Cnam FIP 2 – Systèmes avancés 2021-2022

Projet de programmation Sockets

Ce projet est à réaliser en **binôme** et la note sera intégrée à la note de l'examen final du module. Un petit rapport (cahier de programmation) de 10 à 15 pages devra être rendu avec les sources commentées de votre projet.

Ce projet consiste en la réalisation d'un jeu de *chasse au cavalier* à 2 joueurs en utilisant l'API Socket en langage C pour faire communiquer le programme de deux joueurs exécutés localement sur la même machine.

Règles du jeu : Le jeu se joue sur un échiquier de 8x8. Les deux cavaliers sont disposés initialement sur deux cases d'angles diagonalement opposées. Le cavalier blanc aura le rôle de poursuivant et le cavalier noir le rôle de poursuivi. L'objectif pour le cavalier blanc est d'atteindre la position du cavalier noir.
Les joueurs jouent à tour de rôle, et c'est le cavalier noir qui commence. Les cavaliers se déplacent comme au jeu d'échec vers une case libre. Après un déplacement, un pion rouge est déposé sur la case précédemment occupée par le cavalier lors du dernier déplacement. Ainsi au fur et à mesure de l'évolution de la partie, le nombre de cases libres diminue et le nombre de déplacements possibles se réduit.
La partie se termine dans l'une de ces trois situations :

1. Le cavalier blanc gagne si :
   - c'est à son tour de jouer et il peut atteindre la case du cavalier noir en un coup,
   - ou le cavalier noir ne peut plus se déplacer.
2. Le cavalier noir gagne si :
   - le cavalier noir est à l'abri derrière une barrière de pions rouge que le cavalier blanc ne peut franchir,
   - ou le cavalier blanc ne peut plus se déplacer.
3. Il y a égalité si les deux cavaliers sont dans l'impossibilité simultanée de se déplacer.

Architecture logicielle : L'approche la plus simple pour réaliser ce projet est de choisir une architecture Peer-to-Peer pour les échanges entre les deux joueurs. A savoir, le programme de chaque joueur aura à la fois une partie cliente et une partie serveur pour les échanges de messages liés aux coups des joueurs. Une seconde partie de ce projet consistera à introduire un serveur de jeu stockant les informations sur les joueurs (adresse et numéro de port d'un autre joueur), auquel chaque joueur pourra se connecter pour démarrer une nouvelle partie. Pour cette partie du projet, vous opterez pour une architecture Client/Serveur.
Les communications devront être effectuées en utilisant l'interface Sockets. Nous utiliserons le protocole de transport TCP afin de s'assurer de la bonne transmission et de l'ordre des messages.

Pour implémenter le projet, vous devrez utiliser le langage C et l'API socket.h. Si nécessaire, vous pouvez utiliser les tutoriels (le premier est synthétique et les deux autres sont plus complets) expliquant toutes les bases nécessaires pour la programmation en langage C :
http://franckh.developpez.com/articles/c-ansi/bien-debuter-en-c/
http://emmanuel-delahaye.developpez.com/tutoriels/c/initiation-pragmatique-c/
http://melem.developpez.com/tutoriels/langage-c/initiation-langage-c/

Vous serez amené à vous interfacer avec l'interface graphique fournie réalisée en C avec la librairie graphique GTK+3.0[1] et l'IDE Glade[2].

[1] https://developer.gnome.org/gtk3/stable/index.html
[2] https://glade.gnome.org

Cnam FIP 2 – Systèmes avancés 2021-2022

Le programme *cavalier_GUI.c* comprenant l'interface graphique GTK+ devra être compilé comme suit (exemple en ligne de commande avec gcc) :

```
gcc –Wall –o cavalier_GUI cavalier_GUI.c $(pkg-config --cflags --libs gtk+-3.0)
```

Chaque programme client pourra être lancé en passant en paramètre le numéro de port à utiliser :

```
./cavalier_GUI 55555 &
```

L'interface fournie est décomposée en 3 parties (comme schématisé dans la figure ci-après) :

- Cadre connexion au serveur : permet d'entrer les informations pour la connexion au serveur,
- Cadre joueurs : espace affichant les joueurs et permettant de se connecter à l'un d'entre eux,
- Plateau : affichant le plateau de jeu

Afin de simplifier le développement, nous considérerons que les programmes des deux joueurs s'exécuteront sur la même machine. Chacun utilisera l'adresse locale 127.0.0.1, mais un numéro de port différent pour les différencier (par exemple les ports 55 555, 55 556, …).
Le serveur aura la charge de maintenir la liste des joueurs connectés à lui et de leur envoyer une liste à jour, en cas de modification.
Vous êtes libre d'utiliser l'architecture serveur que vous désirez (itératif ou parallèle) en justifiant votre choix, cela sera pris en compte dans la note finale.

Vous aurez la charge de définir les types de messages nécessaires pour les différentes parties de l'application, ainsi que le format d'échange des données (taille des messages, type de données échangées ...). Un soin particulier devra être mis sur la gestion des erreurs retournées par toutes les différentes primitives utilisées, ainsi que sur la clarté et les commentaires de votre code.

Votre cahier de programmation devra comporter une présentation de vos choix concernant :

- les formats des messages,
- les algorithmes des processus composant le projet,
- la répartition du travail, les difficultés rencontrées,

et sera accompagné des codes sources commentés de votre projet.