MCA Semester- II
Subject: Operating System & Shell Programming
Paper Code: CS2T06

IPC in Client –Server Systems

Some mechanisms that facilitate remote communications are as follows.

Sockets

Sockets is defined as an end point of the communication path between two processes. Each of the communicating processes creates a socket and these sockets are to be connected to enable the communication. Socket is identified by a combination of IP address and the port number . The IP address is used to identify the machine on the network and the port number is used to identify the desired service on that machine.

Usually a machine provides a variety of services such as: electronic mail, Telnet, FTP. Etc. Sockets employ client-server architecture. The server listens to a sockets bound to a specific port for a client to make connection request.

Remote Procedure Calls (RPC)

RPC, is a communication on mechanism that allows a process to call a procedure on a remote system connected via network. The calling process (client) can call the procedure on the remote host (server) in the same way as it would call the local procedure. The syntax of RPC call is very similar to conventional procedure call as given below:

Call <Procedure_id> (<List of parameters>);

The RPC system facilitates the communication between client and server by providing a stub on both client and server. For each remote

procedure, the RPC system provides a separate stub on the client side. When the client process wants to invoke a remote procedure the RPC call is implemented in the following steps.

1. The RPC system invokes the stub for the remote procedure on the client, passing to it the parameters that are to be passed further to the remote procedure. The client process is suspended from execution until completion of the call.
2. The client stub performs parameter marshalling which involves packaging the parameters into a machine-independent form so that they can be transmitted over the network. It now prepares a message containing the identifier of the procedure to be executed and the marshaled parameters.
3. The client sub sends the message to the server. After the message has been sent, the client stub blocks until it gets reply to its message.
4. The corresponding stub on the server side receives the message and converts the parameters into a machine-specific form suitable for the server.
5. The server stub invokes the desired procedure, passing parameters to it. The server stub is suspended from execution until completion of the call.
6. The procedure executes and the results are returned to the server stub.
7. The server stub converts the results into a machine –independent form and prepares a message.
8. The server stub sends the message containing the results to the client stub.
9. The client stub converts the results into machine-specific form suitable for client.
10. The client stub forwards the results to the client process. With this, the execution of RPC is completed and now, the client process can continue its execution.

Remote Method Invocation (RMI)

RMI is java-based approach that facilitates remote communication between programs written in the java programming language. It allows an object executing in on Java Virtual Machine (JVM) to invoke methods on an object executing in another Java Virtual Machine either on the same computer or on some remote host connected via network.

To enable the communication between client and server using RMI, the remote methods must transparent both to client and the server. For this RMI implements the remote objects using stubs and skeletons. A stub is a client-side proxy for a remote object while a skeleton is the server-side proxy for the remote object. On the clent-side, the stub acts on behalf of the actual remote object.

Faculty: Ms Sushmita Chakraborty
Assistant Professor
Deptt. of MCA, Patna Women's College