II Assignment – Report

Authors: Ilenia D’Angelo, Giovanni Di Marco

Code

The aim of this code is to connect a client with a server, using socket, to allow the communication of a command from the client to the server and a position, with its string to explain the situation, from the server to the client. These two processes run separately and are not correlated, so we use a different shell for each one.

- **Client:** The client asks for a connection to the server using a socket, then, when the communication is established, it forks one time. In this way we have two process:
  1. **Father:** which asks, as input from the shell, a command and writes it to server using socket.
  2. **Child:** which reads from the socket what the server has written and stamps it on the screen.

  We have chosen to use two different concurrent processes, one for the reading and one for the writing, because they are asynchronous, so they must run separately to not block the reading phase, which must continue during all the process, with the writing phase, that must work only if a command has been taken from the shell.
- **Server:** The server is composed of 4 processes:
  1. **Super-process:** It waits for a request of connection from the client and it accepts it. Then it forks 3 processes. It stores the PID of process H in a variable and sends it to the process “Server to receive”, that is useful to allow the communication between two child processes, H and “Server to receive”.
  2. **Server to receive:** It receives using an unnamed pipe, the PID of H from the Super-process and stores it in a variable. It waits the command from socket then, after reading that, it sends it to the process H, using another unnamed pipe, with a file descriptor. This part of the server sends a signal to the H process to advert it that there is something, a command, to read in the pipe.
  3. **H:** It simulates the behaviour of a motor. So, it can take a command: go up (1), stop (2), go down (3). We have used a switch to represent the movement of the hoist, which can move (or stop) a hook up and down of 5cm each second. In any case, even if there is no command from the pipe, it sends over a pipe the position of the hook and a string which explains what it is doing to the “Server to send”. It receives a signal which, with a flag allows the reading from pipe.
  4. **Server to send:** It reads what has been written from the H, so the position on the z axis of the hook and the proper string and then sends it to the client using socket.

Each process of client and server runs into an endless loop. The message from the H is a structure with two fields: position and status of the hook. The message from the Client is a single int, the only expected values for the command are: 1, 2 or 3.

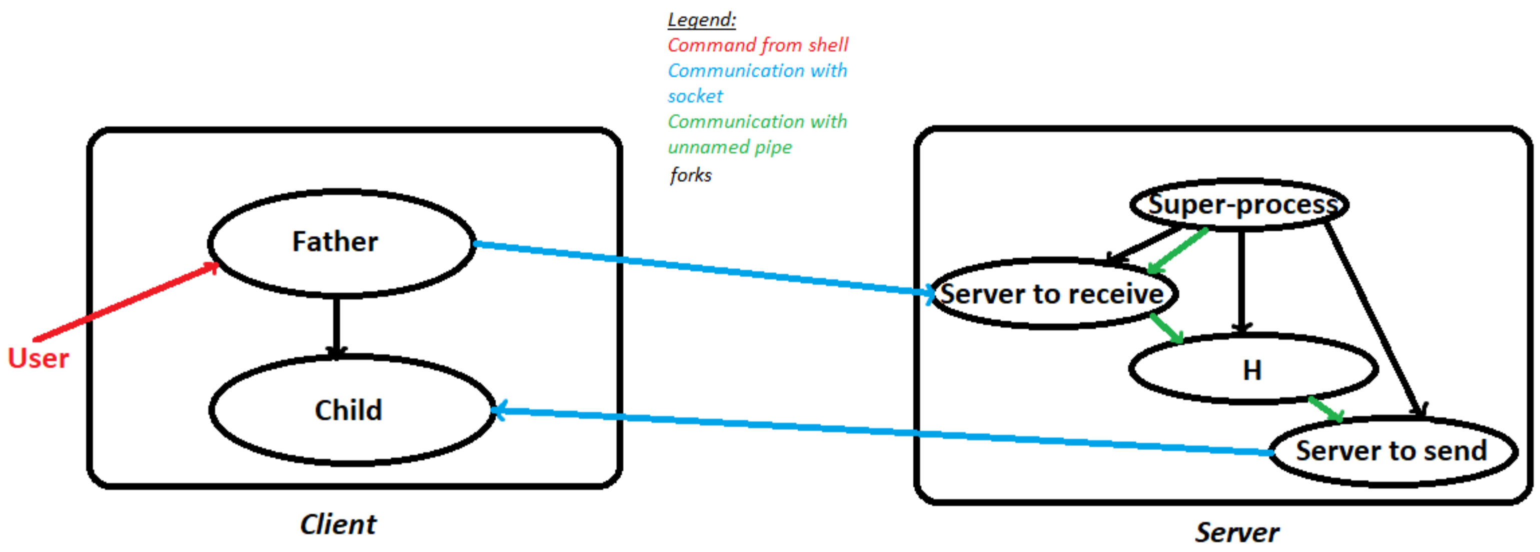
Some issues

To allow the use of the signal between two children, it is necessary to know the PID of the process which must receive the signal. Because only the father process knows, easily, the PID of all its children and in our program the signal is exchanged between two children, we have resolved this issue sending the PID of the process, that we need, from the father, our Super-process, to the child that must use it through an unnamed pipe. Without a Super-process, using the most suitable process as father, this part is more direct. But we have chosen to have a Super-process for connecting socket and forking, to have and easily management of every child process and any future developments. Another possible way, to not use a pipe to communicate the PID, is to think in a different order the forking of the children processes, this approach is surely a good method to improve the program.

Results

In our program we can simulate very well a motor with the H process and all the interaction and exchanging of data between a user, a client and a server.

A sketch of how transmission of data works in this project:


Legend:
Command from shell
Communication with socket
Communication with unnamed pipe
forks
User
Father
Child
Client
Super-process
Server to receive
H
Server to send
Server