CASE STUDY NO. :- 03

Date	/ /20
Page No.	17

Case Study On :- Study of RPC and group communication.

Remote Procedure Call (RPC) -

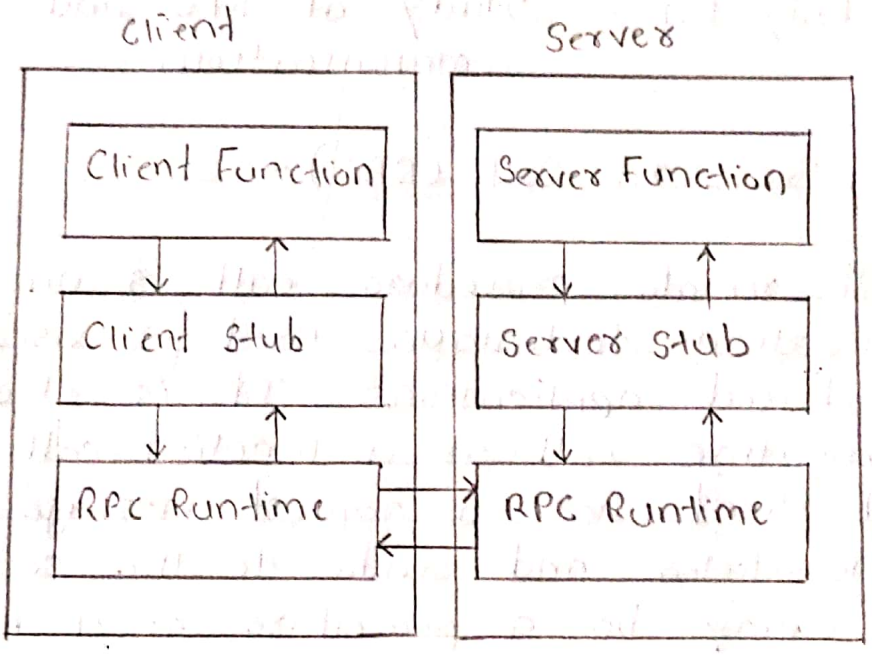
A remote procedure call is an interprocess communication technique that is used for client-server based applications. It is also known as a subroutine call or a function call.

A client has a request message that the RPC translates and sends to the server. This request may be a procedure or a function call to a remote server. When the server receives the request, it sends the required response back to the client. The client is blocked while the server is processing the call and only resumed execution after the server is finished.

The sequence of events in a remote procedure call are given as follows -

- The client stub is called by the client.
- The client stub makes a system call to send the message to the server and puts the parameters in the message.
- The message is passed to the server stub by the server operating system.
- The parameters are removed from the

A diagram that demonstrates this is as follows -



message by the server stub.
- Then, the server procedure is called by the server stub.

* Advantages of Remote Procedure Call-

Some of the advantages of RPC are as follows-

- Remote procedure calls support process oriented and thread oriented models.
- The internal message passing mechanism of RPC is hidden from the user.
- The effort to re-write and re-develop the code is minimum in remote procedure calls.
- Remote procedure calls can be used in distributed environment as well as the local environment.
- Many of the protocol layers are omitted by RPC to improve performance.

Disadvantages of Remote Procedure Call-

Some of the disadvantages of RPC are as follow -

- The remote procedure call is a concept that can be implemented in different ways. It is not a standard.
- There is no flexibility in RPC for hardware architecture. It is only interaction based.

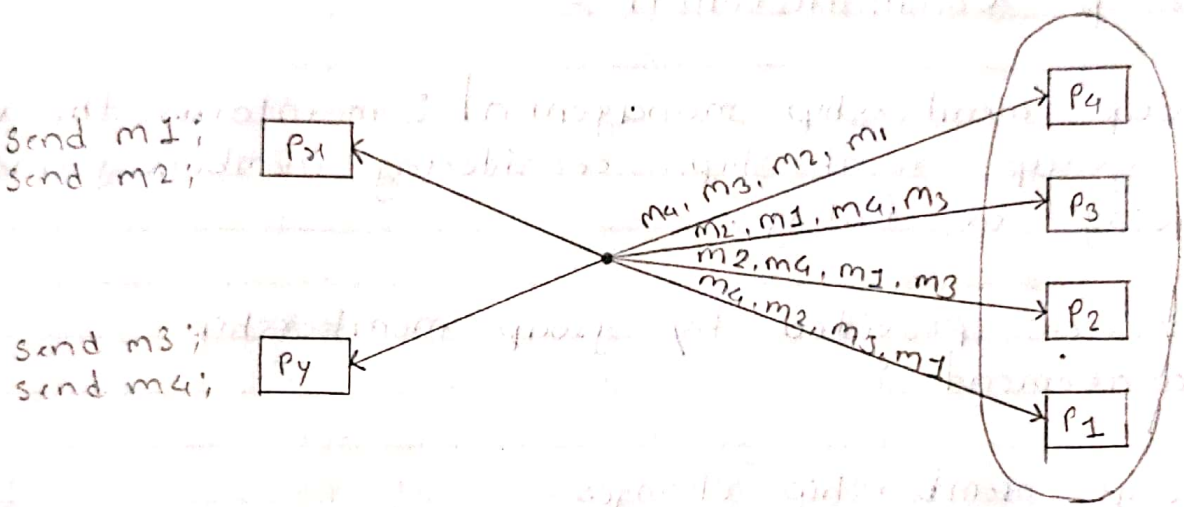
- There is an increase in costs because of remote procedure call.

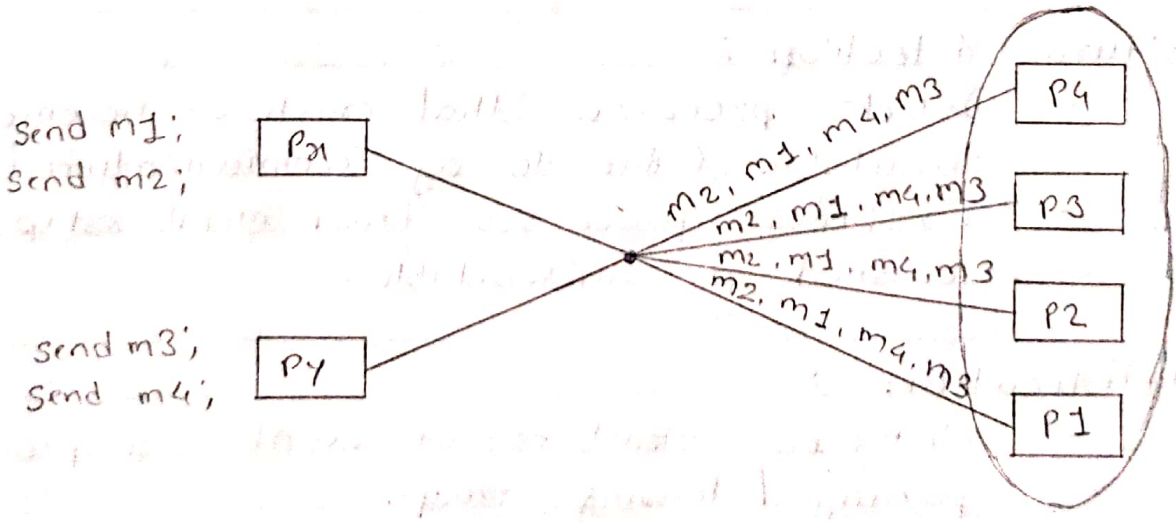
Group Communication -

Group membership management : maintains the view of group membership, considering members joining, leaving or failing.

Services provided by group membership management :

- Group membership changes :
  - create / destroy process groups ;
  - add / withdraw processes to / from group.
- Failure detection :
  - Detects processes that crash or become unavailable (due to e.g. communication failure);
  - Excludes processes from membership if crashed or unavailable.
- Notification :
  - Notifies members of events e.g. processes joining / leaving group.
- Group address expansion :
  - Processes sending to group specify group identifier ; address expansion provides the actual addresses for the multicast operation delivering the message to each group members.



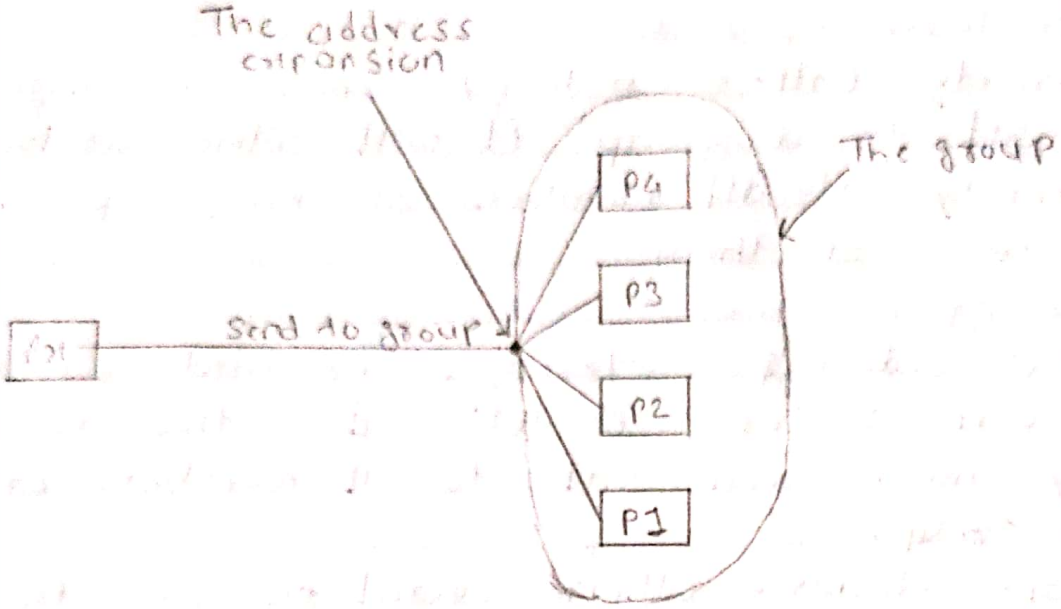


Essential Features :

- Atomicity (all-or-noting) : when a message is send to a group, it will either arrive correctly at all members of the group or at none of them.
- Ordering
  - FIFO ordering : Messages originating from a given sender are delivered in the order they have been send, to all members of the group.
  - Total-ordering : When several message, from different senders, are sent to a group, the messages, ~~from different senders~~, the messages reach all the members of the group in the same order.

Group Communication -

- The assumption with client-server communication and RMI (RPC) is that two parties are involved : The client and the server.
- Sometimes communication involves multiple processes, not only two. A solution is to perform separate message passing operations or RMIs to each receiver.
- With group communication a message can be send to a group and then it is delivered to all members of the group ⇒ multiple receivers in one operation.



Why do we need it?

- Special applications: interest-groups, mail-lists, etc.
- Fault tolerance based on replication: a request is sent to several servers which all execute the same operation (if one fails, the client still will be served).
- Locating a services or object in a distributed system: the client sends a message to all machines but only the one (or those) which holds the server/object responds.
- Replicated data (for reliability or performance) whenever the data changes, the new value has to be send to all processes managing replicas.