



Unit-III

# Remote Procedure Calls



IPC gained popularity because of following reasons;

1. Simple Call Syntax.
2. Familiar Semantics.
3. Its specification of a well-defined interface.
4. Its ease of use.
5. Its generality.
6. Its efficiency.
7. It can be used as an IPC mechanism to communicate between processes on different machines as well as between different processes on the same machine.



The RPC model is similar to the well-known and well-understood procedure call model used for the transfer of control and data within a program in the following manner;

1. To make a procedure call
2. Control transfer
3. Procedure body execution
4. Returning control

The RPC mechanism is an extension of the procedure call mechanism in the sense that it enables a call to be made to a procedure that does not reside in the address space of the calling process. The called procedure( commonly called remote procedure) may be on the same computer or on the different computer.



# RPC Model...[Contd..]

Therefore the mechanism of RPC is;

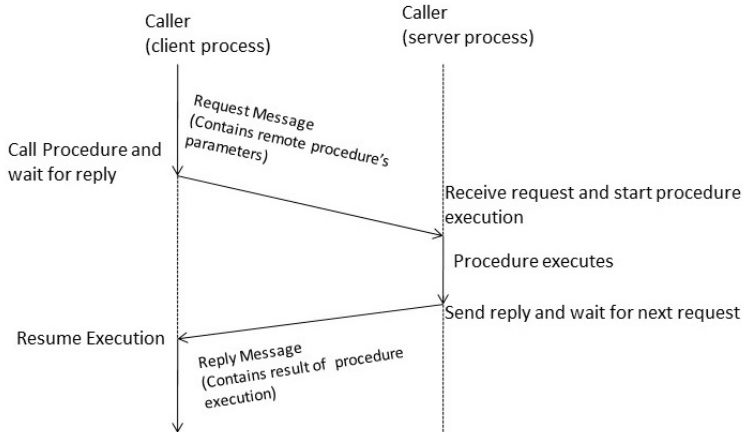


Figure: RPC Model

# Transparency of RPC



A transparent RPC mechanism is one in which local procedures and remote procedures indistinguishable to the programmers. This requires the following two types of transparencies

- ▶ Syntactic Transparency
  - ▶ Semantic Transparency
- 
- Syntactic transparency are easy
  - Semantic Transparency are not easy

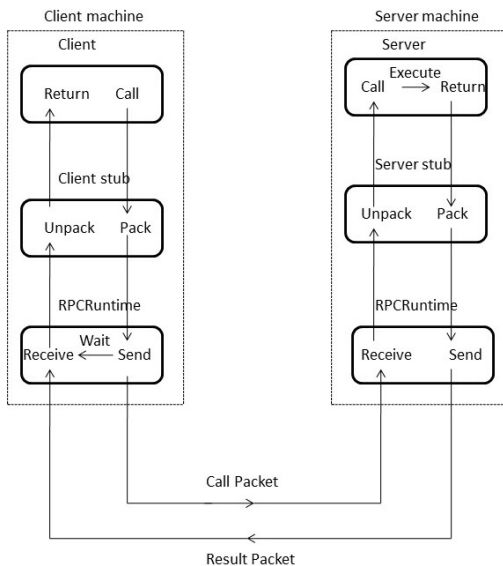
# Implementation of RPC



Implementation of RPC mechanism usually involves the following five elements of program.

1. Client
2. Client Stub
3. RPC Run time
4. Server Stub
5. Server

# Implementation of RPC...[Cntd..]



**Figure:** Implementation of RPC mechanism



# RPC Messages

There are two types of messages involved in RPC implementation.

1. Call Messages
2. Reply Messages

## 1. Call Message

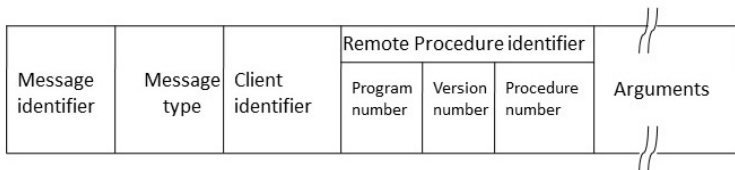


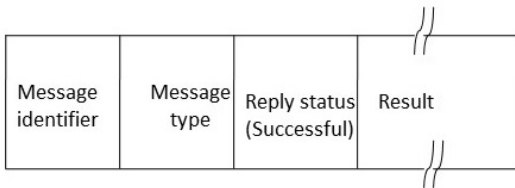
Figure: RPC Call Message Format



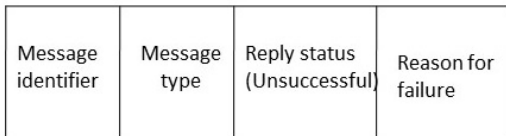
# RPC Messages...[Cntd...]



## 2. Reply Message



(a)



(b)

**Figure:** (a) A Successful Reply Message Format (b) An Unsuccessful Message Format

# Marshaling Arguments and Results



For RPC, encoding and decoding of message data is called Marshaling and involves following actions.

1. Taking the arguments
2. Encoding the message data
3. Decoding the message data

# Server Management



In IPC-based applications, two important issues that need to be considered for server management;

1. Server Implementation
2. Server Creation

# Server Management...[Cntd..]



## 1. Server Implementation

Based on the style of implementation used, servers may be of two types

- ▶ Stateful Servers
- ▶ Stateless Servers

# Server Management...[Cntd..]

## 1.1 Stateful Server

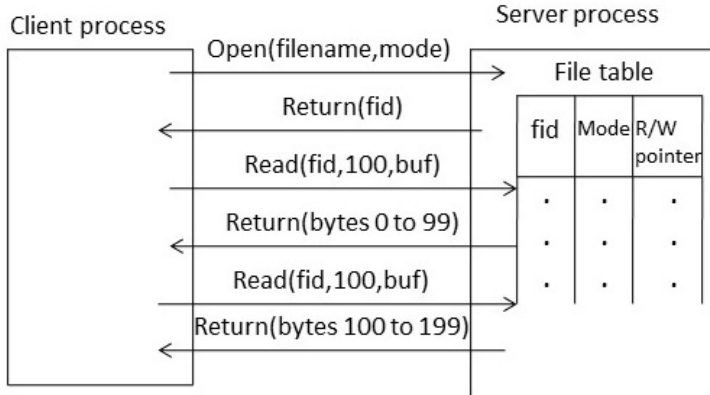


Figure: Example of a Stateful File Server

# Server Management...[Cntd..]



## 1.2 Stateless Servers

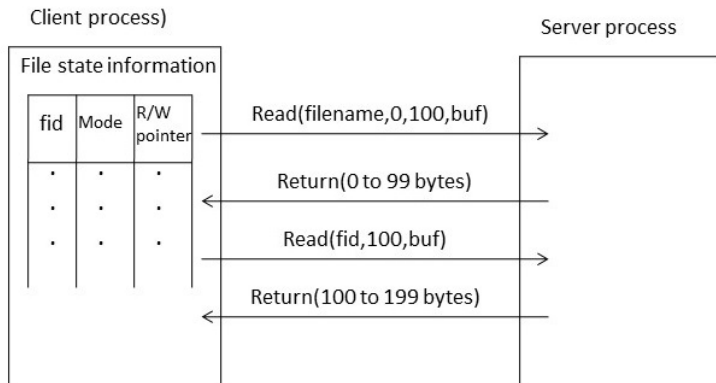


Figure: Example of a Stateless File Server



## 2. Server Creation Semantics

Based on the time duration for which RPC server survive, they may be classified as;

- ▶ Instance-per-Call Servers
- ▶ Instance-per-Session Servers
- ▶ Persistent Servers

# Parameter-Passing Semantics



The choice of parameter-Passing semantics is a crucial to the design of an RPC mechanism. Followings are the choices of parameter passing;

- ▶ Call-by-Value
- ▶ Call-by-Reference





In RPC, the caller and callee processes are possibly located on different nodes. Thus it is possible for either the caller or callee node to fail independently and later to be restarted. In addition, failure of communication links are also possible, therefore, the normal functioning of an RPC may get disrupted due to following reason;

1. The call message gets lost
2. Response message gets lost
3. The callee node crashes and is restarted
4. The caller node crashes and is restarted

# Call Semantics..[Cntd...]



The different types of call semantics used in RPC system are;

1. Possibly or May-be Call Semantics
2. Last-one Call Semantics
3. Last-of-Many Call Semantics
4. At-Least-Once Call Semantics
5. Exactly-Once Call Semantics

# Communication Protocols for RPC's



Based on the needs of different systems, several communication protocols have been proposed for use of RPC's. Followings are the protocols;

1. The Request Protocol
2. The Request/Reply Protocol
3. The Request/Reply/Acknowledge-Reply Protocol

# Communication Protocols for RPC's...[Cnts...]



## 1. The Request Protocol

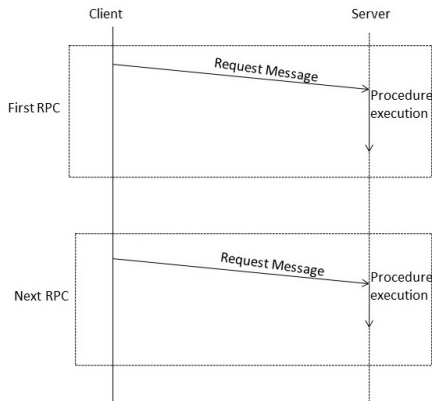


Figure: Request(R) Protocol

# Communication Protocols for RPC's...[Cnts...]



## 2. The Request/Reply Protocol

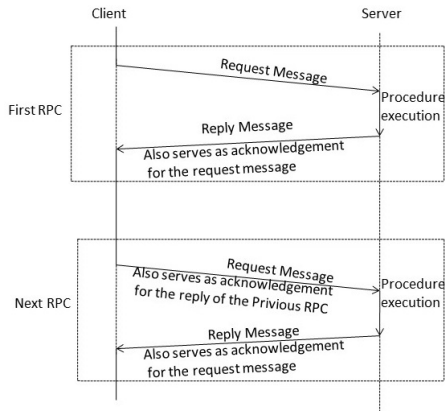
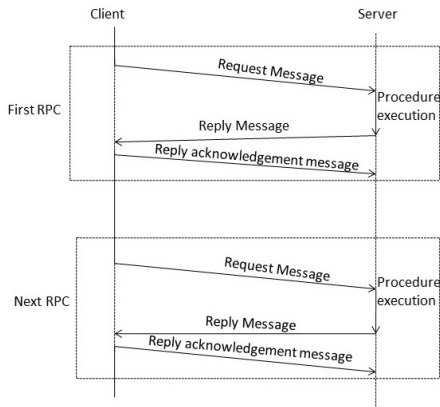


Figure: Request/Reply (RR) Protocol

## 3. The Request/Reply/Acknowledge-Reply Protocol



**Figure:** The Request/Reply/Acknowledge-Reply (RRA) Protocol

# Complicated RPCs



Following two types of RPCs are complicated;

- ▶ RPCs involving long duration calls or large gaps between calls
- ▶ RPCs involving arguments and /or Results that are too large to fit in a single datagram packet.

# Client-Server Binding



The client-server binding process involves proper binding of several issues;

1. Server Naming
2. Server Locating
3. Binding Time
4. Changing Binding
5. Multiple Simultaneous Bindings



# Client-Server Binding...[Cntd...]



## 1. Server Naming

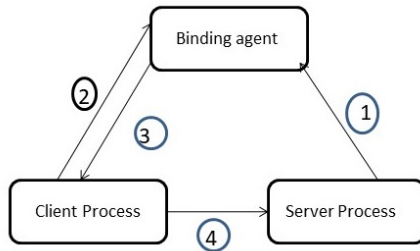
- ▶ How does a client specify a server to which it wants to get bound?
- ▶ Interface Name: a type and an Instance

# Client-Server Binding...[Cntd...]

## 2. Server Locating

To locate server, two commonly methods used

- ▶ Broadcasting
- ▶ Binding Agent



**Figure:** Binding Agent Mechanism for locating a server

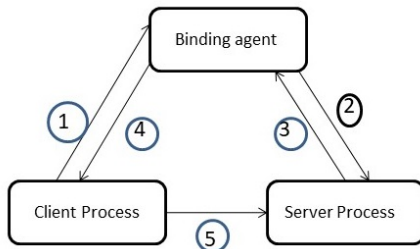
# Client-Server Binding...[Cntd...]



## 3. Binding Time

A Client may be bound to a server at

- ▶ Binding at Compile Time
- ▶ Binding at Link Time
- ▶ Binding at Call Time



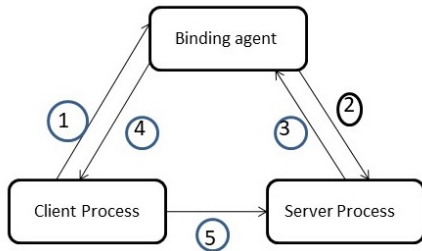
**Figure:** Binding at Call Time by the method of Indirect Call

# Client-Server Binding...[Cntd...]

## 4. Multiple Simultaneous Binding

In a system, a server may be provided by multiple servers. In general, a client is bound to a single server of the several servers of the same time.

- ▶ Binding at Compile Time
- ▶ Binding at Link Time
- ▶ Binding at Call Time

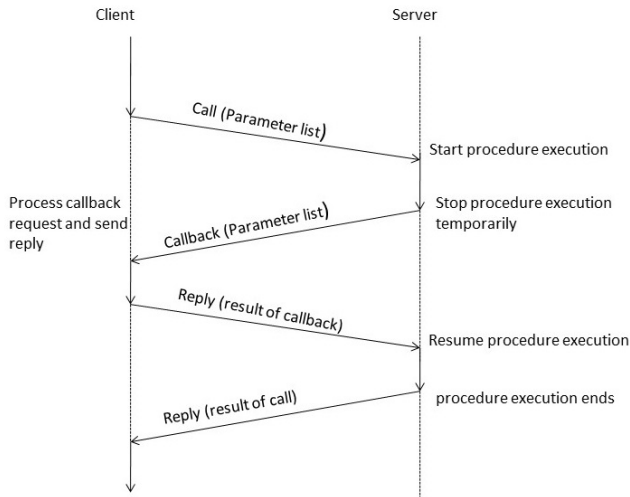


**Figure:** Binding at Call Time by the method of Indirect Call



# Some Special Types of RPC's

## 1. Callback RPC



**Figure: Callback RPC**  
Govt. College of Engg, Jalgaon

# Some Special Types of RPC's...[Cntd...]

To provide callback RPC facilities, followings are necessary

- ▶ Providing the Server with the Client's Handle
- ▶ Making the Client Process Wait for the Callback RPC
- ▶ Handling Callback Deadlocks

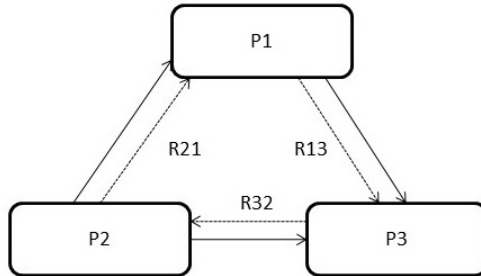


Figure: Callback Deadlock

# Some Special Types of RPC's...[Cntd...]



## 2. Broadcast RPC

A broadcast RPC mechanism may use one of the following two methods for broadcasting a clients request;

- ▶ A special broadcast primitives
- ▶ To declare broadcast ports

# Some Special Types of RPC's...[Cntd...]



## 3. Batch-Mode RPC

- ▶ mode RPC is used to queue separate RPC request in a transmission buffer on the client side and then send them over the network in one batch to the server.
- ▶ However, batch-mode RPC can be used only with those applications in which a client has many RPC requests to send to a server and the client does not need to reply for a sequence of requests.
- ▶ Therefore, the requests are queued on the client side and the entire queue of requests is flushed to the server.



# RPC in Heterogeneous Environment



When designing an RPC system for a heterogeneous environment, the three common types of heterogeneity that need to be considered

1. Data Representation
2. Transport Protocol
3. Control Protocol

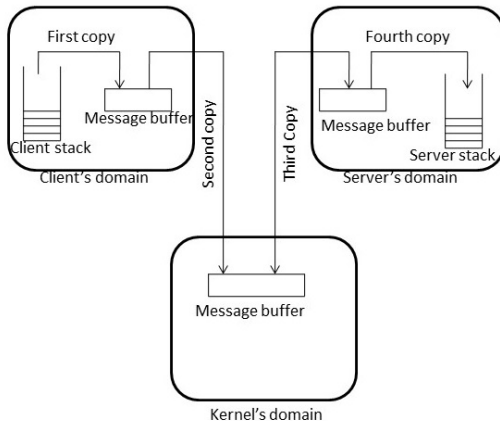
# Lightweight RPC



To achieve better performance than conventional RPC, following four techniques are used by LRPC

1. Simple Control Transfer
2. Simple Data Transfer
3. Simple Stubs
4. Design for Concurrency

# Lightweight RPC...[Cntd...]



(a)

Figure: (a) Traditional Cross Domain RPC

# Lightweight RPC...[Cntd...]

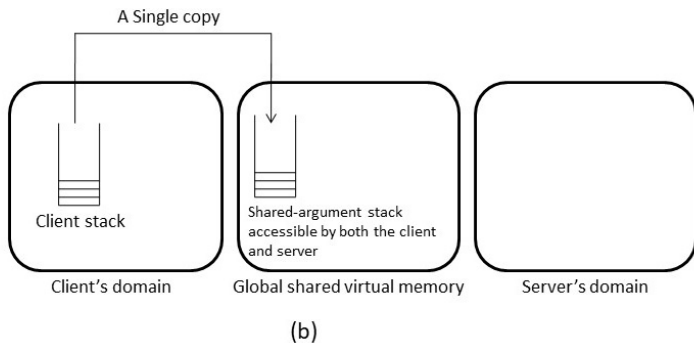


Figure: (b)LightWeight RPC