

## Experiment No:4

**Aim:**WAP to implement Socket Programming using TCP & UDP

### Theory:

**Socket** : the communication object.

A socket connection is a 4-tuple -- (HostA, PortA, HostB, PortB) -- uniquely defining the connection.

### Transmission Control Protocol (TCP)

TCP provides a *connection oriented service*, since it is based on connections between clients and servers.

TCP provides reliability. When a TCP client send data to the server, it requires an acknowledgement in return. If an acknowledgement is not received, TCP automatically retransmit the data and waits for a longer period of time.

**TCP properties:** reliable, connection-oriented, byte-stream, connection established before application-level protocols exchange information, two-way communication

**The client-server model:**The client-server model is one of the most used communication paradigms in networked systems. Clients normally communicates with one server at a time. From a server's perspective, at any point in time, it is not unusual for a server to be communicating with multiple clients. Client need to know of the existence of and the address of the server, but the server does not need to know the address of (or even the existence of) the client prior to the connection being established

### TCP Socket API

The sequence of function calls for the client and a server participating in a TCP connection is presented in below..

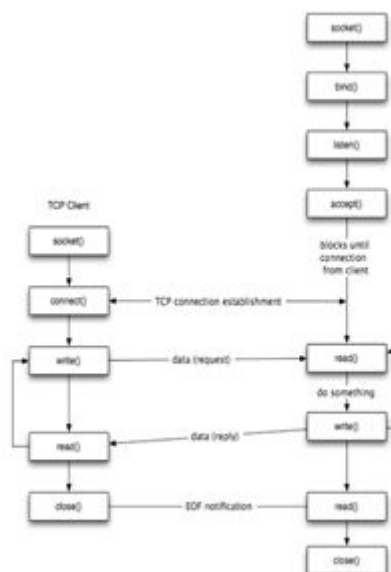


Fig: TCP Client Server

As shown in the figure, the steps for establishing a TCP socket on the client side are the following:

- Create a socket using the `socket()` function;
- Connect the socket to the address of the server using the `connect()` function;
- Send and receive data by means of the `read()` and `write()` functions.

The steps involved in establishing a TCP socket on the server side are as follows:

- Create a socket with the `socket()` function;
- Bind the socket to an address using the `bind()` function;
- Listen for connections with the `listen()` function;
- Accept a connection with the `accept()` function system call. This call typically blocks until a client connects with the server.
- Send and receive data by means of `send()` and `receive()`.

## B) WAP to implement socket programming using UDP.

### Theory:

Datagram sockets, also known as connectionless sockets, which use User Datagram Protocol (UDP). Stream sockets, also known as connection-oriented sockets, which use Transmission Control Protocol (TCP), Stream Control Transmission Protocol (SCTP) or Datagram Congestion Control Protocol (DCCP).

**UDP properties:** unreliable, packet-switched, packet data, no connection overhead, application-level protocols exchange information immediately, two-way communication.

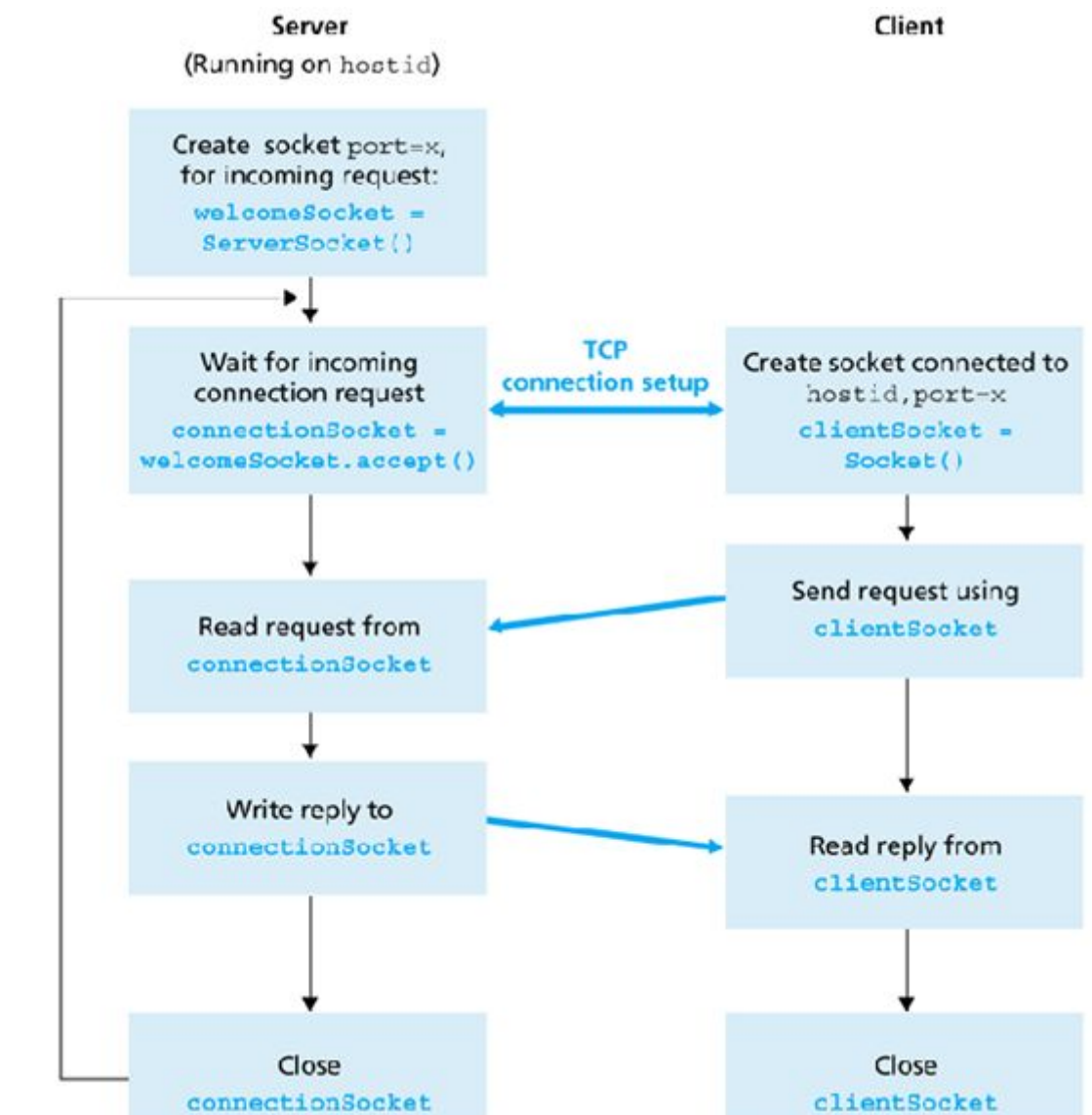


Fig. TCP/IP client/server communication flow

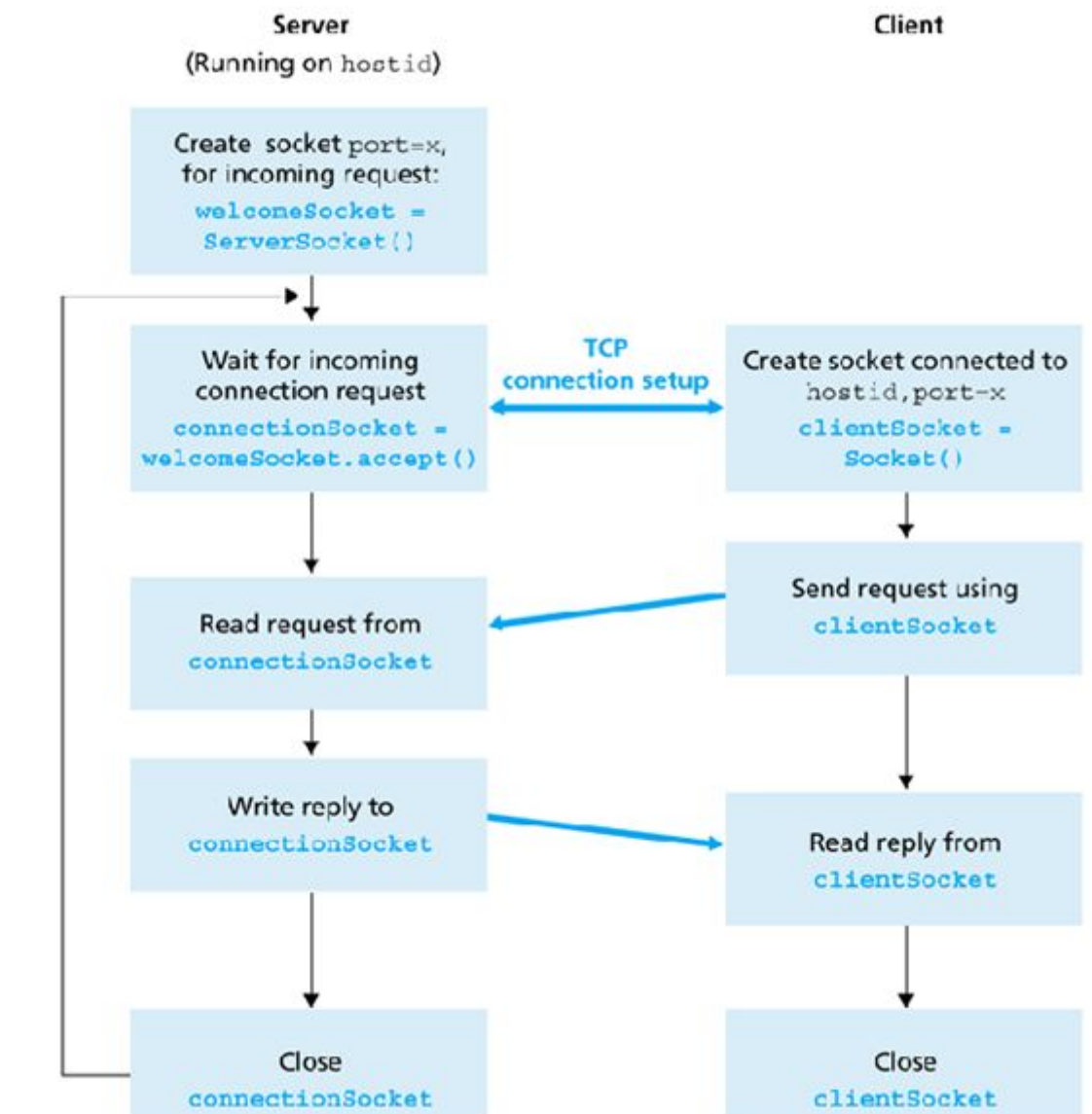


Fig: UDP client/server communication flow:

TCP/IP Vs UDP :

1. Connection oriented (bidirectional communication sockets which keeps track of connection state, established connection , and also create receiver buffer on both sides of the end to end connection) & other is Connection less (does not keep track of the connection state and it does not have receiver buffer on either side sender or receiver)
2. Will have Acknowledgment & other one don't
3. Performance slow & other one fast
4. More secure & other is not much
5. TCP/IP offer guaranteed delivery while UDP does not
6. TCP/IP consumes high bandwidth. UDP is good guy and shares band with everyone.
7. TCP/IP guarantees sequencing of packets (Packet sent first will reach destination first). In UDP you may get last packet first or not at all.
8. TCP does not have message block boundaries (User has to define its own)
9. TCP can Transmit large amount of data as compared to udp
10. Sequencing of packet is guaranteed in TCP. Means the packets that are sent is delivered in time where in UDP it is not guaranteed that the packets will reach in time to the destination.

## Code:

### AJTCP

Server Side:

```
import java.net.*;
import java.io.*;
class ServerSide
{
    public static void main(String[] args) throws Exception
    {
        int choice,a,b,c=0;
        ServerSocket ss = new ServerSocket(1024);
        Socket s = ss.accept();
        BufferedReader br = new BufferedReader(new
InputStreamReader (s.getInputStream() ) );
        choice =Integer.parseInt(br.readLine());
        a =Integer.parseInt(br.readLine());
        b = Integer.parseInt(br.readLine());
        switch(choice)
        {
            case 1 : c = a+b;
            break;
            case 2 : c = a-b;
            break;
            case 3 : c = a*b;
            break;
            case 4 : c = a/b;
            break;
            case 5 : c = (a%b);
            break;
        }
        PrintStream pr = new PrintStream(s.getOutputStream()) ;
        pr.println(c);
        ss.close();
        s.close();
    }
}
```

Client Side:

```
import java.net.*;
import java.io.*;
class ClientSide
```

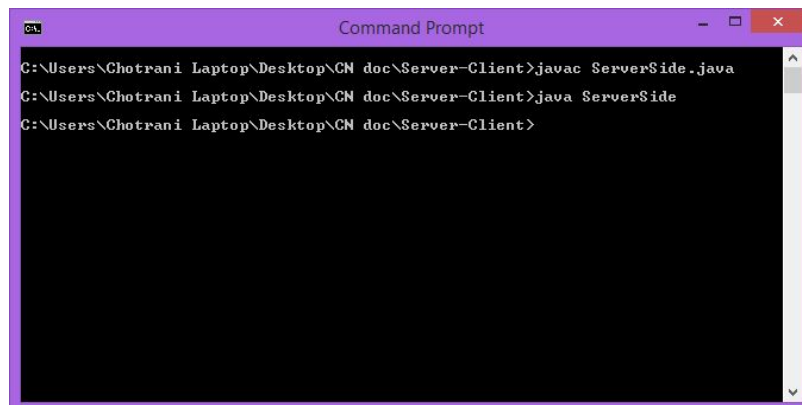
```

{
    public static void main(String[] args) throws Exception
    {
        int ch=0,a,b,c;
        Socket s = new Socket("localhost",1024);
        BufferedReader br =
newBufferedReader(newInputStreamReader(System.in));
        PrintStream ps=new PrintStream(s.getOutputStream());
        System.out.println("Please Enter Number 1:");
        a = Integer.parseInt(br.readLine());
        System.out.println("Please Enter Number 2:");
        b = Integer.parseInt(br.readLine());
        System.out.println("Please Enter The Operation to Be
                                Performed\n");
        System.out.println("1.Addition 2.Subtraction
                                3.Multiplication 4.Divison 5.Modulo 0.Exit"); ch =
                                Integer.parseInt(br.readLine());
        ps.println(ch);
        ps.println(a);
        ps.println(b);
        BufferedReader br1 = new BufferedReader(new
InputStreamReader(s.getInputStream()));
        c=Integer.parseInt(br1.readLine());
        System.out.println("Answer: "+c);
        s.close();
    }
}

```

#### OUTPUT:

Server Side:

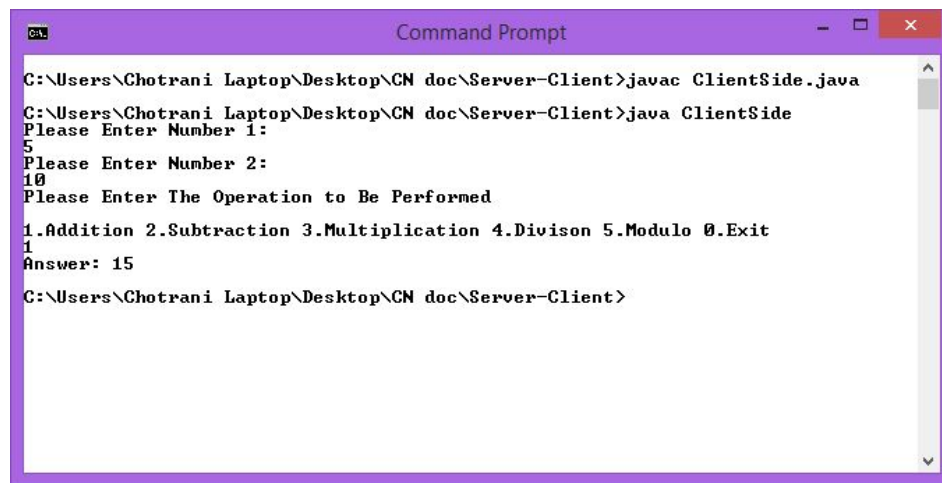


```

C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>javac ServerSide.java
C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>java ServerSide
C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>

```

Client Side:



```
CA Command Prompt
C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>javac ClientSide.java
C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>java ClientSide
Please Enter Number 1:
5
Please Enter Number 2:
10
Please Enter The Operation to Be Performed
1.Addition 2.Subtraction 3.Multiplication 4.Divison 5.Modulo 0.Exit
1
Answer: 15
C:\Users\Chotrani Laptop\Desktop\CN doc\Server-Client>
```

## BJUDP

Server Socket

Program:

```
import java.io.*;
import java.util.*;
import java.net.*;
class ServerUDP
{
    public static void main(String args[])throws IOException
    {
        DatagramSocket ss=new DatagramSocket(2100);
        byte[] sendData=new byte[1024];
        byte[] recData=new byte[1024];
        while(true)
        {
            DatagramPacket dp=new
            DatagramPacket(recData,recData.length);
            ss.receive(dp);
            String input=new String(dp.getData());
            if(input=="end")
                break;
            InetAddress ip=dp.getAddress();
            System.out.println("Received input : "+input);
            String output="Hello ";
            int port=dp.getPort();
            output=output.concat(input);
            sendData=output.getBytes();
            DatagramPacket dp1=new
            DatagramPacket(sendData,sendData.length,ip,port);
            ss.send(dp1);
        }
        ss.close();
    }
}
```

Client Socket

Program:

```
import java.io.*;
import java.util.*;
import java.net.*;

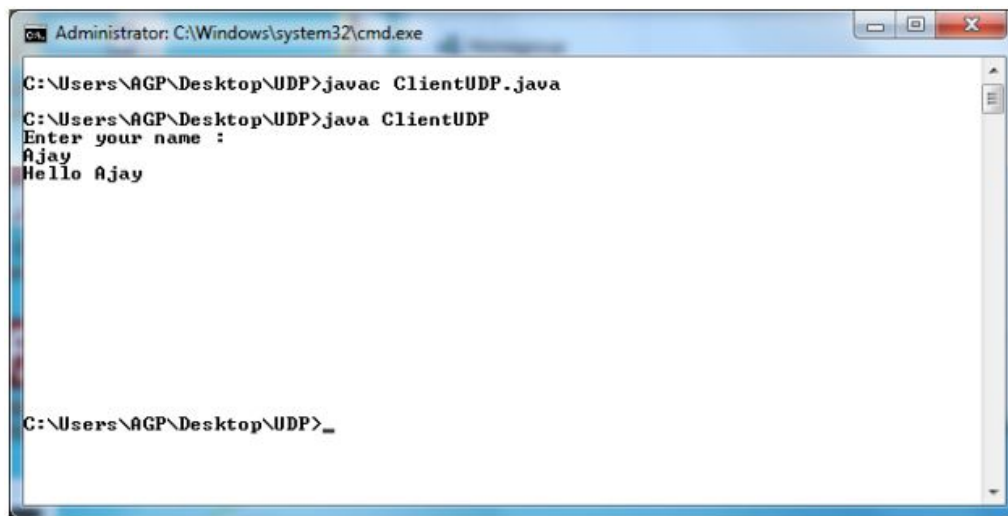
class ClientUDP
{public static void main(String args[])throws IOException
    {
        BufferedReader br=new BufferedReader(new
            InputStreamReader(System.in));
        DatagramSocket ds=new DatagramSocket();
        InetAddress ip=InetAddress.getByName("localhost");
        byte[] sendData=new byte[1024];
        byte[] recData=new byte[1024];
        System.out.println("Enter your name :");
        String input=br.readLine();
        sendData=input.getBytes();
        DatagramPacket dp=new
            DatagramPacket(sendData,sendData.length,ip,2100);
        ds.send(dp);
        DatagramPacket dp1=new
            DatagramPacket(recData,recData.length);
        ds.receive(dp1);
        String output=new String(dp1.getData());
        System.out.println(output);
        ds.close();
    }
}
```

## OUTPUT:

Server Side:



Client Side:



```
Administrator: C:\Windows\system32\cmd.exe
C:\Users\AGP\Desktop\UDP>javac ClientUDP.java
C:\Users\AGP\Desktop\UDP>java ClientUDP
Enter your name :
Ajay
Hello Ajay

C:\Users\AGP\Desktop\UDP>_
```

**Conclusion:** Hence we successfully studied and implemented the program of TCP and UDP.

**Date:**

**Sign:**

**Grade:**