

**UNIVERSIDAD CENTRAL**  
**FACULTAD DE INGENIERÍA Y CIENCIAS BÁSICAS**  
**DEPARTAMENTO DE INGENIERÍA DE SISTEMAS**

**SISTEMAS DISTRIBUIDOS**

(40050172, 43390862)

**Laboratorio: Tecnologías de comunicación distribuida**

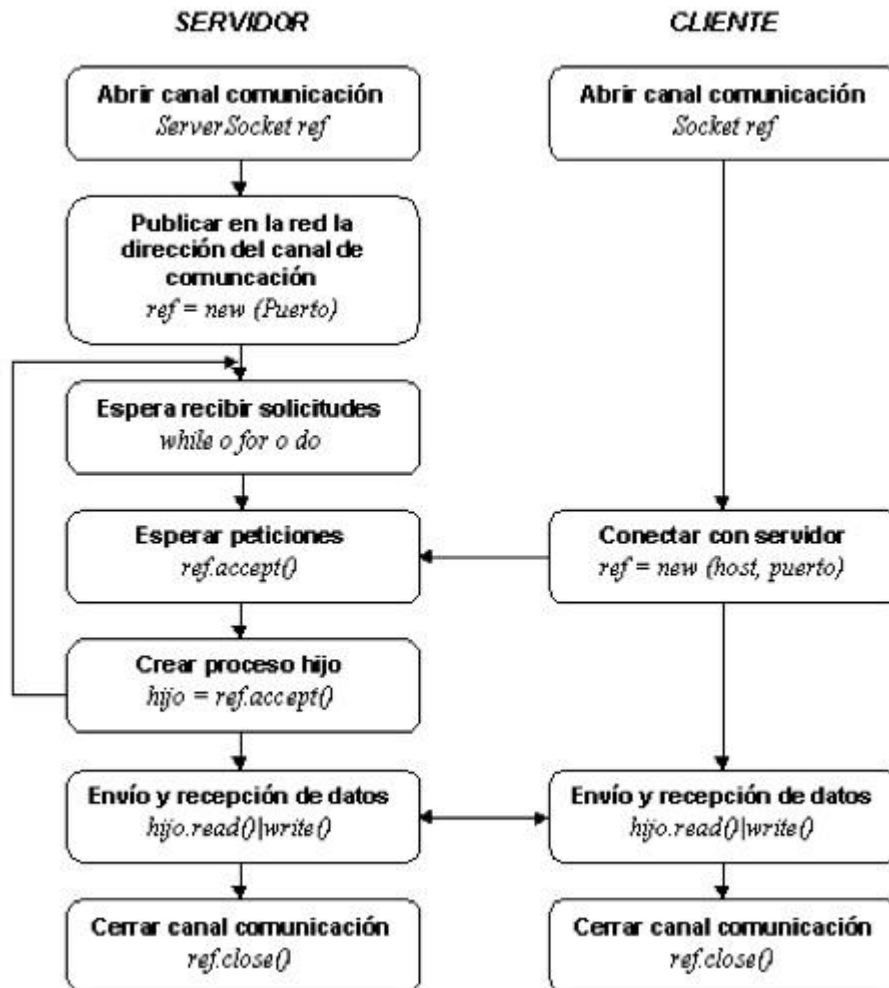
**Primera parte: (2.5)**

**Haga una comprobación del concepto de socket TCP con los datos descritos en esta primera parte de la guía.**

**Sockets**

Los sockets son una forma de comunicación entre procesos que se encuentran en diferentes máquinas de una red, los sockets proporcionan un punto de comunicación por el cual se puede enviar o recibir información entre procesos.

Los sockets tienen un ciclo de vida dependiendo si son sockets de servidor, que esperan a un cliente para establecer una comunicación, o socket cliente que busca a un socket de servidor para establecer la comunicación.



## Clases para los sockets TCP

Clase	Descripción
Socket	Esta clase implementa sockets del cliente Un socket es uno de los extremos en la comunicación entre dos máquinas.
ServerSocket	Esta clase implementa sockets del servidor. Un socket del servidor espera a que una solicitud provenga de la red; lleva a cabo determinadas operaciones basadas en la solicitud recibida; y entonces, posiblemente, retorna un resultado al solicitante.

## Clases para los sockets UDP

Clase	Descripción
DatagramPacket	<p>Esta clase representa un paquete datagrama.</p> <p>Los paquetes datagramas son usados para implementar el servicio de entrega de paquetes sin conexión. Cada mensaje es enrutado desde una máquina a otra con base en la información contenida dentro del paquete, únicamente. Cuando se envían múltiples paquetes de una máquina a otra, estos pueden seguir diferentes rutas, y pueden llegar en cualquier orden. La entrega de paquetes no está garantizada.</p>

## Sockets TCP

A continuación, se muestran dos fragmentos de código en java que hacen uso de sockets TCP, el código se trata de una aplicación que actúa como cliente y otra como servidor, que enviará un mensaje en cadena de caracteres y recibirá una respuesta en cadena de caracteres.

```
import java.net.*; //Biblioteca de funciones de red
import java.io.*;  //Biblioteca de manejo de flujos

public class ServidorTCP {
    public static void main(String[] args) throws Exception {
        ServerSocket server = new ServerSocket(5000); //Iniciar socket de escucha TPC en
                                                    // el puerto 5000
        String msg="Hola cliente te habla un servidor Java\n"; //Mensaje a enviar al
                                                    //cliente
        byte []datos= new byte [256]; //Buffer de datos recibidos
        System.out.println("Iniciando el servidor..");
        while(true) {
            System.out.println("Hecho. Esperando clientes...\n");
            Socket socket = server.accept(); //Aceptar al cliente y establecer
                                                    //comunicación con este
            DataInputStream dis = new DataInputStream(socket.getInputStream()); //Habilitar el flujo de entrada
                                                    //para el socket
            DataOutputStream dos = new DataOutputStream(socket.getOutputStream()); //Habilitar el flujo de salida
                                                    //para el socket
            dis.read(datos,0,datos.length); //Captar el mensaje del cliente
            System.out.println("Se ha detectado a un cliente.\n El cliente dice: "
                + new String(datos)); //Imprimir en pantalla el mensaje
            System.out.println("Enviando el Mensaje de respuesta:"+msg+"...");
            dos.write(msg.getBytes(),0,msg.length()); //Enviar un mensaje de respuesta
                                                    //al cliente
            dis.close(); //Cerrar los flujos de entrada
            dos.close(); //y salida del socket
            socket.close(); //cerrar el socket y finalizar la
                            //comunicación
        }
    }
}
```

Programa Servidor TCP

```

import java.net.*; //Biblioteca de funciones de red
import java.io.*;  //Biblioteca de manejo de flujos

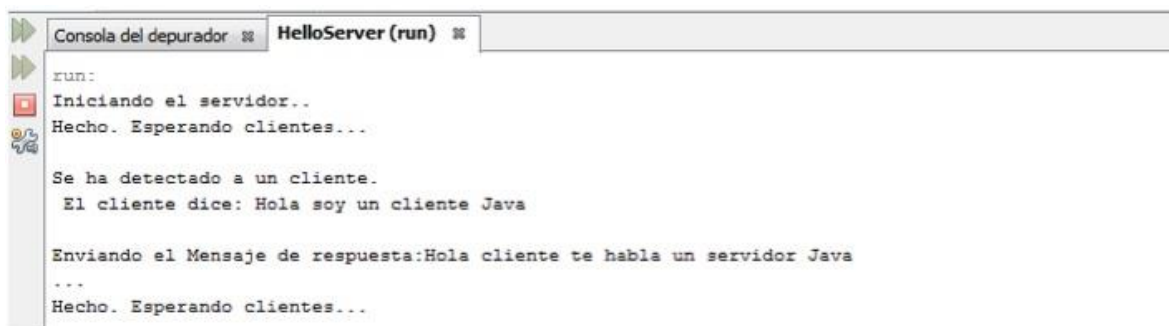
public class Programa2g {
    public static void main(String[] args) throws Exception {
        byte []datos= new byte [256]; //buffer de datos recibidos
        String dirIP; //variable que almacena la IP del servidor
        String msg = "Hola soy un cliente Java\n"; //Mensaje a enviar
        System.out.println("Escriba la dirección IP a conectarse:");
        BufferedReader x = //Habilitar la entrada del teclado
            new BufferedReader(new InputStreamReader(System.in));
        dirIP=x.readLine(); //Pedir la dirección IP del servidor
        Socket socket =new Socket(dirIP,5000); //Inicializar la comunicación con el
            //servidor en el puerto 5000
        DataInputStream din = new //Habilitar el flujo de entrada para
            DataInputStream(socket.getInputStream()); //el socket
        DataOutputStream dos = new //Habilitar el flujo de salida para
            DataOutputStream(socket.getOutputStream()); //el socket
        System.out.println("Enviando un mensaje a servidor:"+msg);
        dos.write(msg.getBytes()); //Enviar el mensaje al servidor en bytes
        dos.flush();
        din.read(datos,0,datos.length); //Leer el mensaje del servidor
        String message = new String(datos); //Convertir a cadena el mensaje
            //recibido
        System.out.println(message); //Imprimir el mensaje del servidor
        din.close(); //Cerrar los flujos de entrada
        dos.close(); //y salida del socket
        socket.close(); //cerrar el socket y finalizar la
            //comunicación
    }
}

```

### Programa Cliente TCP

Al ejecutar el programa se debe obtener unos resultados como los que se muestran a continuación:

El resultado en el servidor como cliente tendrán un aspecto parecido al siguiente:



```

run:
Iniciando el servidor..
Hecho. Esperando clientes...

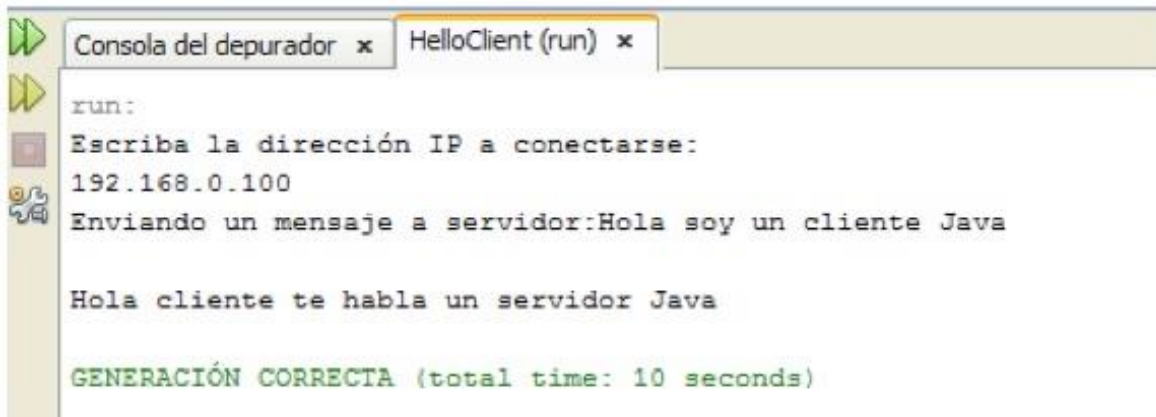
Se ha detectado a un cliente.
El cliente dice: Hola soy un cliente Java

Enviando el Mensaje de respuesta:Hola cliente te habla un servidor Java
...
Hecho. Esperando clientes...

```

## Servidor TCP

El resultado en el cliente como cliente tendrán un aspecto parecido al siguiente:



```
run:
Escriba la dirección IP a conectarse:
192.168.0.100
Enviando un mensaje a servidor:Hola soy un cliente Java

Hola cliente te habla un servidor Java

GENERACIÓN CORRECTA (total time: 10 seconds)
```

## Cliente TCP

**Segunda parte: (2.5)**

**Haga una comprobación del concepto de socket UDP.**

**Haga un informe donde presente:**

1. **Objetivo**
2. **Marco Teórico**
3. **Procedimiento**
4. **Resultados**
5. **Conclusiones**

Recuerden que todos los trabajos (Tareas, talleres, etc.) se presentan en grupo y cada integrante debe subir una copia de su trabajo a su cuenta del aula virtual.

