Protocole réseau – Bataille Navale Multijoueurs

Le protocole utilise 2 parties pour ainsi dire distinctes : la connexion et le jeu lui-même.
Dans la suite le terme « joueur » sera utilisé dans le même sens que « machine d'un joueur » ou « programme permettant de jouer ».

1) Connexion

Cette partie utilise principalement une socket **UDP**. Ceci permet en effet d'écouter sur un seul point de connexion (socket) pour recevoir les messages de tous les joueurs. La connexion se fera sur un port aléatoire de façon à pouvoir lancer plusieurs joueurs sur la même machine.

* Pour rejoindre une partie, un joueur doit envoyer la chaîne « ***HELLO_*** » (4 lettres plus un espace) suivi du nom du joueur, sur au plus 20 caractères. Les caractères non utiles seront remplacés par le caractère nul ('\0'). Le message de connexion fera donc toujours exactement 25 caractères ASCII.
* La machine cible répondra par la liste des joueurs de la partie : « ***JOUEURS*** » suivi du nombre de joueurs codé en binaire naturel (non signé) sur 8 bits (type C : *unsigned char*). Viennent ensuite autant d'enregistrements que de joueurs :
    - *Adresse IP* sur 4 octets
    - *Port* sur 2 octets
    - Nom du joueur sur 20 octets ASCII. (voir ci-dessus)

  Le message de réponse fera donc **7+n*26 octets**, n étant le nombre courant de joueurs (nouveau joueur inclus). On admettra une limite de **20 joueurs maximum**.
* La machine cible enverra en outre le même message **à tous les joueurs**, de façon à les prévenir de l'arrivée de ce nouvel élément.

2) Début du jeu

On utilisera le protocole TCP (sur le même port que précédemment, en UDP), de façon à garantir l'arrivée des messages, et surtout à détecter la disparition inopinée d'un joueur. On activera pour cela l'option Keep-Alive qui envoie un *ping* si la connexion reste inactive trop longtemps :

- **setsockopt**(s, SOL_SOCKET, SO_KEEPALIVE, &optval, optlen) avec optval un **int = 1** (pour activer l'option)
- **setsockopt**(s, SOL_TCP, TCP_KEEPIDLE, &optval, optlen) avec optval un **int = 15** (pour 15 secondes d'inactivité)
- **setsockopt**(s, SOL_TCP, TCP_KEEPINTVL, &optval, optlen) avec optval un **int = 3** (pour 3 secondes entre 2 *pings* sans réponse)

La connexion TCP étant orientée client/serveur on utilisera la convention suivante :
- la connexion TCP est ouverte par le joueur ayant le **port le plus petit**. En cas d'égalité, on comparera les adresses IP.
- Le joueur ayant le port le plus grand sera donc le serveur de cette connexion.

Afin de garantir que la partie continue si un joueur quitte la partie, il n'y a PAS de serveur central. Autrement dit, il y aura une connexion TCP entre chaque paire de joueurs.

On considèrera un délai maximal de début de partie de 60 secondes (1 minute) à partir de l'arrivée du second joueur.

Pour cela, on activera en outre l'option Timeout sur la socket serveur TCP de façon à ce qu'un joueur échouant à rentrer dans la partie ne bloque pas les autres :
- **setsockopt**(s, SOL_SOCKET, SO_RCVTIMEO, &optval, optlen) avec optval un **struct timeval = 60 secondes**

Exemple : dans le cas d'une partie à 3 joueurs, de ports 1000, 2000 et 3000 :
- connexion TCP de 1000 à 2000
- connexion TCP de 1000 à 3000
- connexion TCP de 2000 à 3000

Le port 2000 sera donc client du port 3000 et serveur du port 1000. S'il disparaît, le joueur 1000 peut finir la partie avec le joueur 3000.

* Pour **débuter la partie**, chaque joueur se connecte à ses serveurs en TCP.
* Chaque joueur envoie la chaîne « ***PRET*** ». (4 octets).

Une fois que tous les joueurs ont reçu tous les « *PRET* », ou que les 60 secondes sont écoulées, la partie débute, et les sockets TCP serveur et UDP peuvent être fermées. Aucun autre joueur ne peut alors rejoindre la partie.

3) Déroulement d'un tour de jeu

Un tour de jeu correspond au **tir simultané** de chacune des salves de chaque joueur. Pour cela, chaque joueur envoie chacun de ses tirs via TCP, puis signale la fin de son tir :
* Envoi d'un tir à un joueur : 3 lettres sont envoyées au port TCP de ce joueur : « ***T[A-J][0-9]*** » : 3 caractères ASCII où la lettre de A à J représente la colonne, le chiffre de 0 à 9 représente la ligne.
* Signale la fin du tir : envoi le caractère '*' à tous les joueurs.

Par exemple, dans un jeu à deux joueurs, l'envoi de la chaîne « *TA0TH4** » tire deux salves en A-0 et en H-4 avant de terminer le tir.

Lorsque tous les joueurs ont terminé, chaque joueur indique **à tous les autres** le résultat des tirs qu'il a reçus :
- Tir dans l'eau : « ***[A-J][0-9]-*** » : 3 caractères ASCII.
- Tir au but : « ***[A-J][0-9]+*** » : 3 caractères ASCII.
- Tir critique : « ***[A-J][0-9]*** » : 2 caractères ASCII suivi d'un chiffre « ***[1-5]*** » désignant le navire coulé par ce tir (1 :Porte-avions, Croiseur, Contre-Torpilleur, Sous-Marin, 5 :Torpilleur).

A la fin du tour chaque joueur annonce **à tous les autres** le nombre de ses navires restant (et donc le nombre de tirs qu'il lui reste au prochain tour) :
- « ***RESTE[0-5]*** » : 6 caractères ASCII. Par exemple : « RESTE0 » indique un joueur perdant.

4) Fin du jeu

Le jeu se termine lorsque au plus un joueur possède au moins un navire non détruit.
A ce moment, le joueur gagnant envoie la chaine "***GAGNANT_***" : 7 caractères suivis d'un espace puis du nom du joueur sur 20 caractères (voir ci-dessus). Le joueur gagnant a alors la possibilité d'envoyer un message de taille quelconque aux autres joueurs, puis ferme la connexion, ce qui termine le message.

A ce moment, tous les joueurs affichent le message, ferment leurs connexions et se terminent.

*) A tout moment

Un joueur qui se déconnecte est réputé perdant. Il ne peut rejoindre la partie une fois commencée.