

SIF1015 - Devoir #2

Manuel utilisateur

Projet réalisé par Julio Bangadebia et Quentin Anière

Description

Ce TP est la suite logique du TP2.

Le changement majeur est qu'au lieu d'envoyer une série de commandes au serveur, on envoie opération par opération. D'autres optimisations ont été faites pour améliorer la performance du serveur.

Le principe reste le même : Le serveur gère de façon concurrente une liste de VMS, sur laquelle les clients peuvent effectuer des opérations. Les opérations sont induites par des sockets. Les opérations sont traitées par une FIFO concurrente, First In, First Out, la première opération reçue sera la première opération traitée.

Compilation

Serveur

Avec makefile

```
make server
```

Sans makefile

```
gcc -o server server.c
```

Client

Avec makefile

```
make client
```

Sans makefile

```
gcc -o client client.c
```

Utilisation

Serveur

Avec makefile

```
make launch-server
```

Sans makefile

```
./server
```

Client

Avec makefile

```
make launch-client
```

Sans makefile

```
./client
```

Options

Pour choisir le port (client et serveur) :

Avec makefile

```
make launch-< server | client > port=< port >
```

Sans makefile

```
./< server | client > < port >
```

Opérations possibles

- A - Ajouter une VMS
- L x - y - Lister les VMS avec un identifiant compris entre x et y
- E x - Éliminer une VMS avec un identifiant x
- X x y - Exécuter le code binaire compris dans le fichier y sur la VMS x

Fonctionnement

Serveur

- Le serveur est lancé (via les commandes ci-dessus) sur un port
- Il attend une connexion d'un client
- Une fois le client connecté, le serveur lance un thread pour gérer les opérations du client
- Le serveur attend une commande du client
- Le serveur exécute la commande du client

- Le serveur envoie le résultat de l'opération au client
- Le serveur attend une nouvelle commande du client
- Si le client se déconnecte, il le notifie dans la console, ferme le socket de communication et tue le thread

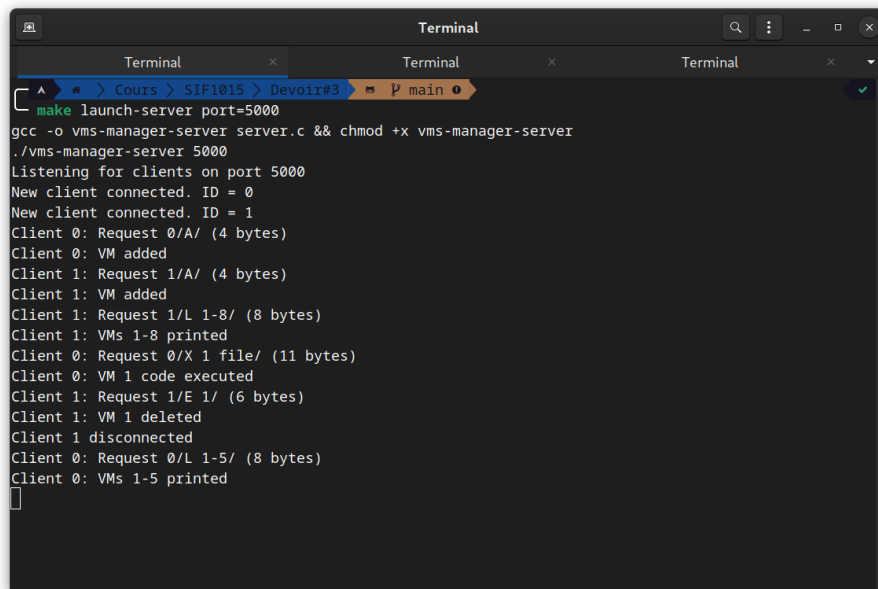
Les opérations sont traitées par une FIFO concurrente :

- Toute opération est ajoutée à la FIFO
- Le client lit la première opération de la FIFO
- Le serveur l'exécute et retourne le résultat au client qui l'a ajouté à la FIFO uniquement

Client

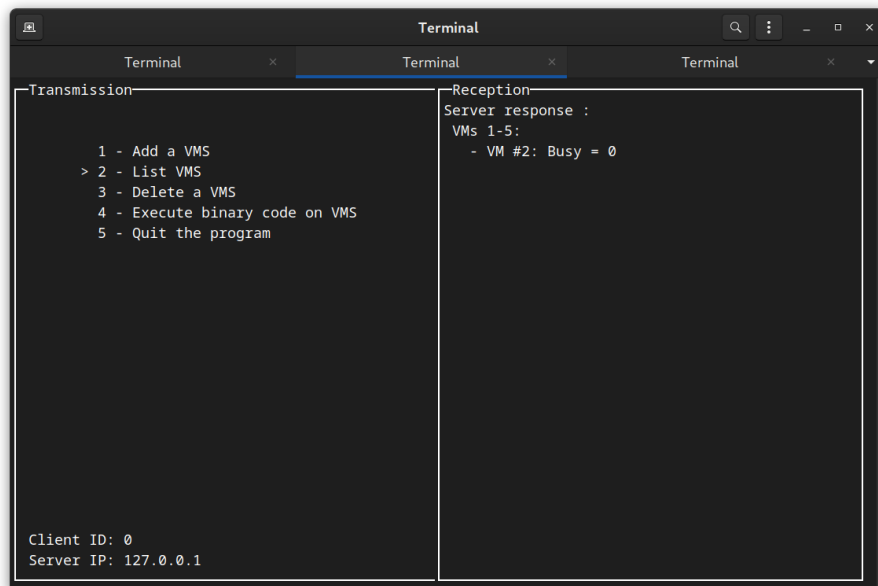
- Le client est lancé (via les commandes ci-dessus) en spécifiant le port du serveur et l'adresse IP du serveur
- Le client se connecte au serveur via le port et l'adresse IP spécifiés
- Il affiche deux fenêtres :
 - Une fenêtre pour saisir les commandes
 - Une fenêtre pour afficher les résultats
 - Un thread est lancé pour lire les résultats du serveur, et les afficher dans la fenêtre de résultats
- Dès que l'utilisateur sélectionne une opération, le client envoie la commande au serveur

Captures d'écran

A screenshot of a terminal window titled "Terminal" with three tabs. The active tab shows the following text:

```
^ * > Cours > SIF1015 > Devoir#3 * P main o
make launch-server port=5000
gcc -o vms-manager-server server.c && chmod +x vms-manager-server
./vms-manager-server 5000
Listening for clients on port 5000
New client connected. ID = 0
New client connected. ID = 1
Client 0: Request 0/A/ (4 bytes)
Client 0: VM added
Client 1: Request 1/A/ (4 bytes)
Client 1: VM added
Client 1: Request 1/L 1-8/ (8 bytes)
Client 1: VMs 1-8 printed
Client 0: Request 0/X 1 file/ (11 bytes)
Client 0: VM 1 code executed
Client 1: Request 1/E 1/ (6 bytes)
Client 1: VM 1 deleted
Client 1 disconnected
Client 0: Request 0/L 1-5/ (8 bytes)
Client 0: VMs 1-5 printed
```

Les logs du serveur sont affichés dans la console. On voit que deux clients étaient connectés et que les deux ont effectué différentes opérations. On voit également que le client 1 s'est déconnecté. On peut aussi constater que j'ai lancé le serveur sur le port 5000.



The image shows a terminal window with three tabs. The active tab is the middle one, which is split into two panes. The left pane, titled 'Transmission', contains a menu of options: '1 - Add a VMS', '> 2 - List VMS', '3 - Delete a VMS', '4 - Execute binary code on VMS', and '5 - Quit the program'. At the bottom of this pane, it shows 'Client ID: 0' and 'Server IP: 127.0.0.1'. The right pane, titled 'Reception', shows the server's response: 'Server response :', 'VMS 1-5:', and '- VM #2: Busy = 0'. The other two tabs are also titled 'Terminal' but are not active.

```
Terminal
Terminal
Terminal

Transmission
1 - Add a VMS
> 2 - List VMS
3 - Delete a VMS
4 - Execute binary code on VMS
5 - Quit the program

Client ID: 0
Server IP: 127.0.0.1

Reception
Server response :
VMS 1-5:
- VM #2: Busy = 0
```

L'interface du côté client, on voit la dernière réponse du serveur.