Network Application Programming

GP23

Programming with sockets

This laboratory develops the competence to programm applications, more precisely we will programm a server application using the socket interfaces. The solution shall be uploaded to Blackboard and reviewed by the instructor.

Programming with Socket interfaces

NETWORK ARCHITECTURE

In this lab we will review an Echo service. The proposed task is to complete the existing code (copy available in Blackboard/Aula Virtual) that implements a connective server. We will use a Telnet client, then **it is not necessary to implement the client part.**

The student shall include the necessary sentences, replacing the "***XXXXXXX***" as needed to enable the communication between the telnet client and the implemented server. Links are provided to the official Python Help services.

Communications will take place with TCP protocol, on a port with number chosen by the student. In order to prevent problems with server listening, the selected port number must be higher than 5000.

The code of the ECHO server part is shown below

```
import socket
import sys

HOST = ''  # Nombre o direccion de la maquina. Puede estar vacio en el servidor
PORT = 8888  # Puerto de escucha del servidor. PORT >= 5000
BUFFER = 1000 # Tamano del buffer de recepcion

s = socket.XXXXXXX (socket.AF_INET, socket.SOCK_STREAM)
print 'Socket creado...'

# Enlaza el socket con la direccion y puerto local
try:
    s.bind((XXXXXXX, XXXXXXX))

# Captura excepciones. Si existe, un error sale del programa
except socket.error as msg:
    print 'Fallo en BIND. Cod. Error: ' + str(msg[0]) + ' Mensaje ' + msg[1]
    sys.exit()
print 'Socket Enlazado...'

# Comienza la escucha en el socket
s.listen(10)
print 'Socket escuchando...'

while 1:
    # Espera a que lleguen clientes.
    cliente, addr = s.XXXXXXX()
    print 'Conectado con ' + addr[0] + ':' + str(addr[1])

    while 1:
            # Recepcion de datos
            datos = cliente.XXXXXXX (XXXXXXX)
            if datos:
                cliente.XXXXXXX (XXXXXXX)
                print 'Recibido: ' + datos
                 # Se dejara de trabajar con el cliente si se recibe "EXIT"
```

```
                if datos[0:4] == 'EXIT':
                        cliente.close()
                        break
s.close()
```

When completing the previous code, you should take into account:

- Indentation (inserting tabs) in Python serves as braces { } in C/C++ .
- Neither use accents in the code, nor in the comments.

After completing the code and verifying its correct operation, respond to the following questions:

1. When the server prints a line starting with “Connected to” , what correspond the data shown in the line to?

2. Connect two clients to the same server. How does the server behave?

3. Connect to the server from a web browser. What happens? What is displayed?