Master 1 Informatique - Réseaux et Communications - TD/TP 4
Communications client-serveur et serveurs multitâches

Nous avons vu en cours comment mettre en œuvre des communications en réseau entre un processus client et un processus serveur suivant différents modes et différents protocoles. L'objectif de l'exercice suivant est de réaliser une simple application en utilisant le mode connecté et le protocole TCP.

Dans la suite, les classes Sock et SockDist, téléchargeables depuis la page du cours, peuvent être utilisées.

Dans un premier temps, nous supposons un seul client. Ce client, envoie au serveur une suite de messages saisis au clavier. Ces messages sont lus et affichés par le serveur.

1. décrire le schéma algorithmique permettant l'établissement d'une connexion entre le client et le serveur ainsi que le dialogue.
2. Écrire le programme correspondant en utilisant les opérations suivantes :

   char *fgets (char *s, int size, FILE *stream) : lit au plus size - 1 caractères depuis stream et les place dans le buffer pointé par s. La lecture s'arrête après EOF ou un retour-chariot. Si un retour-chariot (newline) est lu, il est placé dans le buffer. Un caractère nul '\0' est placé à la fin de la ligne.

   ssize_t write(int fd, const void *buf, size_t count) : écrit jusqu'à count octets dans le fichier associé au descripteur fd depuis le buffer pointé par buf. Elle renvoie le nombre d'octets écrits (0 signifiant aucune écriture), ou -1 en cas d'échec, auquel cas errno contient le code d'erreur.
   ssize_t read(int fd, void *buf, size_t count) : lit jusqu'à count octets depuis le descripteur de fichier fd dans le buffer pointé par buf. Elle renvoie -1 si elle échoue, auquel cas errno contient le code d'erreur, et la position de la tête de lecture est indéfinie.

Dans un deuxième temps, nous supposons deux clients qui se parlent à travers le serveur. Cela consiste à créer un serveur qui attend que deux clients se connectent puis qui transmet à chacun ce que l'autre envoie.

- Ecrire un serveur qui attend que deux clients se connectent et retransmet au client 2 ce que le client 1 envoie et vice-versa. **Note :** Il est demandé d'analyser différentes mises en œuvre du serveur.
- Ecrire le code des clients.

Enfin, nous supposons que le serveur ne transmet pas tous les messages reçus depuis un client vers le deuxième client. Sous la demande d'un client, un message est soit transmis soit simplement affiché sur le serveur.

- Modifier votre programme pour prendre en compte ce comportement.