Base de la Conception Logicielle
TicTacToe

DE MEIRA LIMA Otávio
CHAHIDI Hamza

1. Introduction

L'objectif du travail proposé était d'implémenter le jeu en ligne TicTacToe en langage Ada. La règle du jeu est simple : sur un plateau 3x3, deux joueurs "X" et "O" placent, un par un, leur pièce dans l'une des 9 positions du plateau, et il n'est pas possible de placer une pièce dans un poste différent, déjà occupé. Pour gagner la partie, le joueur doit compléter une séquence de trois pièces placées de manière adjacente, qu'elles soient en ligne, en colonne ou en diagonale. Il est possible que la partie se termine par un match nul s'il n'y a plus de positions vides sur le plateau et qu'aucun joueur n'a réussi à former une séquence de 3 pièces adjacentes.

Le jeu en ligne a été créé sur la base du protocole TCP. TCP (Transmission Control Protocol) et IP (Internet Protocol) sont des ensembles de protocoles qui visent à interconnecter les appareils les plus divers sur Internet. Autrement dit, ils spécifient comment la communication doit être effectuée au sein du réseau. L'IP définit « l'adresse » de la machine. Tout comme tout humain possède un document d'identification dans son pays, chaque machine possède également son identification IP au sein de son réseau actuel. Les deux protocoles sont fondamentaux pour le réseau Internet. Ils garantissent que les paquets d'informations arrivent à destination correctement et en toute sécurité. De cette façon, il est possible de traiter et de définir les données entre les appareils avant d'atteindre leurs destinations. Dans le cadre de ce travail, TCP garantit que l'autre joueur recevra le coup correctement.

2. Mise en œuvre
   2.1. Consignes d'exécution

Les étapes pour lancer le jeu sont décrites ci-dessous :

1. Le projet peut être compilé avec un chef de projet GPR. Dans le dossier principal du programme, exécutez la commande suivante :

gprbuild -P ./projet_tictactoe.gpr

2. Bon, maintenant le projet est compilé. Si l'exécution est réussie, deux fichiers exécutables seront créés dans le dossier "exec". Tout d'abord, exécutez le fichier contenant le code du serveur :

   ./exec/tictactoe_serveur.exe

3. Maintenant que le serveur est opérationnel, vous devez exécuter le fichier client. Comme il y aura deux joueurs, vous devez lancer deux fois la même commande dans deux terminaux différents :

   ./exec/tictactoe_client.exe

4. Tout est prêt pour commencer le jeu. Le premier joueur à se connecter a la pièce « X » et est toujours le premier à jouer. Exécutez à nouveau les mêmes commandes pour rejouer le jeu. Apprécier!

2.2. Détails du conseil

La représentation de ce jeu en langage Ada est assez simple. Créez simplement une matrice 3x3, imprimez un tableau avec les caractères "_" et "|". Un package supplémentaire a été créé, présent dans le dossier « lib », avec des fonctions auxiliaires de la carte. Le forfait s'appelle "Conseil_TicTacToe". Les fonctions dont il dispose sont les suivantes :

```
package Conseil_TicTacToe is

    type T_Conseil is array (1 .. 3, 1 .. 3) of Character;

    function Put_Piece (conseil : in out T_Conseil;
    move : T_Chaine; piece : Character) return Boolean;
    procedure Log_Conseil (conseil : T_Conseil);
    function Verify_Win (conseil : T_Conseil) return Boolean;

end Conseil_TicTacToe;
```

La fonction "Put_Piece" reçoit la position choisie par le joueur et la pièce du joueur. Il renvoie un booléen, vrai si la position est vide et faux si la position est occupée. Ce retour est important pour valider si la position insérée est valide. Si le poste est déjà occupé, le client doit fournir un autre poste. La méthode « Log_Conseil » imprime simplement le plateau sur le terminal avec les pièces positionnées aux bons endroits. La fonction "Verify_Win" vérifie si le jeu s'est terminé.

L'impression de la carte se fait à l'aide de la librairie ANSI_Console. Il a des méthodes d'impression qui aident beaucoup à la visibilité du tableau. Le tableau est imprimé à l'aide de caractères simples déjà présents sur le clavier. La fonction « Clear_Screen » élimine tous les caractères déjà imprimés sur le terminal. L'impression du bac suivant est si

rapide que vous ne pouvez même pas remarquer que le terminal a été effacé pendant quelques instants.

2.3. Détails du serveur

Le langage lui-même fournit des bibliothèques auxiliaires qui facilitent la construction de code pour effectuer la communication entre deux programmes. La bibliothèque principale s'appelle GNAT.Sockets. Il fournit des méthodes pour la création de socket, le processus de liaison, le type d'adresse et le port. Le code serveur utilise cette bibliothèque pour ouvrir des ports et recevoir de nouvelles connexions.

Le serveur doit d'abord créer un « socket ». Les sockets sont un moyen de permettre la communication interprocessus entre des programmes s'exécutant sur un serveur ou entre des programmes s'exécutant sur des serveurs distincts. La communication entre les serveurs repose sur des sockets réseau, qui utilisent le protocole Internet (IP) pour encapsuler et gérer l'envoi et la réception de données. Ensuite, l'adresse de la machine et le port pour recevoir les communications sont créés. Ensuite, cet ensemble adresse+port est lié au socket qui vient d'être créé. Avec ces étapes simples, le client est maintenant libre de se connecter au serveur.

Dans le cadre du jeu TicTacToe, seuls 2 clients sont nécessaires pour jouer. Le serveur attribue la tuile « X » au premier joueur qui se connecte et la tuile « O » au deuxième joueur. Après avoir attribué les pièces, les joueurs sont libres de recevoir et d'envoyer des coups en respectant l'ordre du jeu. Le serveur reçoit une position sous forme de chaîne et transmet les informations au client. Le message reçu du client "X" n'est envoyé qu'au client "O" et vice versa.

Par conséquent, la responsabilité du serveur est uniquement de recevoir le message et de le transmettre au client.

2.4. Détails du client

Le client, comme le serveur, doit également créer le socket. Mais cette fois, vous devez ajouter l'adresse et le port du serveur auquel vous essayez de vous connecter. C'est pourquoi il est essentiel que le serveur s'exécute en premier, de sorte que lorsque le client essaie de se connecter, le serveur existe déjà et ne provoque pas d'erreurs d'exécution. Dès qu'il se connecte, le client sait quelle partie est "X" ou "O". Cette information est extrêmement importante, étant donné que le comportement des clients est différent pour chaque joueur :

- Si le client a été le premier à se connecter, il doit être le premier à jouer. La séquence de code à exécuter est la suivante :
    - Tapez dans le terminal à quelle position vous voulez jouer.
    - Attendez que le joueur suivant soumette son coup.

La séquence de code s'exécute en boucle jusqu'à ce que l'un des joueurs soit le gagnant ou qu'une égalité se produise.

- Si le client était le deuxième à se connecter, il doit d'abord attendre que le premier joueur joue :
    - Attendez que le joueur suivant soumette son coup.
    - Tapez dans le terminal à quelle position vous voulez jouer.

  La séquence de code s'exécute en boucle jusqu'à ce que l'un des joueurs soit le gagnant ou qu'une égalité se produise.

Il est possible de remarquer la différence entre les deux clients, puisque le jeu ne reçoit pas de coups simultanément des joueurs. Pendant que le joueur « X » écrit son coup dans le terminal, le joueur « O » doit attendre. Une fois le déménagement envoyé au serveur, le comportement des clients est inversé.

3. Conclusion

Il était clair que l'objectif du travail était de travailler la programmation de manière pratique et intéressante à travers la mise en place d'un jeu en ligne en langage Ada. Les concepts utilisés dans ce travail, tels que la généralité, l'encapsulation et les protocoles Internet, ont été explorés à travers l'utilisation de bibliothèques existantes qui aident au développement de telles caractéristiques. Le langage Ada s'est avéré être un obstacle majeur, car il n'y a pas de flexibilité avec de simples erreurs, telles que la stylisation, les espaces supplémentaires entre les lignes, la dénomination, etc. Il est entendu que ces restrictions visent à rendre le langage plus rigide, laissant peu de place aux erreurs courantes, mais cela a été l'un des principaux obstacles lors de la mise en œuvre. Cependant, il a également fourni une aide précieuse avec les bibliothèques existantes qui aident à implémenter divers aspects du programme, tels que les sockets, le texte et la console.