* Assignment No :- 02 *

Q.1] Explain Implementation of RPC in detail with diagram.

→ - RPC is an effective mechanism for building client - server systems that are distributed.

- Remote procedure call (RPC) enhance power & ease of programming of the client / server computing concept.

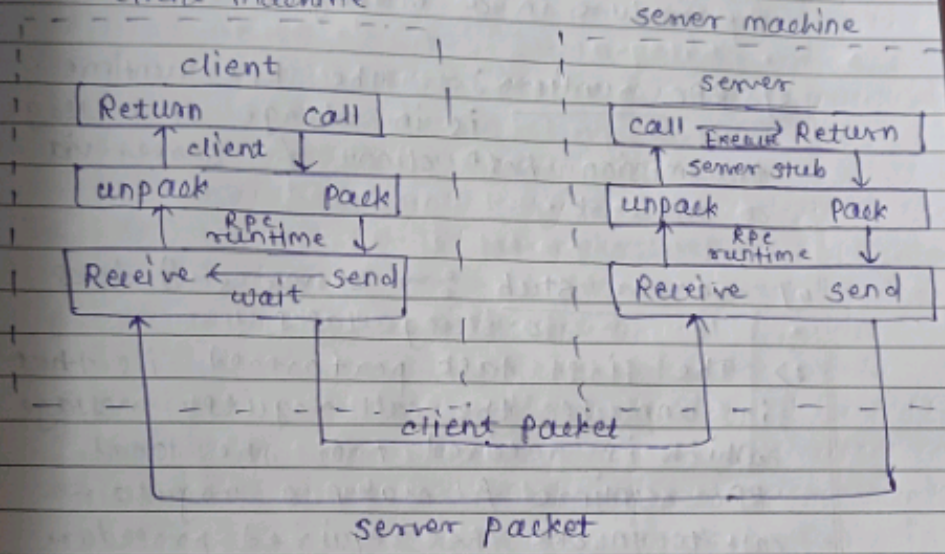


fig. RPC Architecture

- RPC has five elements :-
  1] client
  2] client stub
  3] RPC runtime
  4] server stub
  5] server

1] client :- The client process initiates RPC. client makes a standard call, which triggers a correlated procedure in the client stub.

2] client stub :- stubs are used by RPC to achieve semantic transparency. the client calls the client stub.

- client stub does the following task:-

1> The first task performed by client stub is when it receives a request from a client, it packs the parameters & required specifications of remote/target procedure in a message.

3] RPC runtime :- The RPC runtime is in charge of message transmission betn client & server via a network.

4] server stub :- server stub does following tasks :

1> The first task performed is that it unpacks the call request message which is received from the local RPC runtime & makes a regular call to invoke the required procedure in the server.

5] server :- After receiving a call request from client machine, the server stub passes it to the server.

Q.2] What are the desirable features of good process migration.

→ • Features Of good process migration :-

1> Transparency
2> Efficency
3> Reliability
4> Scalability
5> security
6> Flexibility

7] Transparenc

& the system, continue to without ne

2] Efficienc

in terms of memory, n

3] Reliability to migration, s state or original

4] scalab

large-s nodes.

- The st across

5] se

protecte transf

- The authe ensu sou

2024.08.23 09:56

4) Scalability
5) security
6) Flexibility

1] Transparency :- The migration should be transparent to the user & the system, meaning the process should continue to run as if nothing has changed, without needing modifications.

2] Efficiency :- The migration process should introduce minimal overhead in terms of both time & resource. (CPU, memory, network bandwidth).

3] Reliability :- There should be mechanisms to recover from failures during migration, such as rollback to the original state or restarting the process on the original machine.

4] scalability :- The migration mechanism should work effectively in large-scale distributed system with many nodes.
- The state of the process should be consistent across the source & destination machines.

5] security :- Sensitive data associated with the process should be protected during migration, with secure data transfer & access control.
- The process migration should involve proper authentication & authorization checks to ensure that it is initiated by trusted sources & executed on trusted nodes.

2024.08.23 09:56

7. Flexibility :- The migration system should be capable of handling various types of processes including cpu-bound, I/O bound & real-time process.

Q.3] Demonstrate with example how to keep track of lost & out of sequence packets in a multi-datagram message transmission.

→ • Datagram :- A datagram is a basic transfer unit associated with a packet-switched n/w & provide a connectionless commn service across a packet-switched n/w.
- Each packet is known as datagram

• How much data at a time we can transmit?
- Almost all n/w have an upper bound on the size of data that can be transmitted at a time. the size is known as the maximum transfer unit (MTU) of a network.
- The message whose size is greater than MTU fragramented into multiples of the MTU & each fragment sent separately.

• Steps to track lost & out-of sequence packets;

1. Assign sequence numbers;
- Each packet is assigned a unique sequence no. before being sent.

- For example, if we have a file that is divided into 5 packets, the sequence no. would be 1, 2, 3, 4, 5.

2. Transmit packet :-
- The sender transmits the packets sequentially over the n/w.

3. Receive & Track packets:
- The receiver maintains a list to track received packets. Each entry corresponds to a sequence no. & can be marked as "received" or "missing".
- e.g. A list with the expected sequence no. [1, 2, 3, 4, 5].

4. Check for out of sequence packets:
- As packets arrive, the receiver checks their sequence no.
- If a packet arrives with a sequence no. higher than the next expected no, the receiver marks the intermediate packet as "missing".
- eg. if packet 1 & then packet 3 are received, packet 2 is marked as missing.

5. Handle lost packets :-
- The receiver can send a Negative Acknowledgment (NAck) or a request for retransmission for any missing packets.
- e.g. if packet 3 arrives before packet 2, the receiver might send a NAck for packet 2.

6. Reconstruct the message :-
- The receiver waits until all packets have been received & if necessary, requests

retransmission for any lost packets.
- once all packets are received, they are recorded according to their sequence no. to reconstruct the original message.

- Example :-

1) packet sent :- sequence no. [1,2,3,4,5]

2) packet received :- let's say the receiver gets packets in the following order 1,3,2,5

3) Receiver's list (Tracking) :-

- Initial state : [1: missing, 2: missing, 3: missing, 4: missing, 5: missing].

- After receiving packet 1 : [1: received, 2: missing, 3: missing, 4: missing, 5: missing]

- After receiving packet 3 : [1: received, 2: missing, 3: received, 4: missing, 5: missing] (packet 2 is identified as missing)

- After receiving packet 2 : [1: received, 2: received, 3: received, 4: missing, 5: missing] (packet 2 arrives, list updated)

- After receiving packet 5 : [1: received, 2: received, 3: received, 4: missing, 5: received] (packet 4 is missing)

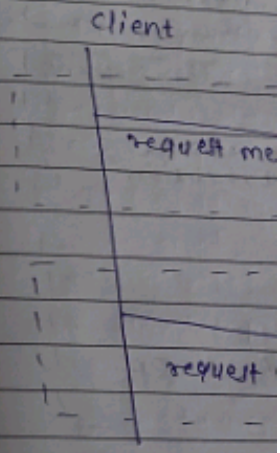
4) Request transmission
- The receiver sends a NACK for packet 4.

5) Reconstruct messa
- After receiving t
the receiver has
2: received, 3: re
- The packets are
sequence no. &
reconstructed.

Q.4] Explain com
RPCs.
→ • Communication

1) Request proto
2) Request / reply
3) The Request / repl
protocol.

1) Request pro



fi

- The requ
R protoc

- It is us
when a res

2024.08.23 09:56

5) Reconstruct message :-
- After receiving the retransmitted packet 4, the receiver has the complete set [1: received, 2: received, 3: received, 4: received, 5: received].
- The packets are ordered according to their sequence no. & the original message is reconstructed.

Q.4] Explain communication protocols for RPCs.

→ • Communication protocols for RPC's :

1) Request protocol
2) Request / reply protocol
3) The Request / reply / Acknowledgement - reply protocol.

1) Request protocol :-

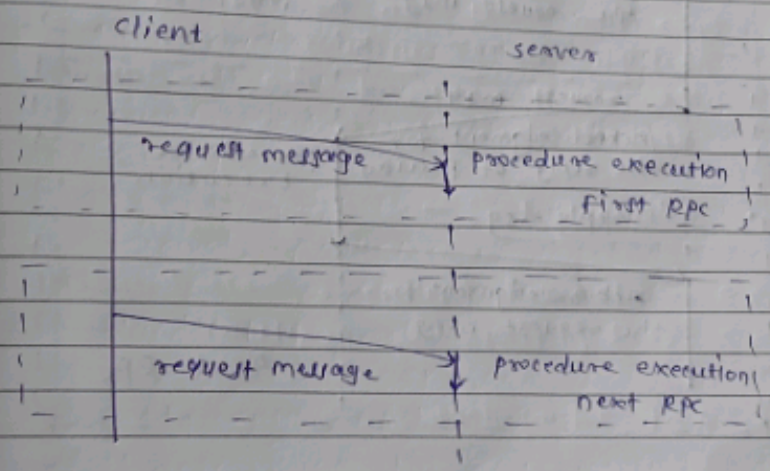


fig.: request protocol

- The request protocol is also known as the R protocol.

- It is used in remote procedure call (RPC) when a request is made from the calling

procedure to the called procedure. After execution of the request, a called procedure has nothing to return & there is no confirmation required of the execution of a procedure.

- Because there is no acknowledgement or reply message, only one message is sent from client to server.

2) Request / Reply protocol :-

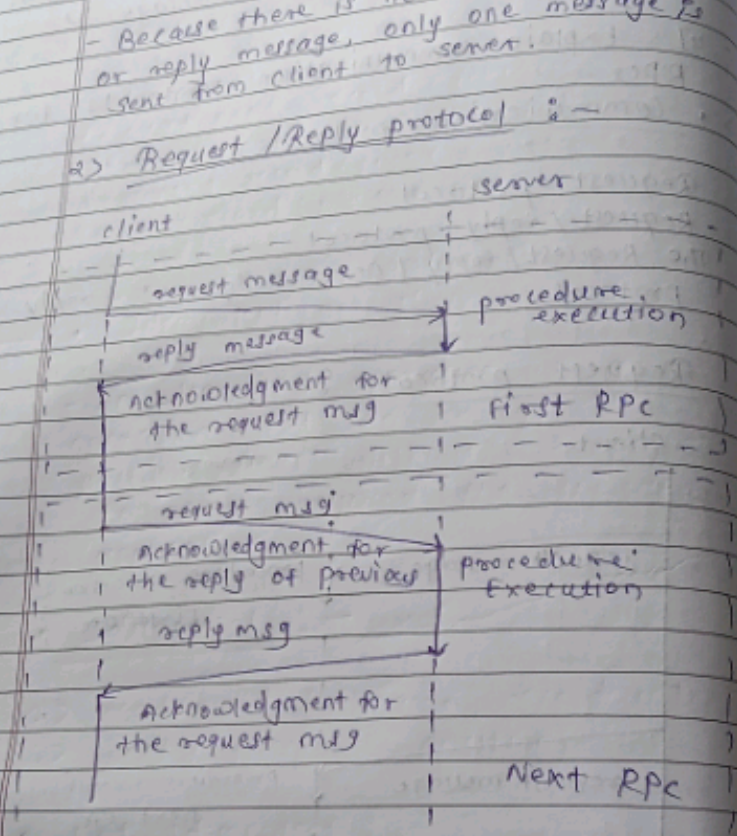


fig. Request/reply protocol

- The Request-Reply protocol is also known as the RR protocol.

- It works well for systems that involve simple RPC's.

- The parameter & resu
in a single packet b
the duration of the
calls are both brie

- This protocol has a
implicit acknowledge
acknowledgements.

3) Request / Reply / Ackne

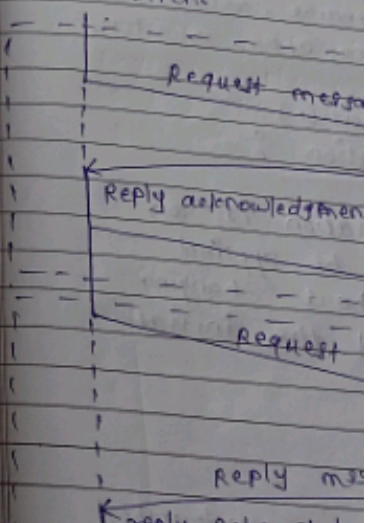


fig. request/re

- This protocol
protocol (requ

- The RRA prot
the drawbacks
protocol.

- The parameter & result values are enclosed in a single packet buffer in simple RPCs. the duration of the call & the time both calls are both brief.

- This protocol has a concept base of using implicit acknowledgements instead of explicit acknowledgements.

a) Request/Reply/Acknowledgement-Reply protocol:-

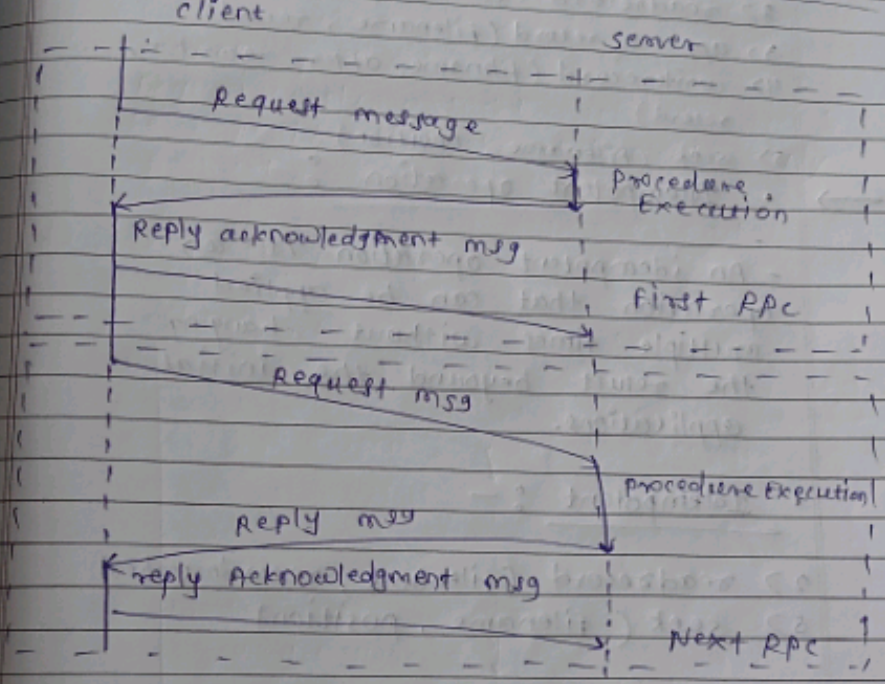


fig. request/reply/Ack-reply protocol

- This protocol is also known as the RRA protocol (request/reply/Acknowledgment-reply).

- The RRA protocol is used to get rid of the drawbacks of the RR (request/reply) protocol.

- Exactly-once semantics is provided by RR protocol which refers to the responses getting held in reply cache of servers resulting in loss of replies that have not been delivered.

Q.57 What is an idempotent operation? Which of the following operations are idempotent;

1) read-next-record (filename)
2) readrecord (filename, recordnumber)
3) append-record (filename, record)
4) writerecord (filename, after-record-n, record)
5) seek (filename, position)

→ • Idempotent operation :-

- An idempotent operation is an operation that can be applied multiple times without changing the result beyond the initial applications.

• Idempotent :-

2) readrecord (filename, recordnumber)
5) seek (filename, position)

• Not Idempotent :-

1) read-next-record (filename)
3) append-record (filename, record)
4) writerecord (filename, after-record-n, record)

2024.08.23 09:56