

CSE 344 - Midterm Project Report

Sena Özbelen - 1901042601

In this project, we are asked to implement a concurrent file system.

I have 2 code files for 2 sides, **client.c** and **server.c**.

Communication between clients and server is done via FIFO. There are one server fifo and multiple client fifo's. Server fifo gets requests from clients and writes the responses to client's fifo. Critical region handling is done by semaphores. Both files includes `fifocommon.h` containing the common libraries and fifo info.

client.c

In the first part of main function,

- signal handling -

Signals are set in order to handle them. SIGINT(Ctrl-C) and SIGTERM are the signals. When SIGINT is caught, it terminates the program using the flag "terminate" and sends a quit command to server so that it can be removed from the server's clients. SIGTERM is caught when server is terminated since all clients of server should be terminated as well.

After that, the given arguments are checked if they are valid or not. If it is "Connect", then it sets the flag to 1. If it is "tryConnect", then it sets the flag to 2. Afterwards, it starts to execute client side.

Firstly, it creates a client fifo to get the response from server side. It opens the server fifo with the given pid and write only mode to send its requests. It sends a message with a flag for Connect/tryConnect to send a first request.

- client waiting handling -

If the queue is full, then server sends "not added". If client's choice is tryConnect, then the program is terminated. If it is Connect, then it waits for the empty slot. Waiting process is done by a semaphore created for the client. When another client exits server, server checks if there is a client waiting. If so, then the first client's semaphore in the queue is called with `sem_post` in server side so that client stops waiting and starts executing the commands.

- client execution handling -

If the queue is not full, then the received command from client side is written to server fifo and the response is read from the client's fifo. Until a signal or quit command, it keeps getting the commands from the client.

server.c

In the first part of main function,

- signal handling -

Signals are set in order to handle them. SIGINT(Ctrl-C), SIGTERM and SIGUSR1 are the signals. When these signals are caught, they terminate the program using the flag "terminate". SIGUSR1 is generated from killServer command and SIGINT is generated from Ctrl-C. When the program is terminated, the connected clients are terminated as well by sending SIGTERM signal.

After that the given arguments are checked. The current working directory of server is changed by using provided dirname/path and the max number of clients provided by arguments is set. Afterwards, it starts to execute server side.

- client add/remove handling -

It creates a server fifo to get requests from clients. In a while loop, it reads the request from the client. There are 2 arrays, waiting and clients. The array of clients is the clients who can send commands to server at that time. The array of waiting stores the clients waiting for an empty slot. When a request is received from read, it checks whether it is a new client or already registered client. If it is a new client, then it checks the number of clients. If there is an empty slot, it adds the client to the array and sends a message "added". If the queue is already full, it sends a message "not added" and then it checks the received command which contains the flag of client's preference about Connect/tryConnect. If it is Connect, then server adds the client to waiting list.

- client command execution handling -

For registered clients, it directly goes to execution part. Server creates a child using fork and executes the received command. When it receives quit command from client, it removes the client from the array and the first client in the waiting queue is added.

While modifying clients and waiting arrays, one semaphore is used.

For command execution part, there is a reader-writer problem for "readF", "writeT", "download" and "upload". Therefore, there are 50 semaphores in an array to assign to each file. Before executing these commands, it checks whether this file has a semaphore. If not, then the next semaphore in the array is assigned to the file. "readF", "download" and "upload" are readers and "writeT" is writer.

Also, log file write operation is done by using a semaphore.

fileop.c

Operations are mostly done by using commands.

Help command shows the available commands and the usage.

List command runs ls command for the server directory.

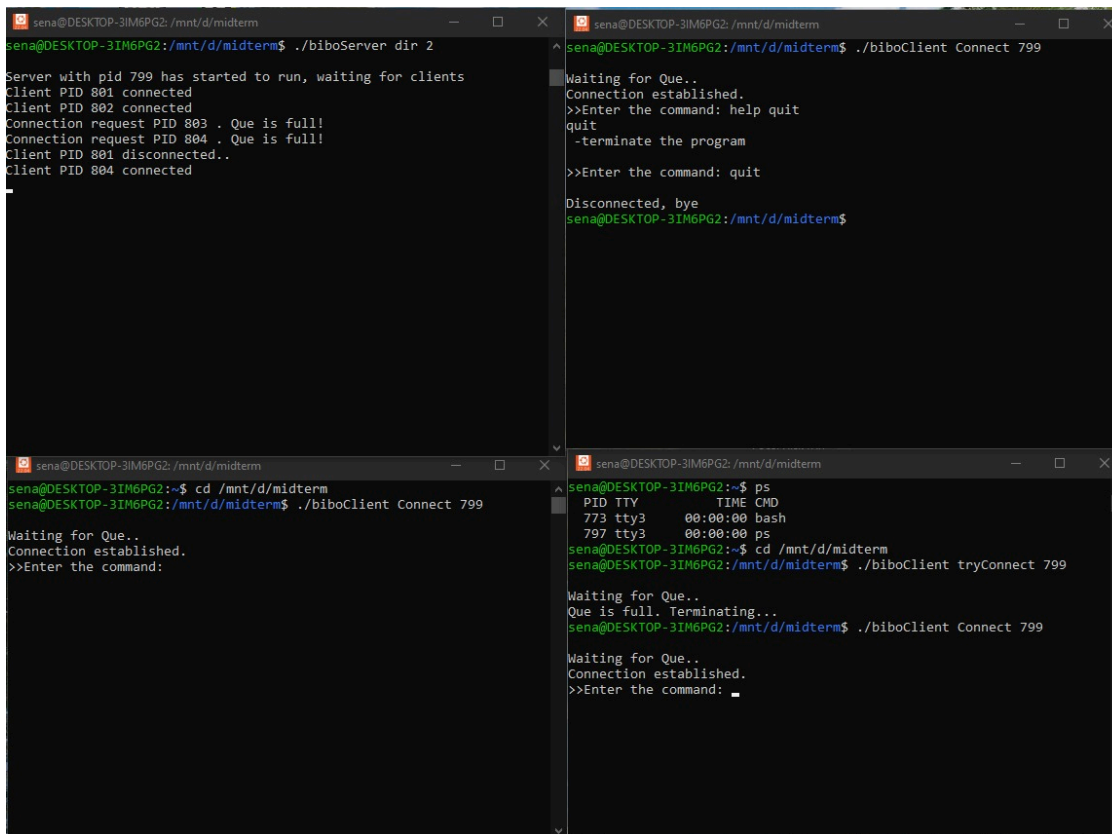
Upload and download commands use cp command since these are all about copying a file to another location. To define the path, it assumes that server's working directory is in the server/client code's directory. It might be inside of other directories. It should work for these cases.

The command popen is used to execute commands because I need the output of the command to send to the client via fifo.

readF and writeT are done by using FILE pointers.

Test Cases

-Waiting/Terminating Case



```
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboServer dir 2
Server with pid 799 has started to run, waiting for clients
Client PID 801 connected
Client PID 802 connected
Connection request PID 803 . Que is full!
Connection request PID 804 . Que is full!
Client PID 801 disconnected..
Client PID 804 connected

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboClient Connect 799
Waiting for Que..
Connection established.
>>Enter the command: help quit
quit
-terminate the program
>>Enter the command: quit

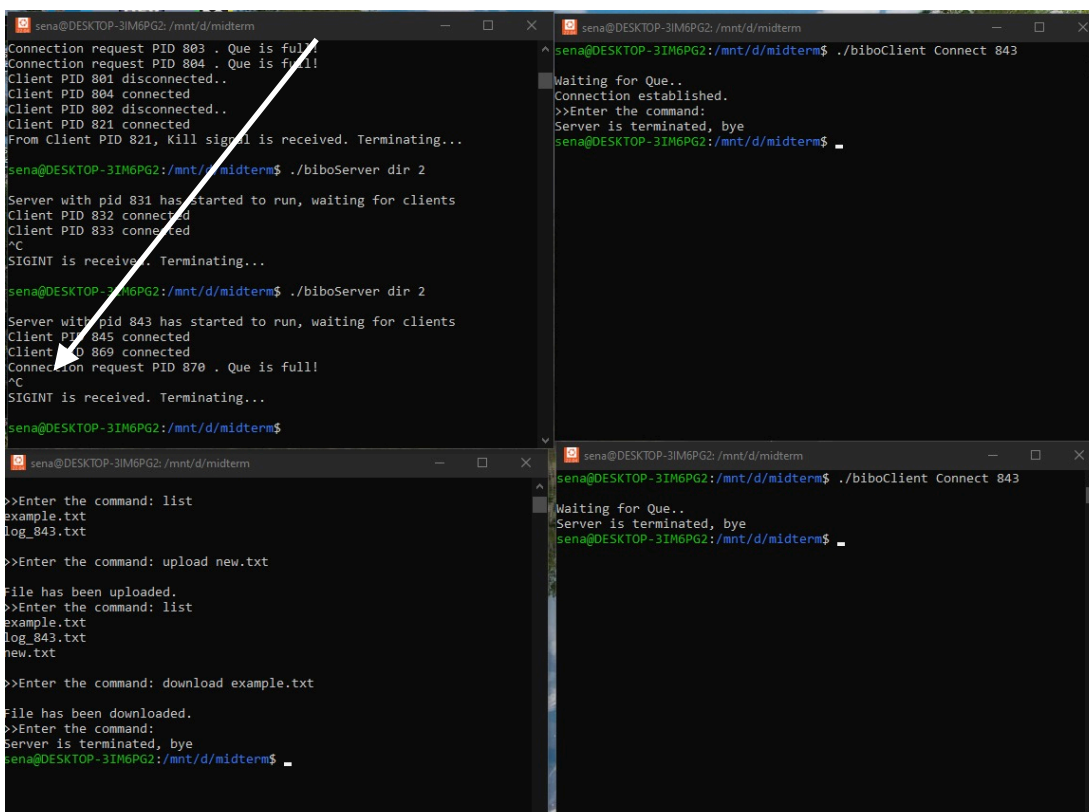
Disconnected, bye
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ cd /mnt/d/midterm
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboClient Connect 799
Waiting for Que..
Connection established.
>>Enter the command:

sena@DESKTOP-3IM6PG2:~$ ps
PID TTY TIME CMD
773 tty3 00:00:00 bash
797 tty3 00:00:00 ps
sena@DESKTOP-3IM6PG2:~$ cd /mnt/d/midterm
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboClient tryConnect 799
Waiting for Que..
Que is full. Terminating...
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboClient Connect 799
Waiting for Que..
Connection established.
>>Enter the command: _
```

The client 803 tries to connect via tryConnect so it is terminated. The client 804 tries to connect via Connect so it waits and when the client 801 exits using quit, 804 is connected to server.

-CTRL-C/killServer Case



```
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboServer dir 2
Connection request PID 803 . Que is full!
Connection request PID 804 . Que is full!
Client PID 801 disconnected..
Client PID 804 connected
Client PID 802 disconnected..
Client PID 821 connected
From Client PID 821, Kill signal is received. Terminating...
^C
SIGINT is received. Terminating...

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboServer dir 2
Server with pid 831 has started to run, waiting for clients
Client PID 832 connected
Client PID 833 connected
^C
SIGINT is received. Terminating...

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboServer dir 2
Server with pid 843 has started to run, waiting for clients
Client PID 845 connected
Client PID 869 connected
Connection request PID 870 . Que is full!
^C
SIGINT is received. Terminating...

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$

>>Enter the command: list
example.txt
log_843.txt

>>Enter the command: upload new.txt
File has been uploaded.

>>Enter the command: list
example.txt
log_843.txt
new.txt

>>Enter the command: download example.txt
File has been downloaded.

>>Enter the command:
Server is terminated, bye
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboClient Connect 843
Waiting for Que..
Connection established.
>>Enter the command:
Server is terminated, bye
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboClient Connect 843
Waiting for Que..
Server is terminated, bye
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$
```

Server receives CTRL-C so all clients including waiting one are terminated by sending SIGTERM.

```
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboServer dir 2
Server with pid 874 has started to run, waiting for clients
Client PID 875 connected
Client PID 876 connected
Connection request PID 877 . Que is full!
From Client PID 876, Kill signal is received. Terminating...
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ _

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboClient Connect 874
Waiting for Que..
Connection established.
>>Enter the command: killServer
Server is terminated, bye
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ _

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboClient Connect 874
Waiting for Que..
Connection established.
>>Enter the command:
Server is terminated, bye
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ _

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboClient Connect 874
Waiting for Que..
Server is terminated, bye
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ _
```

Server receives killServer from the client so all clients are again terminated by sending SIGTERM.

-CTRL-C Client Case

```
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ ./biboServer dir 2
Server with pid 799 has started to run, waiting for clients
Client PID 801 connected
Client PID 802 connected
Connection request PID 803 . Que is full!
Connection request PID 804 . Que is full!
Client PID 801 disconnected..
Client PID 804 connected
Client PID 802 disconnected..
_

sena@DESKTOP-3IM6PG2:/mnt/d/midterm$ >>Enter the command: ^C
Disconnected, bye
sena@DESKTOP-3IM6PG2:/mnt/d/midterm$
```

-upload/download Case

```
>>Enter the command: list
example.txt
log_843.txt

>>Enter the command: upload new.txt
File has been uploaded.
>>Enter the command: list
example.txt
log_843.txt
new.txt
```

```
>>Enter the command: download example.txt
File has been downloaded.
>>Enter the command:

example.txt 5/16/2023 10:11 PM
fifocommon.h 5/16/2023 1:09 PM
fileop.c 5/16/2023 9:45 PM
```

-Client Waiting Queue Case

```
sena@DESKTOP-3IM6PG2: /mnt/d/midterm$ ./biboServer dir 2
Server with pid 893 has started to run, waiting for clients
Client PID 894 connected
Client PID 895 connected
Connection request PID 896 . Que is full!
Connection request PID 897 . Que is full!
Client PID 894 disconnected..
Client PID 896 connected
Client PID 895 disconnected..
Client PID 897 connected

sena@DESKTOP-3IM6PG2: /mnt/d/midterm$ ./biboClient Connect 893
Waiting for Que..
Connection established.
>>Enter the command: ^C
Disconnected, bye
sena@DESKTOP-3IM6PG2: /mnt/d/midterm$

sena@DESKTOP-3IM6PG2: /mnt/d/midterm$ ./biboClient Connect 893
Waiting for Que..
Connection established.
>>Enter the command:

sena@DESKTOP-3IM6PG2: /mnt/d/midterm$ ./biboClient Connect 893
Waiting for Que..
Connection established.
>>Enter the command:
```

Clients 896 and 897 are added to the waiting queue and when the first client exits, the client 896 is connected to server since it is the first client in the queue.

Not Handled Parts

- writeT doesn't handle a string with spaces and quotes. Example usage : writeF ex.txt string. It doesn't write the string if the given line number is larger than the total line number.
- readF can only show the content limited by fifo's response size (4000 bytes in the code)
- There are only 50 semaphores for files so this server can only handle 50 files. (This can be changed from the code)
- The request size is also limited (100 bytes in the code)