

Name:

Department. : *Computer Engineering*

Class & Semester: *BE (Fourth Year), SEM VIII*

Subject: *Distributed Computing Lab (DCL)*

Expt. No. 02

Title: *Write a Program to demonstrate Interprocess
Communication in Client-Server Environment
Using Java.*

Date:

Subject In-charge Sign:

Experiment No. 02

Aim: Write a Program to demonstrate Interprocess Communication in Client-Server Environment using Java.

Software: jdk 1.8.0, Eclipse.

Pre-Lab Questions:

1. What is Inter-process communication?
2. What are the models of IPC?

Post-Lab Questions:

1. Differentiate between synchronous and asynchronous communication?
2. What is event synchronization?

Theory:

Interprocess communication (IPC) refers specifically to the mechanisms an operating system provides to allow processes it manages to share data. Typically, applications can use IPC categorized as clients_and_servers, where the client requests data and the server responds to client requests. Many applications are both clients and servers, as commonly seen in distributed computing.

Interprocess communication differs from communication among threads within the same program in several ways:

1. Different processes execute in different address spaces. (They have separate heaps.)
2. Different processes may execute on different machines.
3. Different processes may execute in different administration domains (and may not trust each other.

These differences have certain implications for the way communication must occur between different processes.

1. Local memory addresses are meaningless on the "other side" if transmitted. So, any data must be passed by value or if a reference is required, special mechanisms must be in place to resolve the references (they can't be ordinary addresses.)
2. The network must be involved when processes live on different machines
3. Care must be taken to ensure that sensitive data is not compromised by untrusted applications involved in the communication (security) or by applications snooping on the wire (encryption).

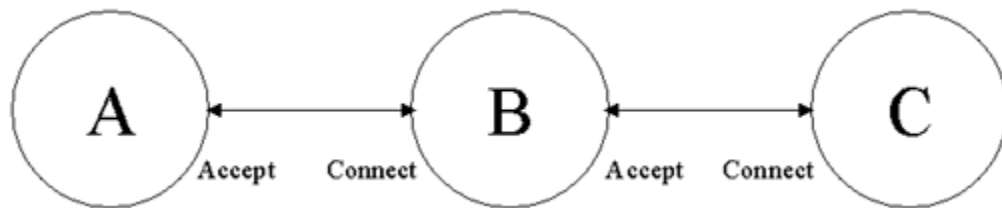
Let's start by spending some time discussing briefly how communication occurs in the network. Then we'll see how JAVA supports using this form of communication. We'll only touch on the security issue in passing.

If we take a simplified view of the internet, it might look something like this:

- A collection of hosts (computers) at the edges of the network, where
- Each computer is connected to a **local area network (LAN)**, and
- Every host has a unique address (called an **IP address -- Internet Protocol address**)

Transmission Control Protocol (TCP)

- A sender transmits packets, but also saves them on the side in a buffer
- When a receiver gets a packet, it sends an acknowledgement (ACK) to the sender for that packet.
- If the sender doesn't get an ACK after a certain period of time, it retransmits the packet.
- To improve **throughput** multiple packets (say n) are sent before waiting for the first ACK and when the first ACK comes back, the next packet (n+1) can be sent and so on. This is called a "sliding window protocol".
- Packets are delivered to the receiving application in the same order they were sent by the sender. (FIFO)



Here, B is a client of A and a server for C.

Program:

Implementation of Interprocess Communication in Client-Server

IPCServer.java

```
import java.net.*;
import java.io.*;
public class IPCServer
{
    public static void main(String args[])
    {
        System.out.println("\n **** INTERPROCESS COMMUNICATION****\n");

        System.out.println("\n **** SERVER PROCESS STARTED ****\n");

        System.out.println("\n **** SERVER IS READY AND WAITING TO
RECEIVE DATA FROM CLIENT PROCESS ON PORT****"+1200);
        try
        {
            ServerSocket ss =new ServerSocket(1200); //1200 is port number
            Socket clientSocket=ss.accept();

            System.out.println("\n **** Client is connected with IP
address****"+clientSocket.getInetAddress() +"and port
Number"+clientSocket.getPort());
            DataOutputStream dos =new
DataOutputStream(clientSocket.getOutputStream());

            DataInputStream dis =new
DataInputStream(clientSocket.getInputStream());
            int a = dis.readInt();
            System.out.println("\n SERVER RECEIVED");
            System.out.println("\n NUMBER 1 =====>" +a);
            int b =dis.readInt();
            System.out.println("\n NUMBER 2=====>" +b);
            int c=a+b;
            dos.writeInt(c);
            System.out.println("\n SERVER PROCESS HAS EXECUTED
REQUESTED PROCESS AND SENT RESULT "+c+"TO THE CLIENT\n");
            clientSocket.close();
            System.out.println("\n SERVER PROCESS
EXITING.....");
            ss.close();
        }
        catch(Exception e)
        {
            System.out.println("Exception: "+e);
        }
    }
}
```

IPCClient.java

```
import java.net.*;
import java.io.*;
public class IPCClient
{
    public static void main(String args[])
    {
        try
        {
            Socket s= new Socket("localhost",1200);
            DataOutputStream dos = new
DataOutputStream(s.getOutputStream());
            DataInputStream dis= new
DataInputStream(s.getInputStream());
            InputStreamReader isr = new InputStreamReader(System.in);
            System.out.println("\n \t***** CLIENT PROCESS
STARTED*****");
            System.out.println("\n *****PLEASE ENTER THE VALUES OF
NUMBER 1 AND NUMBER 2 TO PASS THEM TO SERVER PROCESS*****\n");
            BufferedReader br= new BufferedReader(isr);
            int a = Integer.parseInt(br.readLine());
            System.out.println("Number 1==>" +a);
            dos.writeInt(+a);
            int b = Integer.parseInt(br.readLine());
            System.out.println("Number 2====>" +b);
            dos.writeInt(+b);
            int result = dis.readInt();
            System.out.println("\n.....CLIENT PROCESS
HAS RECEIVED RESULT FROM SERVER.....\n");
            System.out.println("\n THE ADDITION OF "+a+"IS"+result);
            s.close();
        }
        catch(Exception e)
        {
            System.out.println("Exception is "+e);
        }
    }
}
```

Output:

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\student>cd\
C:\>d:
D:\>cd pds
D:\pds>set path="C:\Program Files\Java\jdk1.7.0_17\bin"
D:\pds>javac *.java
D:\pds>_
```

```
C:\WINDOWS\system32\cmd.exe - java IPCServer
D:\pds>javac *.java
D:\pds>java IPCServer
**** INTERPROCESS COMMUNICATION****

**** SERVER PROCESS STARTED ****

* SERVER IS READY AND WAITING TO RECEIVE DATA FROM CLIENT PROCESS ON PORT*1200
_
```

```
C:\ C:\WINDOWS\system32\cmd.exe - java IPCClient

D:\pds>java IPCClient

***** CLIENT PROCESS STARTED*****

*PLEASE ENTER THE VALUES OF NUMBER 1 AND NUMBER 2 TO PASS THEM TO SERVER PROCES
S*
```

Server Side

```
C:\ C:\WINDOWS\system32\cmd.exe

D:\pds>java IPCServer

**** INTERPROCESS COMMUNICATION****

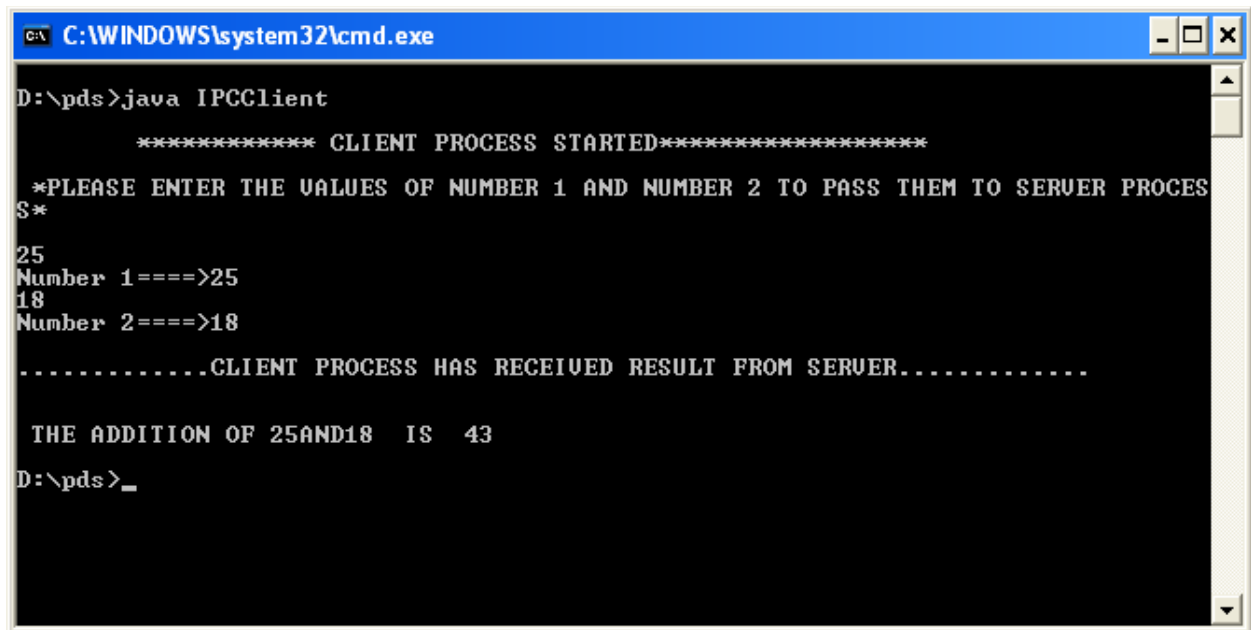
**** SERVER PROCESS STARTED ****

* SERVER IS READY AND WAITING TO RECEIVE DATA FROM CLIENT PROCESS ON PORT*1200
**** Client is connected with IP address****/127.0.0.1and port Number1350
SERVER RECEIVED
NUMBER 1====>25
NUMBER 2====>18
SERVER PROCESS HAS EXECUTED REQUESTED PROCESS AND SENT RESULT 43TO THE CLIENT

SERVER PROCESS EXITING.....

D:\pds>_
```

Client side



```
C:\WINDOWS\system32\cmd.exe

D:\pds>java IPCClient

***** CLIENT PROCESS STARTED*****

*PLEASE ENTER THE VALUES OF NUMBER 1 AND NUMBER 2 TO PASS THEM TO SERVER PROCESS*

25
Number 1====>25
18
Number 2====>18

.....CLIENT PROCESS HAS RECEIVED RESULT FROM SERVER.....

THE ADDITION OF 25AND18 IS 43

D:\pds>_
```

Conclusion: Thus we have demonstrated Interprocess Communication in Client-Server Environment using Java.