##### 9.6 Implement chating using TCP/UDP socket (between two or more users.)

**OBJECTIVES:**

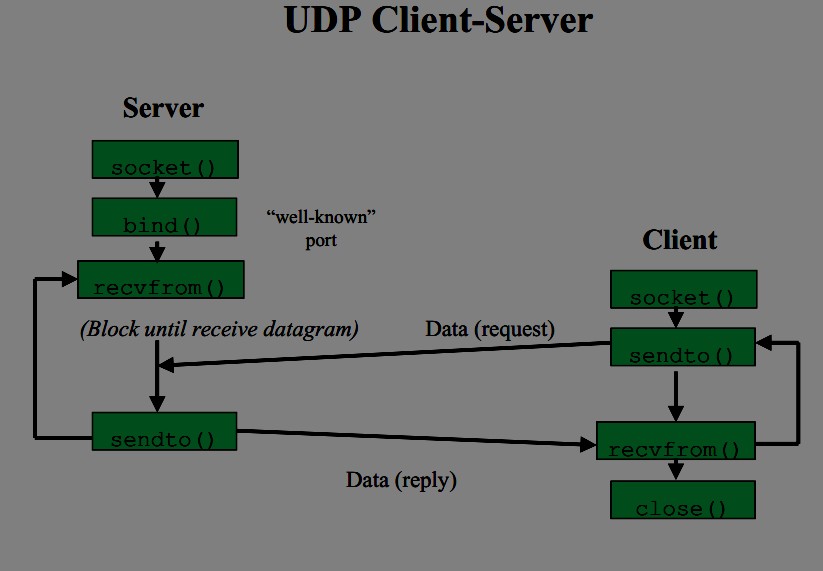
1. To learn about fundamentals of IPC through C socket programming.
2. Learn and understand the OS intraction with socket programming.
3. Use of system call and IPC mechanism to write effective application programs.
4. To know the port numbersing and process relation.
5. To knows the iterative and concurrent server concept.

#### THEORY:

JAVA SOCKET PROGRAMMING

TCP is a connection-oriented protocol layered on the top of IP of the TCP/IP stack with the ability to acknowledge receipt of packets at both ends. Acknowledgement ensures that the lost/corrupt packets can be retransmitted upon request. It also maintains a sequence in the sense that packets can be put back in the same order at the receiving end as they were transmitted.

Although everything seems fair and advantageous at first look, it is also its weakness on occasion, because maintaining a guaranteed data transmission carries a fair amount of overhead (the header size of TCP packet is 20 bit whereas UDP header is 8 bit). In a situation where the order of the data is not that important or say, loss of a few packets does not matter to the verge of completely corrupting the data, TCP can be a real bottleneck. UDP is an unreliable connectionless protocol that neither guarantees that the packets will ever reach the destination nor that they will arrive in the same order they were sent. But, it works and surprisingly reaches the destination, without the slightest aura of "guarantee" or "reliability." TCP can be best suited for file transfer or the like where loss of bits is unacceptable. UDP, on the other hand, is best suited where a little loss in the transmission bits does not matter. For example, a few lost bits in video or audio signals are less severe without much quality degradation. Further, error correction in UDP can be built into data streams at the application level to account for missing information. So, UDP is not a total loss, after all.



CODE:

SERVER:

import java.net.\*; import java.io.\*; import java.util.\*; class Server2

{

public static void main(String []args)throws Exception

{

Scanner sc=new Scanner(System.in);

ServerSocket ss=new ServerSocket(5050);//5050 is port no. System.out.println("Server is Waiting ");

Socket s=ss.accept();//waiting for client

DataOutputStream dos=new DataOutputStream(s.getOutputStream()); DataInputStream dis=new DataInputStream(s.getInputStream()); String str="Welcomes u";

dos.writeUTF(str); //sends msg to client

str=dis.readUTF(); //reads msg send by client System.out.println("From client:"+" "+str); while(true)

{

System.out.print("\nEnter Message(from Server to client):"); str=sc.nextLine();

dos.writeUTF(str); if(str.equals("bye")) break; str=dis.readUTF();

System.out.print("From Client:"+" "+str); if(str.equals("bye"))

break;

}

try

{

ss.close();

s.close();

dos.close();

dis.close();

}

catch(Exception e){}

}

} CLIENT:

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

class Client2

{

public static void main(String []args)throws Exception

{

Scanner sc=new Scanner(System.in); Socket s=new Socket("localhost",5050);

DataOutputStream dos=new DataOutputStream(s.getOutputStream()); DataInputStream dis=new DataInputStream(s.getInputStream()); String str=dis.readUTF(); //receives msg send by server System.out.println("From server:"+" "+str);

str="Thank u";

dos.writeUTF(str); //writes to server while(true)

{

str=dis.readUTF(); System.out.println("From Server:"+" "+str); if(str.equals("bye"))

break;

System.out.print("Enter Message(from Client to Server):"); str=sc.nextLine();

dos.writeUTF(str); if(str.equals("bye")) break;

}

try

{

s.close();

dos.close();

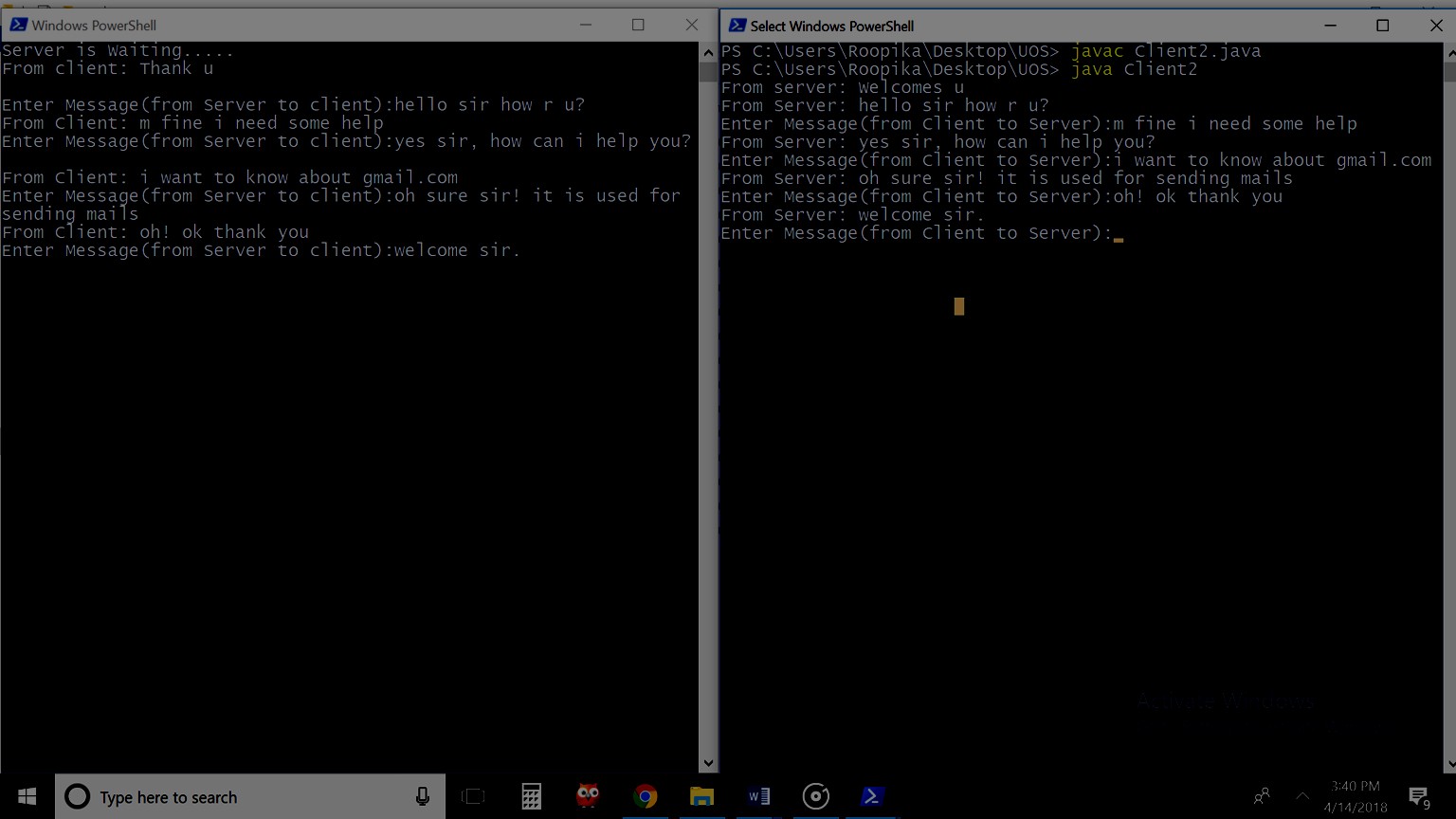
dis.close();

}

catch(Exception e){}

}

}



#### Conclusion:

Multiway communication from client to server and vice versa can be implemented in the form of chatting by using TCP connection and socket programming in C.

#### References :

[www.cs.cf.ac.uk/Dave/C/CE.html](http://www.cs.cf.ac.uk/Dave/C/CE.html)