à jour de l'échiquier noir directement après le coup joué par les blancs. Cela aurait permis d'éviter les problèmes de synchronisation de l'état de l'échiquier et d'améliorer l'expérience utilisateur.

4 Conclusion

Dans l'ensemble, nous avons pris beaucoup de plaisir à réaliser ce projet de jeu d'échecs en réseau. Il nous a permis de mettre en pratique nos compétences en C et en programmation système, tout en étant ludique et agréable à réaliser.

Nous avons appris beaucoup de choses sur la gestion de la communication entre les processus, la validation des données et la synchronisation de l'état de l'application. Ces compétences seront très utiles pour nos futurs projets de développement logiciel.

Nous sommes fiers du travail que nous avons accompli et nous pensons que le résultat final est un jeu d'échecs en réseau fonctionnel et amusant à jouer.



FIGURE 1 – Jeu d'échecs

FIN DU RAPPORT

Merci pour votre attention !

TALEB Samy, VEYRAT Ylies