Faculté des Sciences & Techniques — Université de Limoges

Master 1ère année

Réseaux & Système

TD n°2

Création de processus & Programmation réseau

Chat

1 – Écrire un programme **serveur** (utilisant `accept`) :
- ▷ attendant sur le port 8080 ;
- ▷ affichant sur la sortie standard (l'écran) les **lignes** qu'il reçoit depuis le client qui lui est connecté.

Optimisation

Lorsque votre programme se termine, il restitue le numéro de port qu'il utilisait.
Néanmoins, pour des raisons de protection (éviter que des anciens messages en cours de transit n'arrivent sur un port qui a été redonné à un nouveau programme), un **temps d'attente assez long** est mis en place par le système d'exploitation.
Pour pouvoir **récupérer immédiatement** l'usage du même numéro de port dans la phase de développement du programme serveur, il est possible de désactiver cette temporisation en configurant la socket :

```
ma_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
```

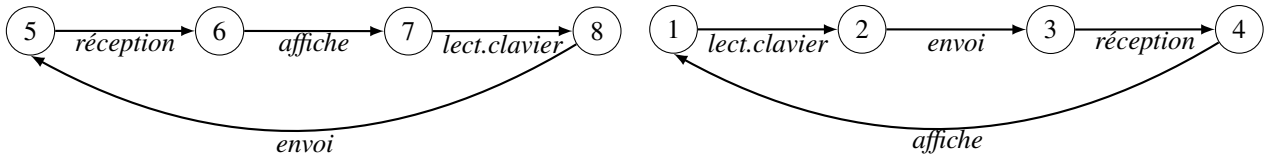
Vous pourrez tester votre programme avec l'outil socat (voir fin de la fiche pour la documentation).

2 – Écrire un programme **client** (utilisant `connect`) envoyant tout ce que l'on tape au clavier vers le serveur (envoi de lignes). *Vous testerez ce programme avec celui de la question 1.*

3 – Écrire deux programmes, l'un client et l'autre serveur, permettant le dialogue entre deux utilisateurs par alternance (« chat » élémentaire).
L'échange se déroule de la manière suivante :

1) *Le client lit une ligne au clavier*
2) Le client transmet une ligne au serveur
3) Le client attend une ligne du serveur
4) *Le client affiche la ligne reçue à l'écran*
5) Le serveur attend une ligne du client
6) *Le serveur affiche la ligne reçue à l'écran*
7) *Le serveur lit une ligne au clavier*
8) Le serveur envoie une ligne au client

Chaque programme réalise une boucle sans fin en « cyclant » entre les différentes lignes de l'algorithme :



Que peut-il arriver si on commence **différemment** le travail du client et du serveur ?

Utilisation de processus exécutés de manière concurrente

4 – Sous Python, il est possible d'obtenir un nouveau processus à partir du processus courant à l'aide de l'instruction « `fork` ».

Les deux processus obtenus après l'exécution de l'instruction fork, sont exécutés de manière concurrente et partage l'ensemble des entrées/sorties du processus initial, **ainsi que les sockets**, ce qui permettra à chacun de ces processus de s'occuper **d'un sens de la communication**.

Question :
Améliorer les programmes de client et de serveur des questions 2 & 3 de manière à ce que chaque interlocuteur puisse transmettre et recevoir une ligne de données à l'autre dès que possible.