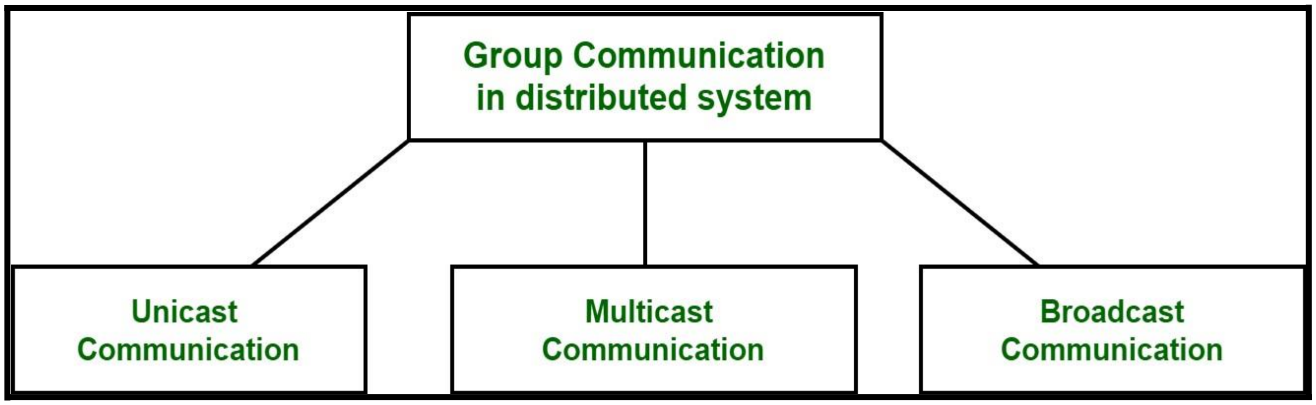
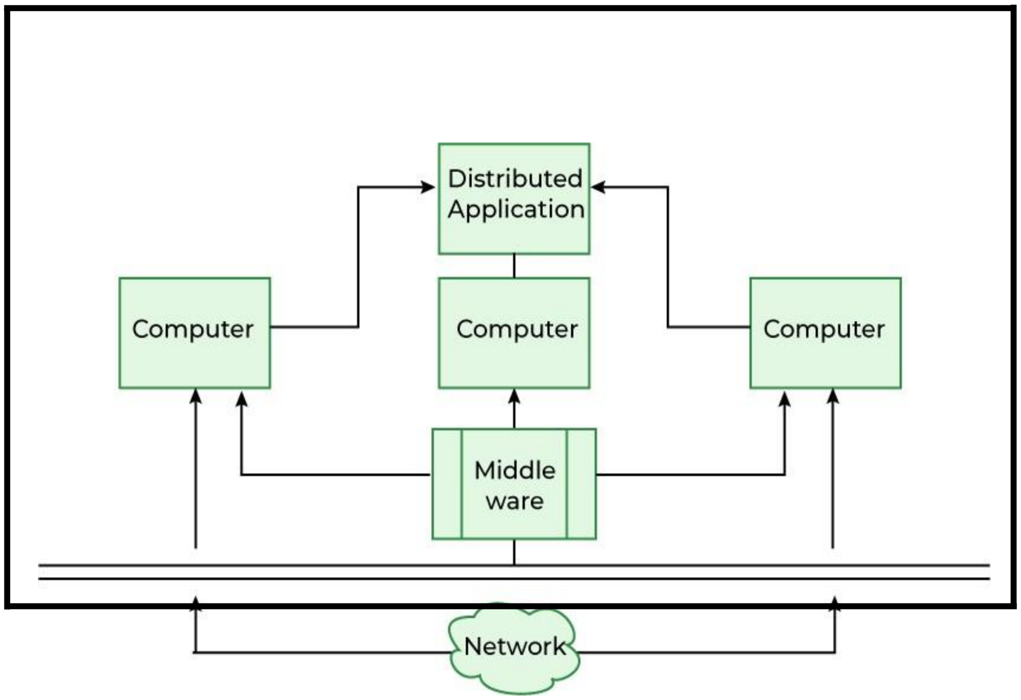
EXPERIMENT 3

Aim: Implement a program for Group Communication Theory:

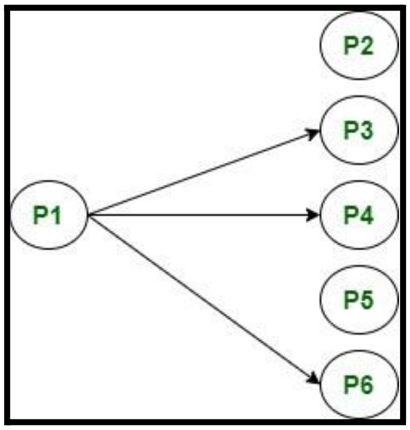
Communication between two processes in a distributed system is required to exchange various data, such as code or a file, between the processes. When one source process tries to communicate with multiple processes at once, it is called Group Communication. A group is a collection of interconnected processes with abstraction. This abstraction is to hide the message passing so that the communication looks like a normal procedure call. Group communication also helps the processes from different hosts to work together and perform operations in a synchronized manner, therefore increasing the overall performance of the system.



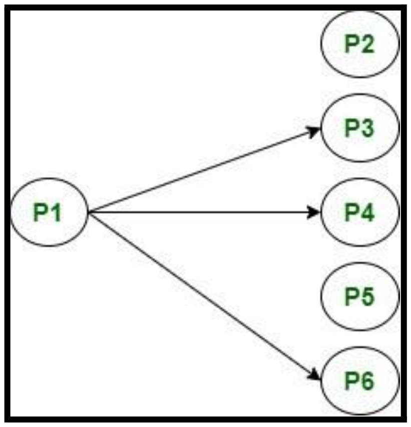


Types of Group Communication in a Distributed System:

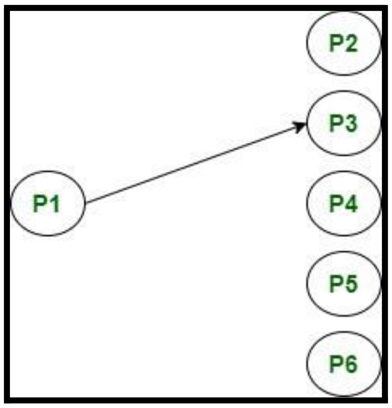
- Broadcast Communication: When the host process tries to communicate with every process in a distributed system at same time. Broadcast communication comes in handy when a common stream of information is to be delivered to each and every process in the most efficient manner possible. Since it does not require any processing whatsoever, communication is very fast in comparison to other modes of communication. However, it does not support a large number of processes and cannot treat a specific process individually.



- Multicast Communication: When the host process tries to communicate with a designated group of processes in a distributed system at the same time. This technique is mainly used to find a way to address the problem of a high workload on the host system and redundant information from processes in the system. Multitasking can significantly decrease time taken for message handling.



- Unicast Communication: When the host process tries to communicate with a single process in a distributed system at the same time. Although, the same information may be passed to multiple processes. This works best for two processes communicating as only it has to treat a specific process only. However, it leads to overheads as it has to find the exact process and then exchange information/data.



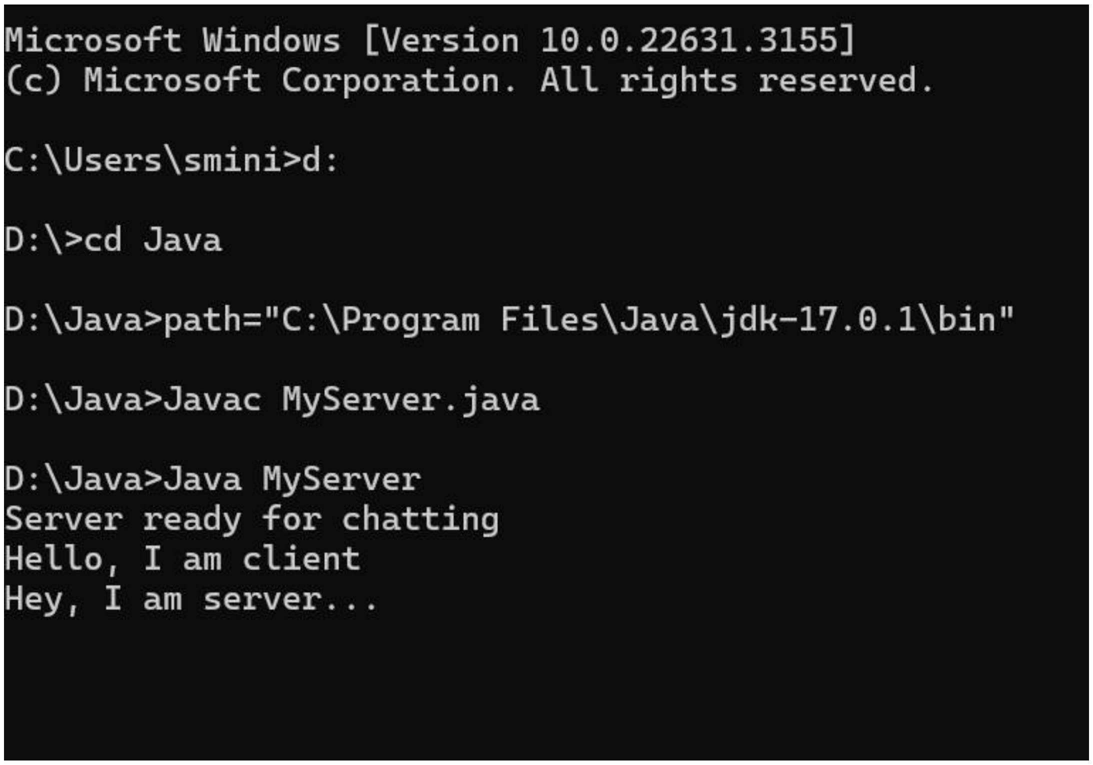
Code :

```
import java.io.*; import java.net.*;
public          class          MyServer{
@SuppressWarnings("resource")
public static void main(String args[]) throws Exception{
    ServerSocket ss = new ServerSocket(3000);
    System.out.println("Server ready for chatting");

    Socket so = ss.accept();
    BufferedReader keyread = new BufferedReader(new InputStreamReader(System.in));
    OutputStream os = so.getOutputStream();

    PrintWriter pWriter = new PrintWriter(os,true);
    InputStream iStream = so.getInputStream();
    BufferedReader receiveRead = new BufferedReader(new
InputStreamReader(iStream));
    String receiveMessage, sendMessage;
     while(true){
        if((receiveMessage = receiveRead.readLine())!=null){
            System.out.println(receiveMessage);
        }
         sendMessage       =       keyread.readLine();
pWriter.println(sendMessage);
pWriter.flush();
    }
}
}
```

```
import     java.io.*;     import
java.net.*;      public      class
MyClient{

@SuppressWarnings("resource")
public static void main(String args[]) throws Exception{      Socket sock =
new Socket("192.168.137.1",3000);
    BufferedReader keyReader = new BufferedReader(new
InputStreamReader(System.in));
    OutputStream oStream = sock.getOutputStream();
    PrintWriter pWriter = new PrintWriter(oStream,true);
    InputStream iStream = sock.getInputStream();
    BufferedReader receiveRead = new BufferedReader(new
InputStreamReader(iStream));
    System.out.println("Start the communication, type and press enter key");        String
receiveMessage, sendMessage;     while (true) {         sendMessage = keyReader.readLine();
pWriter.println(sendMessage);         pWriter.flush();

        if ((receiveMessage = receiveRead.readLine())!=null) {
            System.out.println(receiveMessage);
        }
    }
}
}
```

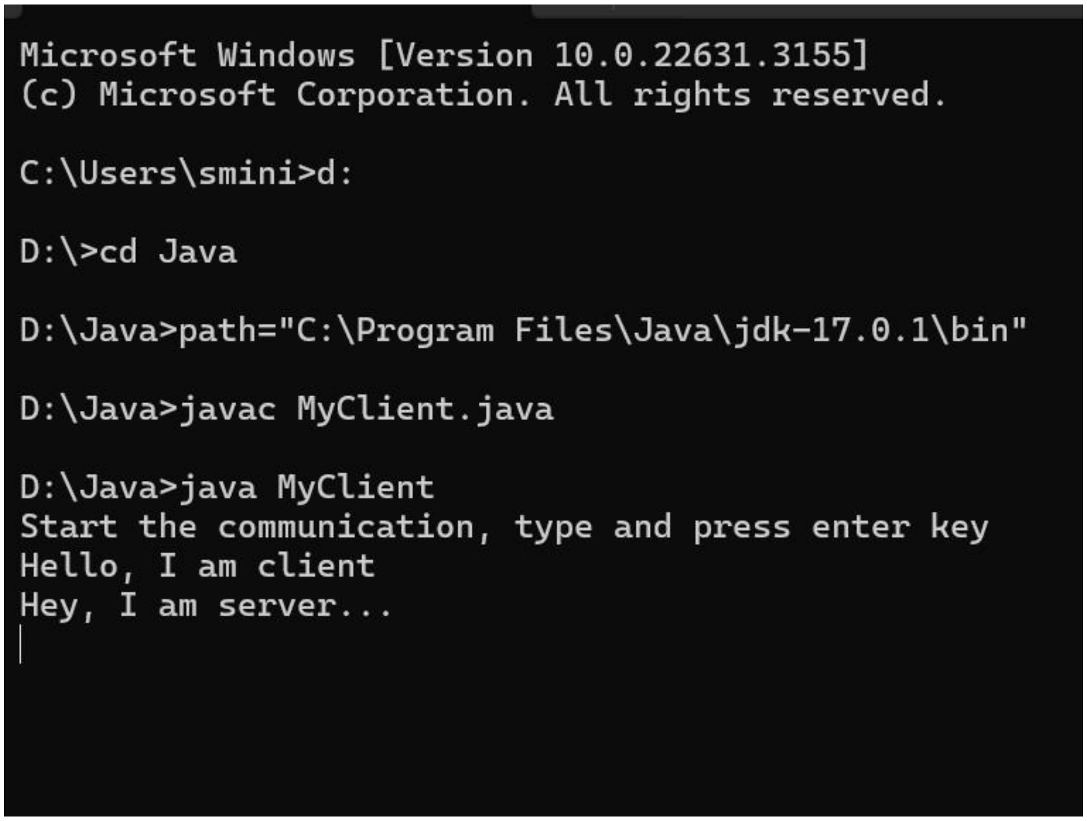
Output:

Server side:



```
Microsoft Windows [Version 10.0.22631.3155]
(c) Microsoft Corporation. All rights reserved.

C:\Users\smini>d:

D:\>cd Java

D:\Java>path="C:\Program Files\Java\jdk-17.0.1\bin"

D:\Java>Javac MyServer.java

D:\Java>Java MyServer
Server ready for chatting
Hello, I am client
Hey, I am server...
```

Client Side:



```
Microsoft Windows [Version 10.0.22631.3155]
(c) Microsoft Corporation. All rights reserved.

C:\Users\smini>d:

D:\>cd Java

D:\Java>path="C:\Program Files\Java\jdk-17.0.1\bin"

D:\Java>javac MyClient.java

D:\Java>java MyClient
Start the communication, type and press enter key
Hello, I am client
Hey, I am server...
```